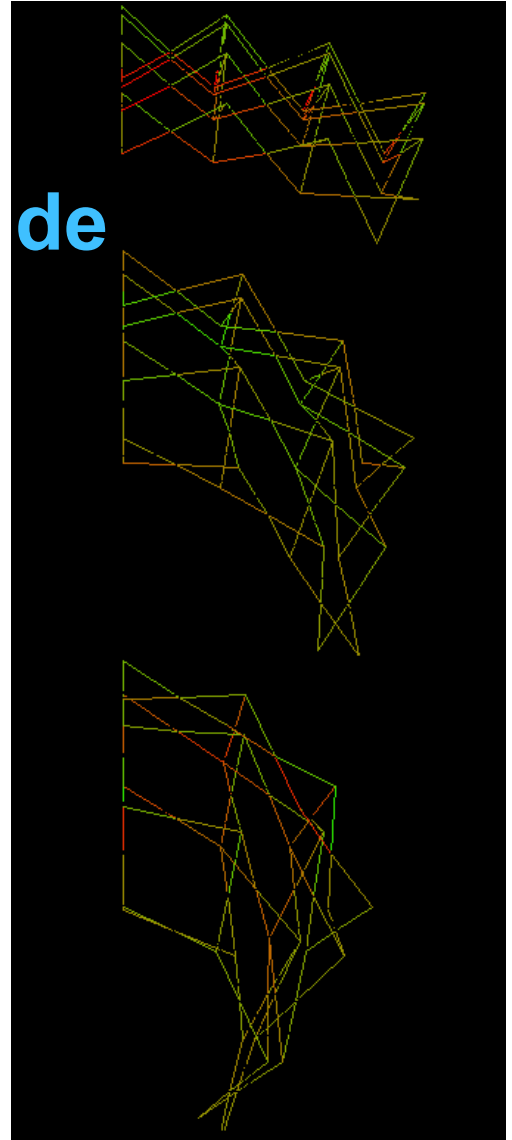


Tesis Doctoral.

Universidade da Coruña

Año 2001

# Cálculo Dinámico de Estructuras Desplegables.



Autor:

**Manuel Muñoz Vidal**

Director tesis:

Enrique López Hernández

Tesis Doctoral.

Universidade da Coruña

Año 2001

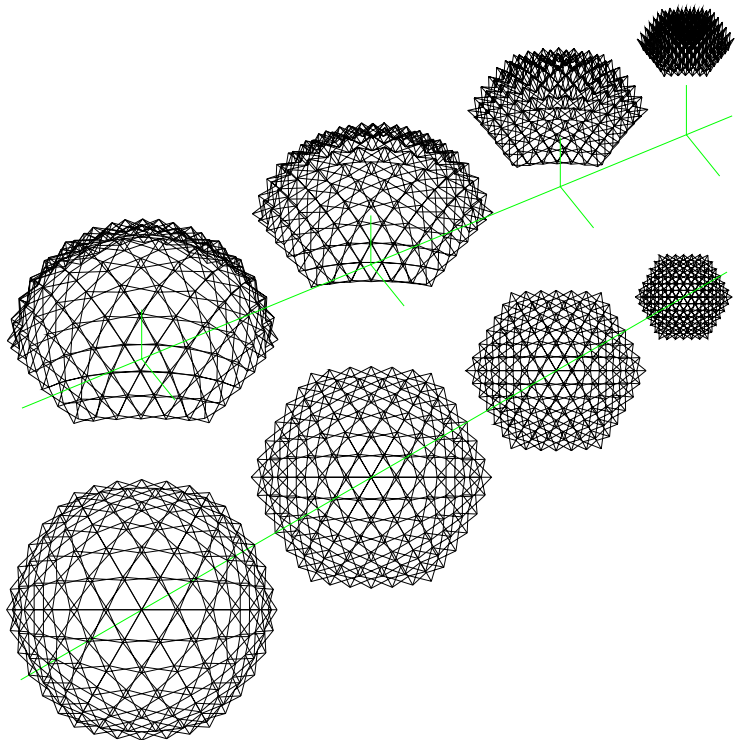
# Cálculo Dinámico de Estructuras Desplegables.

Autor:

**Manuel Muñoz Vidal**

Director tesis:

Enrique López Hernández



**Agradecimientos:**

*A mi director de tesis, Enrique López Hernández, quien sentó las bases de lo que sería este trabajo, sin cuyo apoyo nunca me hubiese introducido en el terreno del cálculo dinámico. Gracias por saber encauzar estos estudios en la dirección correcta.*

*A Juan Pérez Valcarcel, director del departamento de Tecnología da Construcción, por su perseverancia y apoyo. Gracias por las correcciones efectuadas y tiempo dedicado.*

*A mi familia, y en especial a mi tía, a quien espero darle una gran alegría con la finalización de este trabajo.*

*A todos muchas gracias.*

*A Miluca, Marta, Jorge y ... , por su cariño y comprensión.*



---

# Indice

---

# Indice.

## 1. Las Mallas Desplegables

<b>1.1 INTRODUCCIÓN HISTÓRICA Y DESCRIPCIÓN DEL TIPO ESTRUCTURAL.</b>	2
Estructuras modernas .....	8
Referencias .....	14
Bibliografía .....	14
<b>1.2 GEOMETRÍA DE LAS MALLAS DESPLEGABLES</b>	15
▪ 1.2.1 Diseño y Construcción .....	15
▪ 1.2.2 Geometría del Despliegue .....	24
▪ 1.2.3 Clasificación tipológica .....	28
▪ 1.2.4 Mallas Compatibles e Incompatibles. ....	33
Mallas Cilíndricas. ....	33
Mallas Esféricas. ....	35
Referencias .....	40
<b>1.3 CÁLCULO ESTRUCTURAL</b>	41
▪ 1.3.1 Cálculo Matricial .....	41
Barras con articulación central fija .....	42
Matriz de rigidez en coordenadas locales. ....	42
Matriz de compatibilidad. ....	43
Matriz de rigidez en coordenadas globales. ....	44
Cálculo de esfuerzos .....	45
Comprobación de resultados. ....	45
▪ 1.3.2 Consideraciones de Segundo Orden .....	46
Efecto del axil en la flexión. ....	46
Efecto de la excentricidad de los nudos .....	48
Consideración de barras flexionadas inicialmente. ....	49
Grandes deformaciones. ....	50
Proceso de cálculo. ....	50
Otras soluciones constructivas .....	51
Referencias .....	52

## 2. Metodología

<b>2.1 OBJETIVOS</b>	54
Consideraciones previas	54
▪ 2.1.1 Cálculo Dinámico	55
Despliegue	55
Análisis de resultados.	56
▪ 2.1.2 Generación de Mallas.	57
Referencias:	58
<b>2.2 METODOLOGÍA</b>	59
▪ 2.2.1 Análisis Dinámico	59
Introducción.	59
Equilibrio estático y equilibrio dinámico.	60
Análisis Estático	61
Análisis Armónico Estacionario	61
Análisis Modal.	62
Análisis de vectores de Ritz	63
Análisis del Espectro de Respuesta	63
Análisis en el Tiempo.	64
Análisis no lineal en el Tiempo	64
▪ 2.2.2 Modelo Matemático	66
Ecuaciones de comportamiento	66
Sistema de ecuaciones diferenciales	66
Diferencias finitas	66
Modelización de las barras.	74
Esfuerzos en la barra.	76
Variación de la rigidez por efecto del axil.	77
Movimiento cinético del sólido rígido.	81
Ecuaciones de movimiento aplicadas a una barra.	84
Resolución de las ecuaciones de movimiento.	86
Determinación de las masas e inercia.	87
▪ 2.2.3 Análisis de Resultados	92
Análisis global.	92
Análisis local.	93
Dominio del tiempo.	93
Dominio de las frecuencias.	94
▪ 2.2.4 Generación de mallas	100
Cúpula de módulos cuadrados.	100
Cúpula de módulos triangulares.	105
Referencias:	115

## 3. Validación

<b>3.1 BARRA SIMPLE</b>	117
Generalidades	117
Vibración longitudinal. Modos fundamentales	118
Vibración transversal. Modos fundamentales	118
Barra biapoyada	119
Barra libre	120
<b>3.2 CONJUNTO DE BARRAS.</b>	121
Modos propios de vibración.	121
Análisis de despliegue	126

## 4. Resultados.

<b>4.1 ESTUDIO DE INCOMPATIBILIDADES.</b>	129
▪ Introducción	129
Mallas estudiadas en este trabajo	130
La malla de módulo Cuadrangular	130
La malla de módulo Triangular	131
Verificación de incompatibilidades	132
Estudio del despliegue completo	135
Influencia de la geometría en las fuerzas	138
Conclusiones	139
<b>4.2 FENÓMENOS INERCIALES.</b>	140
Definición de la estructura	140
Resultados	141
Conclusiones	145
<b>4.3 OTRAS POSIBILIDADES</b>	146
▪ 4.3.1 Vibraciones y frecuencias propias	146
Malla Desplegable	146
Malla Espacial	147
Conclusiones.	148
▪ 4.3.2 Estructuras Tensadas	149
Métodos de Generación.	149
Generación de Estructuras Tensadas.	151
Conclusiones.	152
Referencias	153



## 5. Conclusiones.

<b>5.1 CONCLUSIONES.</b>	155
▪ Método de Cálculo. ....	155
▪ Análisis de Resultados. ....	156
▪ Estudio de Incompatibilidades. ....	156
▪ Generación de Mallas. ....	157
▪ Implementación Informática. ....	157

## 6. Bibliografía

<b>6.1 BIBLIOGRAFÍA PRINCIPAL</b>	160
▪ 6.1.1 Mallas Desplegables. ....	160
▪ 6.1.2 Cálculo Dinámico y Numérico. ....	175
<b>6.2 BIBLIOGRAFÍA SECUNDARIA</b>	180
▪ 6.2.1 Mallas Espaciales. ....	180
▪ 6.2.2 Cálculo Matricial y Elementos Finitos. ....	182

## 7. Anexos

<b>7.1 EL PROGRAMA ARTIC.</b>	189
▪ 7.1.1 generalidades. ....	189
GENERALIDADES	189
▪ 7.1.2 Uso del programa. ....	190
Estructura	191
Generación	194
Editar	196
Estática	200
Dinámica	202
▪ 7.1.3 ESTRUCTURA DE FICHEROS. ....	206
Definición de la estructura	206
Resultados del cálculo	210
<b>7.2 EL PROCESO DE GENERACIÓN DE MALLAS ESPACIALES</b>	212
▪ 7.2.1 Generación	212
1) Lectura de los datos del módulo	213
2) Núcleo de la generación	213
▪ 7.2.2 Definición de nuevos módulos	216

Descripción de un módulo. ....	216
Datos numéricos del módulo. ....	216
▪ 7.2.3 Descripción de los módulos predefinidos .....	220
<b>7.3 LISTADOS DEL PROGRAMA.</b>	231
7.3.1 Módulo principal de cálculo dinámico. ....	231
7.3.2 Módulo de Análisis Global .....	253
7.3.3 Módulo de Análisis Local .....	261
7.3.4 Generación Malla Desplegable Cuadrados. ....	275
7.3.5 Generación Malla Desplegable Triangulos. ....	280
7.3.6 Generación Mallas Espaciales. ....	287
7.3.7 Módulo de Cálculo Matricial. ....	291
7.3.8 Módulo de Edición de la Estructura. ....	302
7.3.9 Módulo de Edición de las Cargas Dinámicas. ....	324
7.3.10 Módulo de Cálculo dinámico en Fortran .....	330

## LAS SIMULACIONES NUMÉRICAS

Si bien todo este trabajo se basa en la simulación de fenómenos reales mediante cálculos numéricos intensivos, debemos recalcar de entrada que detrás de todo cálculo habrá una persona que deberá discernir y conocer los límites de validez de todo el proceso realizado y de los resultados obtenidos. Para mantener este espíritu crítico baste un par de párrafos de un artículo de Arturo Olvera (IIMAS-UNAM):

*"Una de las primeras personas en incursionar en esta nueva disciplina fue Enrico Fermi, pues tuvo la oportunidad de trabajar con una de las primeras computadoras modernas, la ENIAC. él ideó un experimento que consistía en simular la dinámica de una cadena de moléculas unidimensionales con potenciales del tipo cúbico. Fermi esperaba encontrar un comportamiento dinámico donde se cumpliera la hipótesis ergódica de Boltzmann al incluir sólo unas cuantas decenas de partículas. Para su sorpresa, la simulación numérica mostró un comportamiento recurrente y la distribución de energía distaba en mucho de ser equipartida. El dilema que se le presentó a Fermi fue determinar si esta simulación numérica reproducía correctamente el fenómeno físico. Tal cuestionamiento no ha sido exclusivo de Fermi, todos aquellos que realizan un cálculo numérico con la idea de entender un fenómeno físico tienen que discernir sobre la validez de sus resultados. Esperamos que la información obtenida reproduzca en forma correcta el comportamiento de nuestro problema real. Entonces, ¿qué debemos hacer para que sean plausibles nuestros resultados?"*

[ . . . . ]

*Considerando que los métodos numéricos son en sí una perturbación al sistema de ecuaciones, estos métodos pueden ser los responsables del aparente comportamiento caótico. Así también pueden ocultar fenómenos de inestabilidad cuando el método numérico genera una perturbación de tipo disipativo.*

*Podemos concluir que realizar una simulación numérica que represente correctamente la solución de alguna ecuación es un problema nada trivial y es indispensable tener un buen conocimiento de los algoritmos numéricos y sobre todo una idea del comportamiento cualitativo de las soluciones. Como hemos apuntado en la discusión anterior, es necesario estudiar la ecuación por otros métodos analíticos o asintóticos para poder hacer un pronóstico adecuado del tipo de solución que esperamos obtener y así elegir el método numérico más apropiado. Si somos capaces de realizar simulaciones numéricas correctas, habremos logrado añadir un elemento más al ciclo teoría-experimento, dando un potencial mayor a nuestra capacidad de entender los fenómenos de la naturaleza, tal como lo previó Fermi."*

# 1. Las Mallas Desplegables



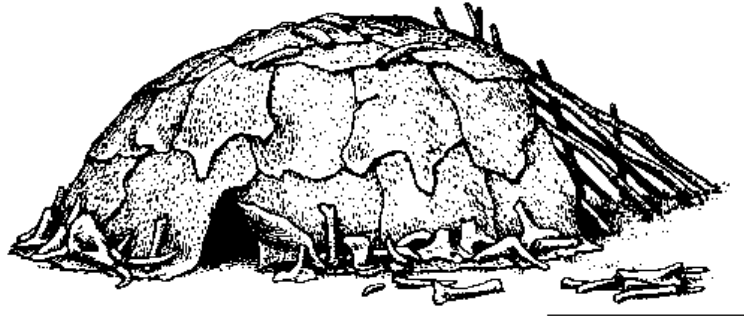
---

## 1.1 INTRODUCCIÓN HISTÓRICA Y DESCRIPCIÓN DEL TIPO ESTRUCTURAL.

---

Un breve recorrido histórico siempre nos dará una perspectiva más amplia del problema, a la vez que nos permitirá ir centrándonos poco a poco en el mismo.

Los precedentes de estructuras plegables y desplegadas no son muchos. En un principio suele relacionarse con los pueblos guerreros y nómadas de los albores de la historia, que precisaban construcciones ligeras de montaje y desmontaje fácil, siendo en un principio los materiales empleados pieles y cuerdas.

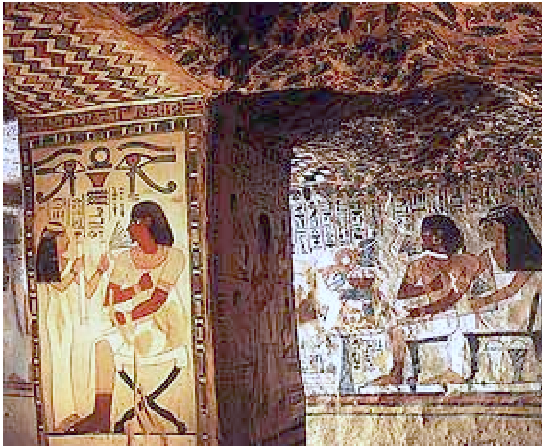


Las primeras tiendas del paleolítico estaban realizadas a base de pieles de animales sobre una estructura de ramas de árboles. Se han obtenido datos de tiendas de 44.000 años de antigüedad, con formas semiesféricas, evolucionando hacia las cónicas, que son usuales entre los pueblos nómadas del norte de América y Eurasia.

Los campamentos militares egipcios se realizaban con tiendas con estructuras sustentantes a base de cañas o ramas de palmeras. El material de cubrición y método de construcción no se conoce. En algunos casos las tiendas adoptaban formas arqueadas, como se pone de manifiesto en un relieve del 1.282 a.C. que representa un campamento de Ramsés II.



Los usos iniciales no siempre estaban relacionados con la construcción. Un ejemplo es la silla de tijera, de la que se ve un gráfico muy claro en una terracota babilónica de 2.000 a.C., que también se encuentra en numerosos dibujos egipcios, como en la tumba de Senefer (480 a.C.), de la época de Amenofis II (680 a.C.) [1]



De esta derivaría la 'silla egipcia' o 'silla de cazador', que consta de tres patas unidas por una articulación intermedia y coronada con una piel que sirve de asiento.

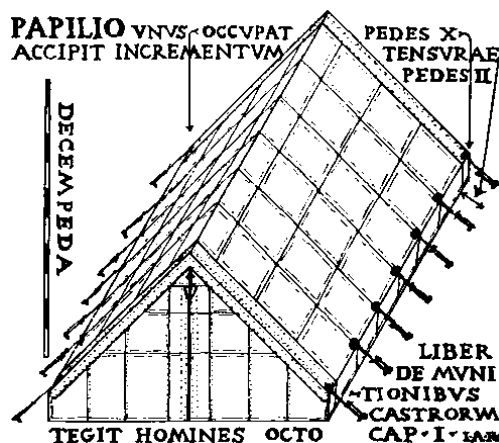
Otro objeto donde se da la plegabilidad es en la sombrilla, que ya se observa en relieves egipcios como el obelisco negro (820 a.C.) donde el rey Salmanasar III está cobijado por un parasol. Pero se ve con más detalle en el relieve de Assurbanipal, donde incluso se aprecian las varillas deslizantes.

Otro tipo de estructuras fuera de la construcción donde se dan nuestros objetos de estudio es en los velámenes de los barcos, donde hay gran superficie que cubrir y debe ser rápido de extender y recoger. En pinturas sobre piedra hallas en el sur de Suecia se observan redes que sirven de refuerzo a las velas de las embarcaciones vikingas. Los cartagineses desarrollaron una compleja tecnología para el despliegado de grandes superficies textiles para la impulsión de las tirremes.

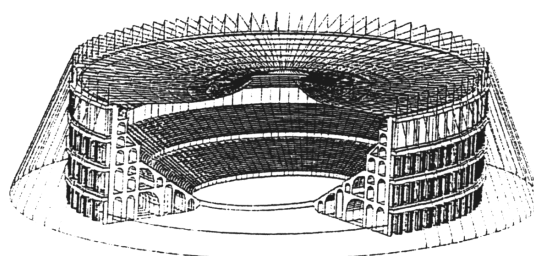
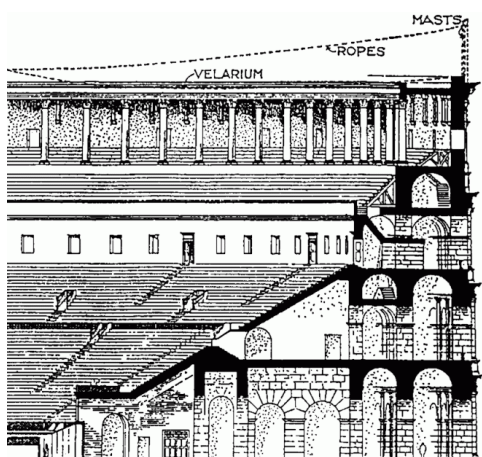


No se tienen datos directos de ésta, pero se conoce por que fue copiada ampliamente por los romanos, haciéndola extensiva incluso a las obras civiles.

En Roma, encontramos un muestra de transformabilidad en el uso de tiendas en los campamentos militares (las *papilio*), que podían albergar hasta una docena de soldados. Estamos ahora en el año 100 a.C.



También usaron la experiencia naval para cubrir los anfiteatros con velas de grandes dimensiones que protegían a los espectadores del sol. Al parecer incluso el Coliseo romano, con sus 25.000 m<sup>2</sup> (187 y 155 m. en cada dirección) tenía un velarium que se encargaban de extender y recoger un numeroso grupo de marinos del puerto de Ostia a quienes se llamaba ex-profeso para esta labor.



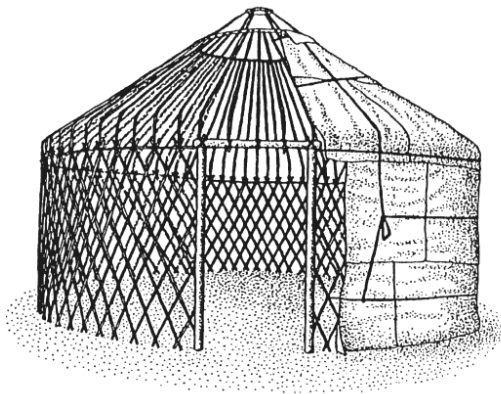
Las tiendas laponas del este de Siberia adoptan una forma cónica, con mástiles rectos perimetrales pero auxiliados por un par de arcos interiores que dan más consistencia a la estructura. Los esquimales del norte de Alaska tuvieron un talento especial para eliminar los materiales no esenciales, sobre todo la madera, que escasea en estas latitudes, llegando a estructuras sustentantes muy simples.

En los pueblos indios de América del Norte sus tiendas cónicas se desarrollaron, aparte de servir de morada, hacia el objetivo de conseguir en su interior una combustión eficiente en un clima frío y ventoso. La cubrición se realiza cosiendo pieles de bisonte, y el patrón es prácticamente una semicircunferencia con unas aletas en la zona que formará la abertura de la chimenea. Estas alas se sujetarán con unas varas que permitirán controlar la apertura del agujero.

Es fácil y rápida de montar y desmontar, y una vez desmontada, los mástiles hacen las veces de un carro sin ruedas que desliza sobre el suelo arrastrado por un caballo [3].



En el siglo XIX, en Asia central la evolución de las tiendas nómadas alcanza un punto culminante, con la construcción denominada Kibitka o Yurta, usada por los pueblos Turcos y Mongoles (conocida aquí como Yurta) [6]. Esta construcción ya tiene una estructura sustentante bastante estándar y depurada, sobre la que se dispone una cubrición que es de una importancia secundaria y menor durabilidad. Es sumamente sólida, resistente y puede transportarse en camello y una familia la puede montar en media hora.

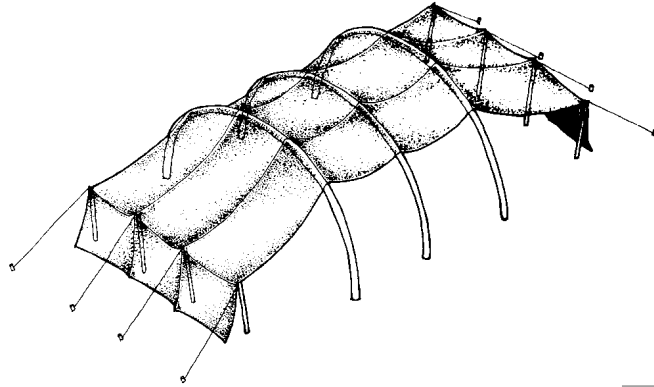


Las tiendas de los pueblos nómadas norteafricanos son las que empiezan a dar mayor importancia a la cubrición, que realizada a base de tejidos reforzados por una serie de bandas, con lo admite bastante bien la tracción y permite minimizar la necesidad de estructura sustentante interior, que queda reducida a una serie de puntales de madera.

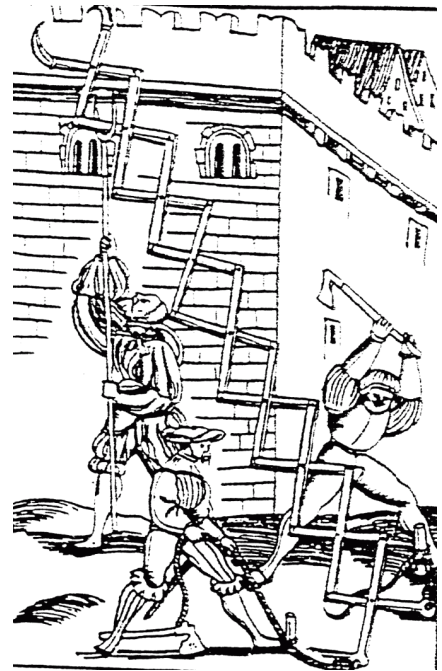




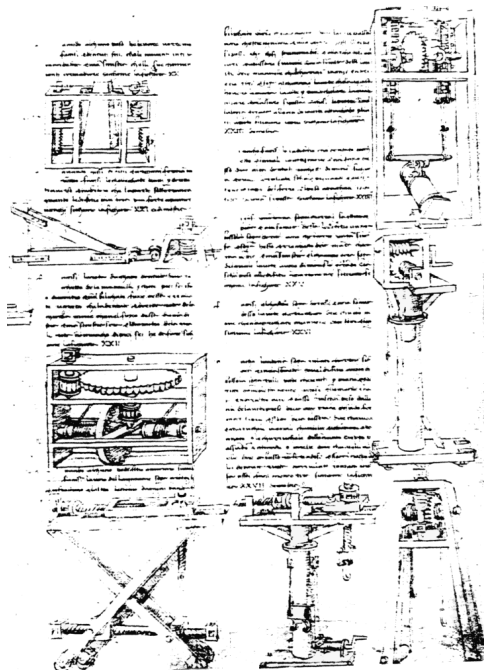
Las tiendas nómadas de Afganistán adoptan una forma de bóveda en barril, forma que se consigue dotando a la estructura sustentante de madera con unos arcos formados por ramas curvadas, lo que les da una forma muy característica.



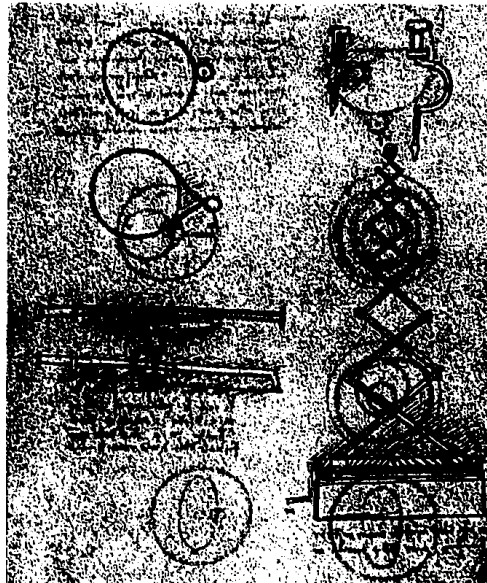
Más cercanamente se van perfilando los mecanismos y acercándose a elementos de despliegue relativamente rápido. En la figura podemos ver el grabado de una máquina de asedio medieval en la que ya se observa claramente el mecanismo de despliegue, muy similar al de las mallas que estudiaremos más adelante.



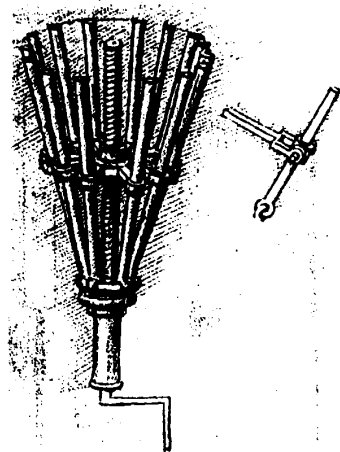
Las estructuras de aspa también se utilizaron en esta época para levantar pesos, como se observa en el códice Laurantiano de Francisco de Giorgio. [2]



Leonardo da Vinci (1452-1519) también realiza un estudio de estos temas como muestra un dibujo del códice de Madrid I, que representa una máquina para levantar pesos

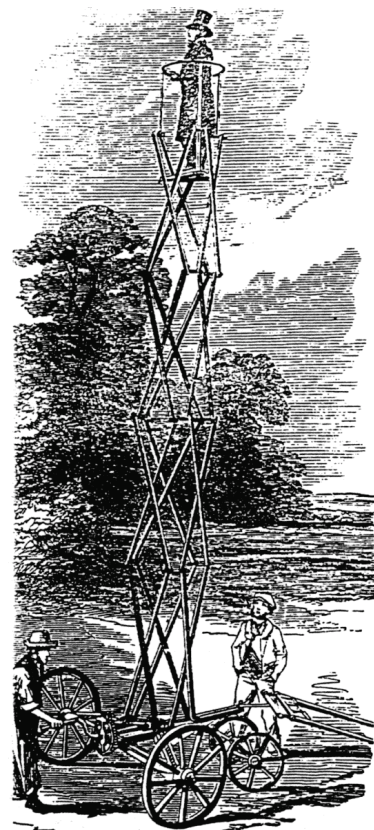


Pero también plantea nuevos conceptos estructurales, como la sombrilla de gran tamaño que pliega por medio de un eje roscado.



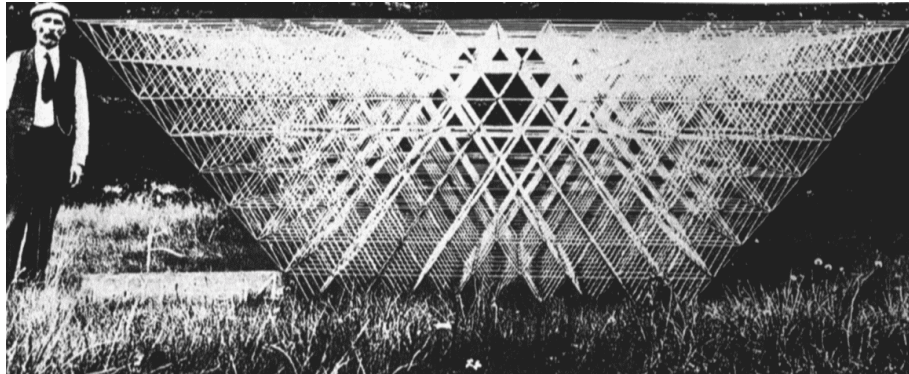
Los s.XVI y XVII son fructíferos en cuanto al estudio de mecanismos móviles, con nombres destacados como Turriano, Carduchi, Mariotte o Hooke.

En un dibujo de principios del siglo pasado se puede observar un artefacto desplegable empleado para levantar pesos, que nos recuerda los artilugios medievales, solo que con una fisonomía ya más próxima a las mallas desplegadas de aspas.



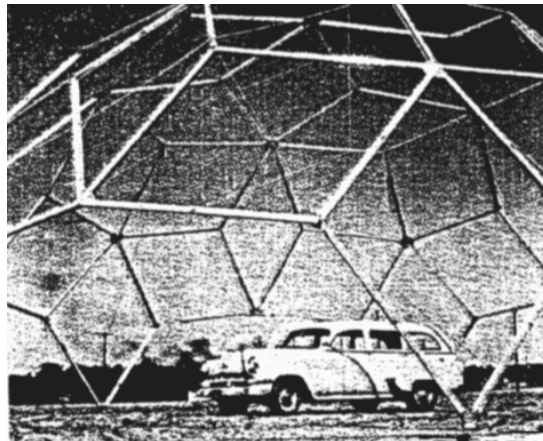
## ESTRUCTURAS MODERNAS

Como pionero en el estudio de las mallas espaciales tenemos a Graham Bell (1847-1922) conocido por ser el inventor del teléfono. En la siguiente imagen lo vemos al lado de un modelo de malla espacial, que es un tipo de estructura a la que le dedicó bastantes estudios.

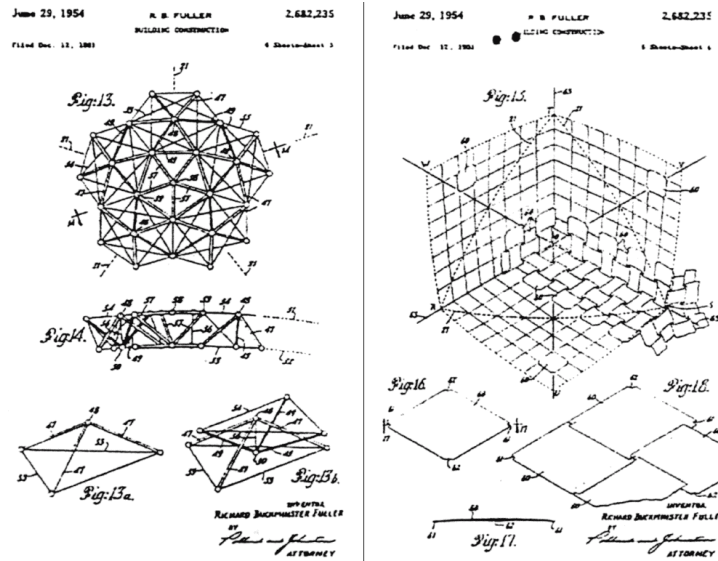


Konrad Wachsmann sistematiza las geometrías espaciales ultraligeras en 1940, y desarrolla un conector universal que permite a cada nudo recibir hasta 20 barras tubulares. Menherinhausen patenta en 1945 el nudo de bola y que abre el camino para el conocido sistema MERO de mallas espaciales.

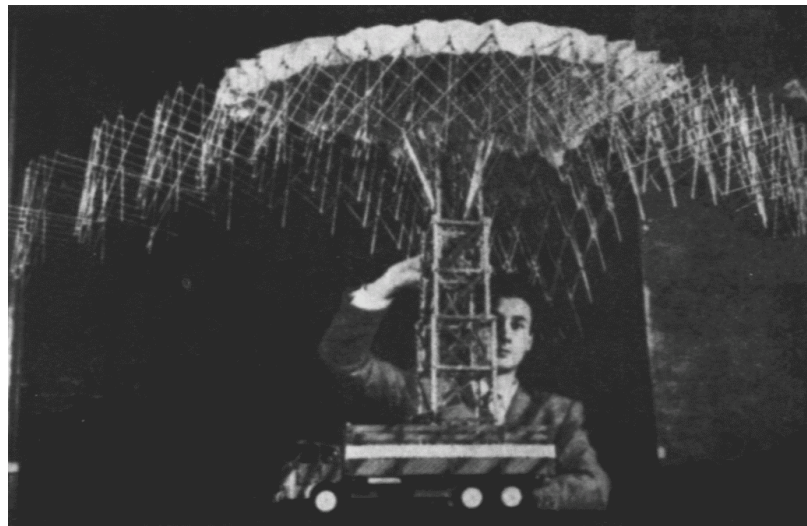
Durante las décadas de 1950 y 1960 arquitectos e ingenieros exploran la estética y modularidad de las mallas espaciales, pero es Buckminster Fuller (1895-1981) quien puede considerarse el padre de la moderna plegabilidad. En 1945 empieza sus investigaciones sobre arquitectura de emergencia. En 1953 realiza algunas cúpulas geodésicas desplegadas, como la que vemos en la fotografía siguiente.



En 1954 patenta el sistema geodésico en varias alternativas: mallas rígidas, plegable y chapas machihembradas.

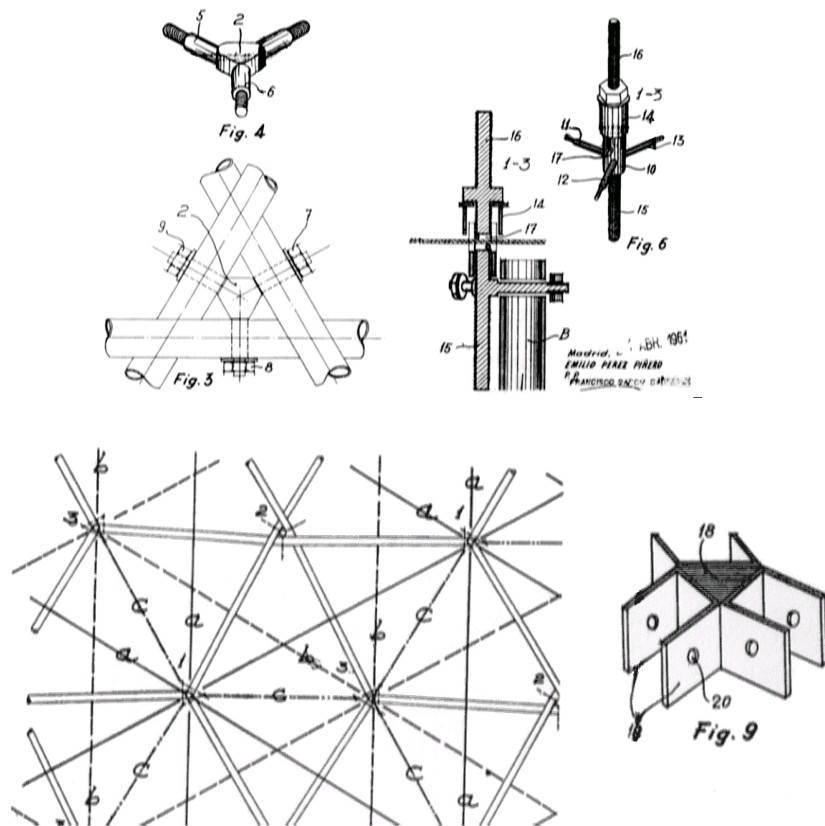


Emilio Pérez Piñero (1936-1972) inició su interés por las estructuras desplegables en el año 1961 con un proyecto con el que consiguió un premio de la Unión Internacional de Arquitectos. Se trataba de una cúpula desplegable, de 32 m de luz, 11 de flecha y 3.000 Kg. de peso que serviría de teatro ambulante para 500 personas. Su malla es sustancialmente diferente a la de Fuller en cuanto que esta es de doble capa, de espesor constante, y aquella de una sola.

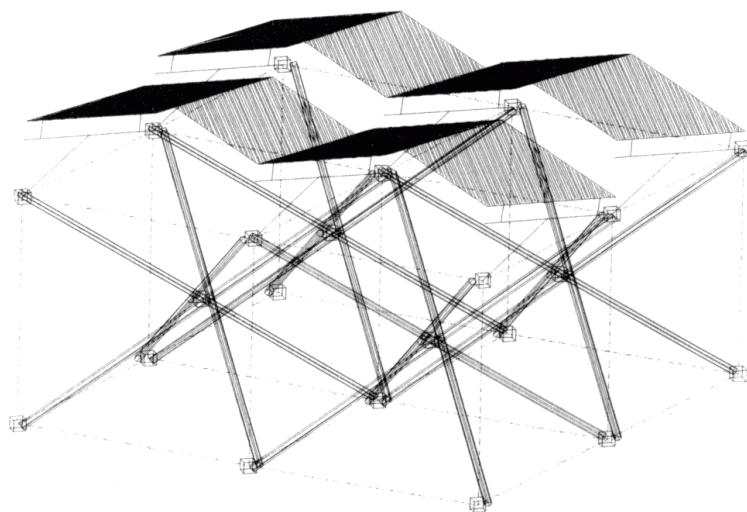


Como vemos la idea de esta malla es que fuese fácilmente transportable y desplegable desde un único camión.

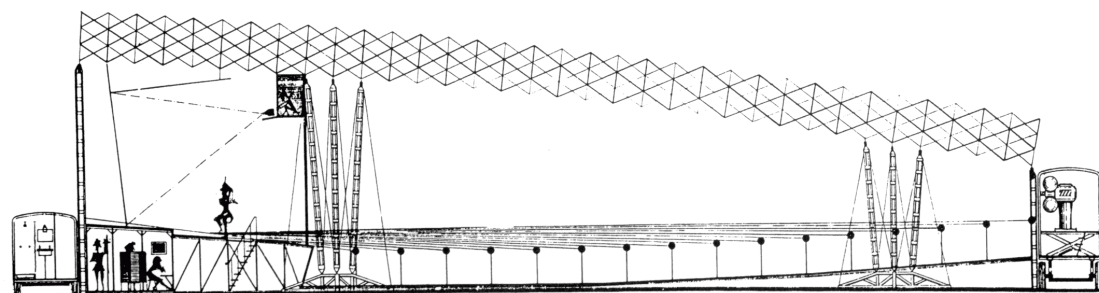
Estos estudios quedan resumidos en varias patentes, de las que a continuación vemos algunos detalles de la nº 266801 del año 1961. [5]



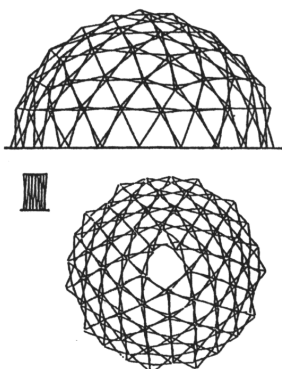
Con Dalí plantea el plegado de una superficie rígida apoyada sobre la malla para crear una vidriera. De aquí nace la posibilidad de efectuar el plegado empleando incluso paneles rígidos como material de cubrición.



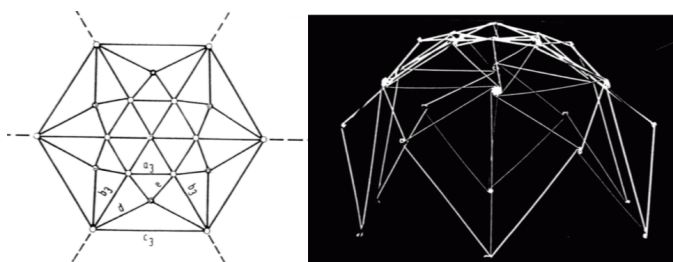
En 1971 diseña una cubierta desplegable para un teatro o cine ambulante, pero esta vez de planta rectangular, de 34x22 m. [4]



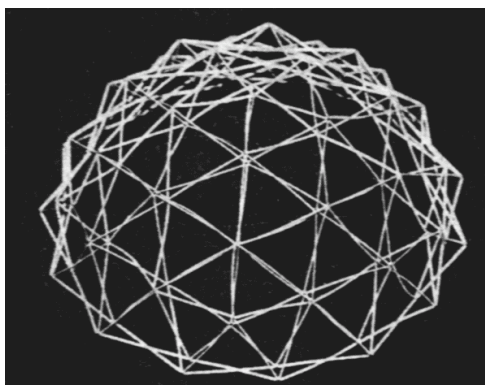
Los desarrollos posteriores se deben a los trabajos de Ziegler (1976), del que vemos un ejemplo de una malla desplegable generada en base a una cúpula geodésica.



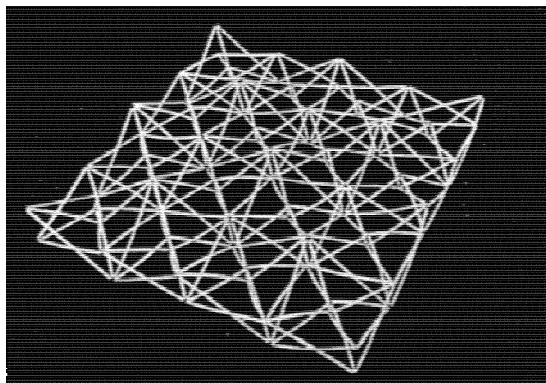
Santiago Calatrava (1980) realiza una una interesante propuesta con articulaciones desenganchables,



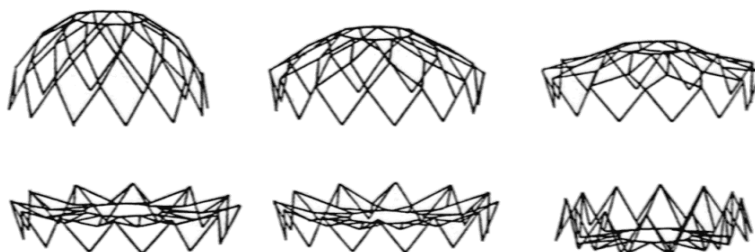
Charis Gantes empieza sus estudios sobre estas estructuras en 1989 y en 1991 diseña modelos para el programa ADINA que simulen el comportamiento de la estructura real, y estudia el despliegue de mallas a través del método de Newton-Raphson [7].



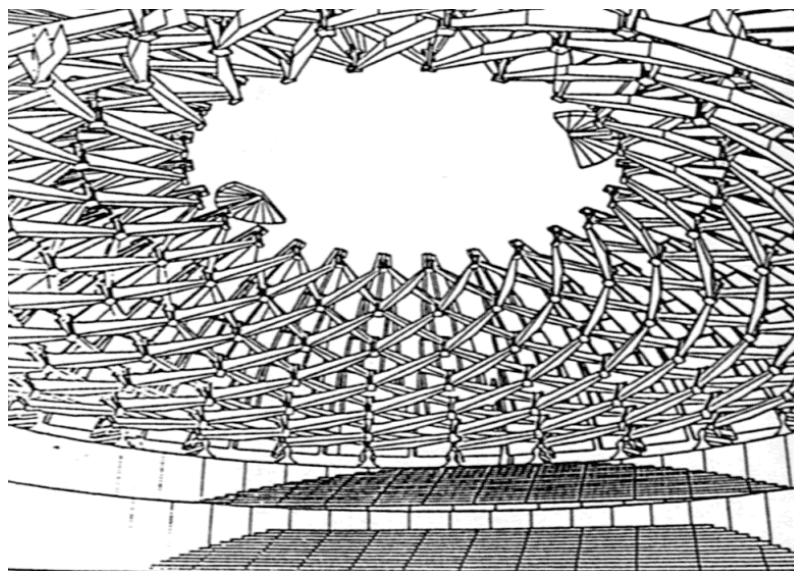
W. Shan presenta estudios sobre el cálculo matricial de estas estructuras en 1992 [8], y presenta una formulación formex de algunas estructuras muy elementales.



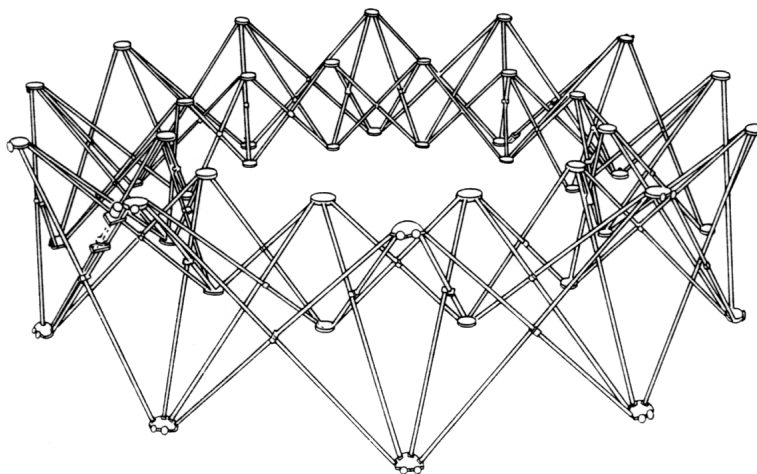
Kawaguchi realiza en 1993 estudios de mallas de una sola capa que define como colapsables.



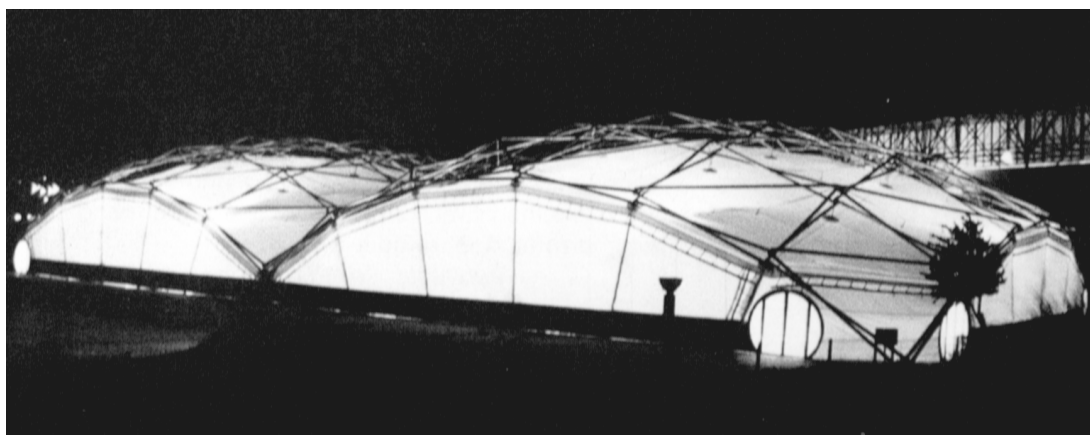
y Hobbeman realiza la propuesta de una cubierta con anillo desplegable desde su perímetro



Basándose en estos estudios, S. Pellegrino [9] también propone estructuras en anillo con ejes de giro no centrados en las barras que soluciona con un diseño excéntrico de las articulaciones.



Félix Escrig, J. Pérez Valcarcel y José Sánchez continúan realizando estudios sobre este tema, esto se concreta en la cubrición de una piscina en Sevilla, de 30x60 m. sobre el año 1994.



A través de este recorrido histórico hemos podido ver la evolución de los conceptos de trabajo en que se basan las mallas espaciales, y nos podemos hacer una idea de cual es el estado actual de desarrollo de este tipo de estructuras. En los capítulos siguientes iremos profundizando más en el estudio de estas mallas analizándolas desde otros puntos de vista.



## REFERENCIAS:

- [1] F. Escrig  
"General survey of deployability in architecture". Mobile and Rapidly Assembled Structures II. Computational Mechanics Publications. 1996
- [2] José Sánchez Sánchez  
"Estructuras desplegables de aspas para mallas poliédricas curvas". Tesis doctoral. Universidad de Sevilla. 1996
- [3] Horst Berger.  
"Light Structures Structures of Light. The art and engineering o tensile architecture". Birkhäuser Verlag. Basel - Boston - Berlín. 1996
- [4] John Chilton.  
"Space Grids Structures". Architectural Press. 2000
- [5] Felix Candela, E. Pérez Piñero, S. Calatrava, F. Escrig, J. Pérez Valcarcel.  
"Arquitectura Transformable". Textos de arquitectura. E. T. S. A. de Sevilla. 1993
- [6] J. Pérez Valcarcel.  
"Cúpulas desplegables de grandes luces con módulos de aspas". I Encuentro internacional de estructuras ligeras para grandes luces.
- [7] Gantes, C.; Connor, J.J. & Logcher, R.D.  
"Combining Numerical Analysis and Engineering Judgement to Design Deployable Structures" Computer & Structures. 1991
- [8] Shan, W.  
"Computer analysis of foldable Structures"  
Computer and Structures Vol. 42 nº6. 1992 pp.903-912
- [9] Pellegrino, S. & You, Z.  
"Foldable ring structures"  
Space Swctures. Ed. Parke, Thomas Telford. London. 1993

## BIBLIOGRAFÍA:

- Pérez Piñero, E.  
"Teatro ambulante" Arquitectura. Madrid nº30 1961
- Pérez Piñero, E.  
"Estructuras Reticulares Tridimensionales". Hogar y Arquitectura nº 40. Madrid. 1962
- Pérez Piñero, E.  
"Estructures Reticulées". L'Architecture d'Aujourd'hui. Vol. 141 1968
- Pérez Piñero, E.  
"Teatros Desmontables". Informes de la Construcción nº 231. 1971
- Puertas del Río, L.  
"Estructuras espaciales desmontables y desplegables". Informes de la Construcción. IET nº 409. Madrid. 1990

---

## 1.2 GEOMETRÍA DE LAS MALLAS DESPLEGABLES

---

---

### 1.2.1 DISEÑO Y CONSTRUCCIÓN

---

Como primer acercamiento al proceso de diseño y podemos reproducir una disertación del propio Emilio Pérez Piñero que desarrolla este tema [1], y nos va presentando las distintas fases y factores influyentes en el mismo:

*Supuesto fijo un sistema de fuerzas exteriores actuando sobre un cuerpo, al variar la forma de éste varían sus tensiones internas y trayectorias de éstas. Si hemos de evitar su rotura, y dado que los materiales distintos tienen cualidades resistentes muy diferentes, resulta premisa inicial la necesidad de adaptar y hermanar en lazo insoluble la forma y el material.*

*En el caso de que el sistema de fuerzas externas sea la gravedad, la intensidad del campo de fuerzas es proporcional a la densidad del material, y si los cuerpos sometidos a sus efectos no son otros que las construcciones, reducimos el problema a determinar dos variables: la forma y el material. La forma ha de estudiarse para conseguir la más adecuada distribución de tensiones. En el material serán sus cualidades mecánicas y su densidad. en concordancia con la forma elegida y las tensiones que en ella se produzcan, lo que nos fijará su elección.*

*La estructura en la forma es lo que hace posible su subsistencia como tal. En el concepto primero estructura y forma se identifican. Recubrimiento, aislamiento, decoración, son conceptos posteriores. Son el "acondicionamiento" para desarrollar una función vital.*

*La caverna, la roca salediza, son la estructura primera natural que el hombre emplea. En ellas recubre, aísla y decora su vivienda.*

*Interesante resulta señalar que son sus necesidades espirituales las que hacen a aquellos gigantes de la primera época crear las primeras estructuras producto del*

*propio ingenio del hombre. Con el dolmen y su derivación inmediata, el sepulcro de corredor, aparece la estructura adintelada, tan primitiva como actual.*

*Con los materiales pétreos el hombre construye sus tumbas y templos; con ellos desarrolla, sucesiva y escalonadamente, aquellas formas que, dando cabida a sus necesidades espirituales, determinan esfuerzos que la piedra es capaz de resistir. Así, tras de los dinteles, aparecen el arco, la bóveda y la cúpula; y pronto sienta las bases sobre las que ha de moverse miles de años después la arquitectura pétreo.*

*La choza y el palafito, más de acuerdo con su propia provisionalidad sobre la tierra, marcan el comienzo de la arquitectura carpinteril, desarrollando a su vez las formas más adecuadas a ésta. Es en el cruce e interferencia del material y la forma, con la inadaptación en su empleo, cuando aparece lo monstruoso, que si en alguna ocasión llega a prevalecer es sólo para mostrarnos, como en la cubierta del templo de Carnak, el elevado tributo que en ello se ha de pagar.*

*En el palacio de Stesifonte, con sus casi 26 metros de luz en su vestíbulo abovedado; en la cúpula del Panteón de Roma, o en la basílica de Constantinopla, estamos en el límite de lo que es posible hacer con los materiales empleados.*

*La cúpula de Santa Sofía, construida con ánforas, muestra cómo sus constructores, conscientes de que el mantenimiento de la forma determina el mantenimiento de la distribución de tensiones, y necesitando disminuir la intensidad de las tensiones máximas, debido a que las ventanas de la base de la cúpula reducían enormemente la sección de trabajo en su arranque, no encuentran otra solución que disminuir el sistema de fuerzas que sobre ella origina la gravedad, lo cual sólo pueden lograr disminuyendo la densidad del material a emplear.*

*El acero y las aleaciones ligeras, con su enorme resistencia a todo tipo de esfuerzos, marcan la liberación de la forma.*

*Su enorme resistencia unitaria hace de estos metales el material idóneo para el trabajo a tracción.*

*El aprovechamiento de su capacidad de resistencia a compresión, igualmente grande, se ve limitado por el efecto de pandeo. Secciones más que suficientes respecto de su resistencia unitaria resultan insuficientes, ya que conducen a formas excesivamente esbeltas. Resulta imprescindible modificar la forma, aumentando los momentos de inercia de las secciones para evitar este efecto. Pero este aumento ocasiona un gran incremento del costo y del peso muerto.*

*Como solución y síntesis de este problema surgen las estructuras reticulares. Con ellas se modifica la forma y, por tanto, las tensiones y trayectorias internas, sin aumentar las secciones efectivas de trabajo.*

*Las barras periféricas nos determinan la forma ideal dentro de la cual hemos de considerar las trayectorias efectivas de tensión, que se concentran y canalizan sobre las barras de triangulación interna.*

*Las estructuras reticulares se han clasificado impropriamente en planas y tridimensionales; realmente todas son tridimensionales, concebidas como realmente son en su conjunto. Los clásicos cuchillos de armadura o las vigas planas son realmente parte integrante de una estructura tridimensional. El considerar y*

*descomponer la estructura en elementos planos facilita el cálculo y la construcción. Caso similar lo constituyen las estructuras tridimensionales, integradas por vigas de sección triangular que son de fácil cálculo, y una vez unidas aparecen como un entramado tridimensional continuo. La clasificación de estas estructuras en planas y tridimensionales es puramente conceptual y subjetiva.*

*La concepción y realización de una estructura reticular es el resultado de una actitud mental ante la naturaleza, que se descompone, analiza y condensa nuevamente en un típico producto manufacturado por la mente del hombre.*

*En su génesis pueden concebirse y desarrollarse de dos formas:*

*a) Como canalización de las tensiones internas originadas por el sistema de fuerzas exteriores actuantes en la forma previamente determinada.*

*b) Como conducción de los esfuerzos exteriores hasta los apoyos.*

*En el primer caso la retícula aparece como esqueleto de la forma general, que se supone determinada y fijada previamente. La barra es una condensación de masa en la línea de fuerza y los huecos son una supresión de masa inerte.*

*En el segundo caso la barra tiene personalidad propia, la forma inicial no existe. La retícula se construirá por equilibrio sucesivo de puntos en el espacio, siguiendo líneas funiculares de las fuerzas a equilibrar. La forma surge al final, determinada por la estructura.*

*Es el primer método el más adecuado en la mayoría de los casos, pues se parte de formas adecuadas a la función que hemos de desempeñar. La retícula interior deberá adaptarse a las direcciones de las trayectorias de tensión producidas por el sistema exterior en la forma elegida.*

*Aunque de esta forma los esfuerzos en las barras son mínimos, es posible (y necesario casi siempre) apartarse con las barras de las trayectorias teóricas de tensión. Y aquí radica el verdadero milagro de las estructuras reticulares, pues con ellas logramos no sólo reducir la masa general y separar exactamente los esfuerzos de tracción, compresión y cortadura, sino también conducir estos esfuerzos por cauces prefijados.*

*Las estructuras reticulares pueden ya considerarse como formas rígidas autónomas, con las cuales podrá operarse independientemente de su génesis conceptual.*

*El segundo método, aparentemente más exacto, choca inmediatamente con el primer inconveniente, que es la variedad y variabilidad de los esfuerzos externos que actúan sobre una estructura. Baste señalar la actuación del viento, que en muchos casos llega a tener una importancia predominante.*

*La fijación de la forma general previamente, que sin duda es el método más adecuado, requiere un concienzudo análisis de fuerzas actuantes, recintos y volúmenes requeridos, pues "fijación previa de la forma" en modo alguno significa que dicha fijación sea arbitraria.*

*En la concepción, proyecto y ejecución de una estructura reticular aparecen las siguientes **fases**:*

- 1.- Determinación de la forma general del conjunto.*
- 2.- Determinación de la retícula, disposición y longitud de las barras. Esto puede llamarse "cálculo geométrico" de la estructura.*
- 3.- Cálculo y dimensionamiento de las barras.*
- 4.- Resolución constructiva de la conexión de las distintas barras.*
- 5.- Formación efectiva de la estructura en su emplazamiento con el montaje de sus elementos.*

*Cada una de estas cinco fases puede tomar una importancia preponderante, según los casos, quedando las restantes aparentemente en un plano secundario; pero siempre es preciso cuidar muy concienzudamente cada una de ellas.*

***La determinación de la forma general** de la estructura es preciso realizarla con sumo cuidado. La configuración de la planta y sus dimensiones, el volumen interior que se precisa, el aspecto externo y, fundamentalmente, los esfuerzos a que haya de estar sometida son los factores con los que habrá de intuirse la forma, ponderando cuidadosamente ventajas e inconvenientes hasta centrarse en el límite común de todas las exigencias y condicionamientos.*

***La determinación de la retícula interna**, o "micro-forma", deberá estar inspirada, como ya hemos dicho, en las trayectorias de tensión que en la forma general, o "macro-forma" produzca el sistema de cargas exteriores. Sin perder nunca de vista esta distribución de trayectorias de tensiones internas, habrá que buscar la mayor continuidad y armonía en la distribución de las barras, tratando a su vez de lograr que el mayor número posible de ellas sean iguales entre sí, a fin de simplificar su ejecución. El efecto estético del conjunto será siempre muy importante, y a él hay que atender con mucho cuidado, no sólo porque estas estructuras suelen tomarse como elemento decorativo de primer orden, sino porque las estructuras mecánicamente mejor resueltas suelen ser también las más bellas.*

***El cálculo y dimensionamiento de las barras**, salvo en determinados casos simplificable (de las cuales son elementales las denominadas planas), es de graves inconvenientes teóricos, haciéndose preciso el ensayo en modelos a escala.*

***La resolución constructiva de la conexión** de las distintas barras es problema de tal importancia que por sí sólo distingue la totalidad de los sistemas existentes, que suelen ser objeto de patente por sus creadores.*

***Por último, el sistema de montaje** tiene una importancia económico-práctica decisiva, y está íntimamente ligado con el anterior. Él es, en definitiva, el que hace*

*malo o bueno un sistema. El que no se precise andamiaje y que la parte prefabricada en taller sea lo mayor posible es fundamental.*

*En el caso de cubiertas para grandes luces, es preciso condicionar todo el proyecto a la forma de la estructura. Dejando al margen las formas colgantes y alabeadas como típicas de otros procedimientos (aunque su solución reticular, sobre todo en combinación con otras formas, es también del mayor interés), es la cúpula la forma y solución reina de este problema.*

*Con las cúpulas reticulares se logran espacios internos extraordinariamente limpios y diáfanos; no existen problemas de anclajes ni de empujes; son formas absolutamente autónomas, y su instalación, que no precisa de andamiaje, puede llegar a ser fundamentalmente rápida.*

*En la realización de una cúpula reticular cobran especialísimo relieve y dificultad cada una de las cinco fases que hemos señalado como fundamentales en el proyecto y ejecución de una estructura reticular.*

*Referente a la **forma general**, son datos característicos:*

*a) Superficie geométrica a que pertenece la cúpula.*

*b) Manera de apoyarla. Ya sea en puntos aislados o a lo largo de un paralelo. Los esfuerzos son totalmente distintos al variar los apoyos.*

*c) Relación entre la fecha y el diámetro de la cúpula. De ello depende la distribución de tensiones, sobre todo por influencia del viento. Esta relación debe cuidarse muy bien en cúpulas de gran tamaño.*

*d) Espesor de la cúpula y conveniencia de que sea de dos o una sola capa de barras. Esto depende del tamaño de la cúpula, de su curvatura, de la "frecuencia" o relación entre la longitud de la barra y el radio de curvatura y del sistema de unión de las barras en los nudos.*

*Entre las cúpulas de dos capas tienen especial interés las que incorporan las planchas de cubrición como elemento resistente, constituyendo con ellas una de las capas resistentes.*

*Particular dificultad cobra en las cúpulas la determinación de la **retícula y la longitud de las barras**. Primeramente fijaremos la frecuencia o tamaño relativo de la retícula; seguidamente, tomando como superficie la geométrica directriz de la cúpula, realizaremos la triangulación que servirá de base a dicha retícula.*

*La triangulación de la superficie directriz puede realizarse por varios procedimientos; en cada uno de ellos se atiende a mantener la mayor uniformidad posible entre las longitudes obtenidas. condicionándose en cada caso a que las que tienen una determinada relación sean iguales entre sí. Cuanto menor número de dimensiones distintas existan tanto más económica será la realización de la cúpula.*

*Cuando se trata de cúpulas esféricas y la relación flecha/diámetro se acerca o sobrepasa el valor de 1/3, puede recurrirse a una triangulación poliédrica de la esfera. Elegido un poliedro esférico regular, triangularemos una de sus caras, y ésta se repite*

en las demás. Este sistema permite llegar, como es lógico, a la triangulación total de la esfera sin aumentar el número de barras distintas. Estéticamente y mecánicamente deja bastante que desear, pues en los vértices del poliedro se pierden triángulos, apareciendo claramente la falta de continuidad de unas a otras caras.

Una vez fijada y calculada la triangulación básica de la cúpula, ha de pasarse a la determinación de las longitudes reales que forman la retícula.

Tras este "cálculo geométrico" es preciso realizar el **cálculo mecánico**, determinando las secciones y uniones de las barras. De la complejidad de este problema baste señalar que una cúpula de doble capa, de frecuencia no muy elevada, tiene más de tres mil barras. Los ensayos en modelos a escala resultan imprescindibles.

La resolución adecuada de la **conexión de barras** es, en definitiva, lo que hará económicamente posible la construcción de la cúpula. Fundamentalmente hay tres maneras de "hacer la cúpula":

I) Diseñando un nudo en el que puedan conectarse las barras con ángulos variables. Así, se fabricarán los nudos, se fabricarán las barras con sus longitudes y conexiones, realizándose el montaje *in situ*. Esta modalidad de barras y nudos independientes permite tres variantes:

- a) Que las barras vayan de nudo a nudo, en cuyo caso en él hay que conectar los extremos de las barras que allí concurren. Sistema seguido por Füller.
- b) Que una de las barras atraviese el nudo enteriza, en cuyo caso el nudo se forma sobre dicha barra.
- c) Que varias o todas las barras atraviesen enterizas el nudo, en cuyo caso éste se reduce a una pieza que las aprisiona superpuestas.

II) Considerando la cúpula subdividida en piezas integradas por conjuntos de barras previamente unidos. Estos conjuntos deberán conectarse entre sí. lo cual implica un nuevo concepto de nudo. En el proyecto de estos conjuntos es preciso tener muy en cuenta el transporte a que hayan de someterse.

III) Soluciones desplegadas. La cúpula podrá prefabricarse en su totalidad o subdividida en fragmentos de gran tamaño. Posteriormente se transporta y despliega, realizándose la instalación muy rápidamente.

Una vez construida y transportada a su lugar de emplazamiento, es preciso **instalar la cúpula**. Según sea el sistema que se haya empleado para su construcción, sus dimensiones y su peralte, el sistema adecuado para su instalación será distinto.

I) El montaje por anillos sucesivos. partiendo de la base. que es el más generalizado, puede adoptar las siguientes modalidades:

- a) Si la cúpula es de barras y nudos totalmente independientes, resultará conveniente montar previamente fragmentos capaces de automantenerse al ir formando con ellos la cúpula. Como elementos de elevación se precisarán grúas de brazo giratorio. Para los obreros la propia estructura es el mejor andamio,

completándose con torres desplazables o cestas colgantes si las dimensiones de la cúpula lo requieren.

b) Cuando las barras atraviesan enterizas los nudos, no suele resultar factible el previo montaje de fragmentos, pues las barras van tejiéndose sucesivamente. Entonces resulta preciso apuntalar determinados puntos durante el montaje. Puede esto realizarse con puntales telescópicos o auténticas torres si las dimensiones son muy grandes.

c) Cuando la estructura está constituida por fragmentos, éstos deben precisamente diseñarse para facilitar el montaje y hacer innecesario el uso de cualquier tipo de andamiaje o apuntalamiento. Es preciso entonces elegir el tipo de grúa más conveniente y que haga posible la totalidad de la instalación.

Cuando la relación fecha/diámetro es inferior a  $1/3$ , resulta siempre preciso apuntalar las primeras vueltas de la cúpula. hasta llegar a formar un anillo lo suficientemente rígido para autosostenerse. Inconveniente de todos los montajes por vueltas, partiendo de la base, es la posibilidad de encontrar dificultades al ajustar las últimas vueltas. No hemos de perder nunca de vista que una cúpula reticular se calcula partiendo de una superficie geométrica perfecta. Las dimensiones de sus barras y las conexiones en los nudos han de calcularse y construirse con aproximación hasta la décima de milímetro. La nivelación y replanteo de la base de apoyo debe mantener, asimismo, gran exactitud. Pequeños errores en estas operaciones ocasionan una deformación inicial cuando la cúpula sólo tiene unas vueltas, ya que en este momento el conjunto es deformable. Conforme se va cerrando, va llegándose a una forma rígida, y si ésta no corresponde a la forma ideal calculada exactamente. por vicio inicial en el apoyo, llegará un momento que las sucesivas barras o piezas no encajarán debidamente. Para seguir el montaje es necesario corregir esta deformación general. Esto puede lograrse empleando elementos de tracción y levantamiento, que distribuidos estratégicamente en la cúpula, nos vayan haciendo posible la sucesiva colocación de las piezas restantes. Así ellas mismas van obligando a la cúpula a adoptar su verdadera forma. Esta operación provoca, al mismo tiempo, corrimientos en la base que corrigen los iniciales errores. En cúpulas de grandes dimensiones el intento de realizar esta operación puede producir roturas locales en los puntos en que se aplican las sollicitaciones. Si la corrección resulta imposible puede hacerse necesario, incluso, desmontar la cúpula, comenzando de nuevo. El llegar a modificar las piezas de terminación, adaptándolas a la nueva forma adoptada por la cúpula, sería verdaderamente una chapuza inaceptable. Los inconvenientes que por estas causas llegan a producirse pueden ser tan graves que, en ocasiones, aconsejan invertir el procedimiento de montaje.

II) Montaje por anillos sucesivos partiendo de la clave. Consiste simplemente en invertir el procedimiento anterior. Se arranca del suelo colocando vueltas sucesivas alrededor de la clave. A cada vuelta es preciso elevar el conjunto ya montado. Disponiendo de elementos de elevación adecuados y suficientes el montaje resulta de gran sencillez y vistosidad; las piezas sucesivas de la periferia van superponiéndose y ajustando perfectamente. Al final, la estructura, como una gran tapadera rígida, nos marca la posición exacta de los apoyos.





(1957 Honolulu. 20 horas de montaje. luz 145 pies, alto 50 pies. con paneles de cubrición resistentes de aluminio.)

*III) Montaje por gajos. Ese sistema es adecuado cuando la cúpula puede descomponerse en fragmentos rígidos de este tipo, lo cual ocurre casi exclusivamente cuando el diseño y el cálculo se han hecho atendiendo precisamente a esta subdivisión. Una vez dispuestos radialmente los distintos gajos, se procede a su erección y conexión mediante un elevador central.*



(Palacio de deportes de Palafolls )

**IV) Montaje en las estructuras desplegadas.** Poder prefabricar totalmente o subdividida en grandes trozos una cúpula es característica fundamental de estas estructuras. Cúpulas de más de 30 metros de diámetro y relación flecha/diámetro igual a 1/3, es posible, prefabricadas totalmente, transportadas en un camión o un helicóptero. Su despliegue e instalación puede realizarse desde el propio camión si éste dispone de una plataforma o torre capaz de elevar la clave hasta su posición definitiva. La "rigidización" de la estructura, una vez desplegada, ya sea manual o automática, es muy rápida. Particularmente simple es el despliegue desde un helicóptero cuando éste es el medio de transporte. La estructura plegada y suspendida de seis nudos inferiores se despliega automáticamente al soltar el cinturón que la mantiene plegada; si la rigidización es automática, queda inmediatamente instalada.

Independientemente del transporte, puede realizarse el despliegue e instalación en el suelo, valiéndose de ruedas y elementos auxiliares de elevación. Estas estructuras pueden dotarse de dispositivos automáticos de despliegue, creciendo como un auténtico ser vivo. Su fijación, igualmente automática, determina la total instalación de la estructura con sus propios elementos. Una cúpula de este tipo, lanzada y depositada sobre la superficie de la luna, se autoinstalaría sin intervención de la mano del hombre. Cuando la cúpula se compone de vanos fragmentos desplegados, se procede al montaje de cada uno de estos fragmentos, procediéndose después a su conexión como si de estructuras desmontables se tratara. En este caso se ha de proyectar la subdivisión de la cúpula precisamente atendiendo a facilitar posteriormente el montaje de los fragmentos desplegados.

**Las estructuras desplegadas,** con cuyo estudio inicié mi interés por las estructuras reticulares, en el año 1961. precisan consideración aparte no sólo por esta cualidad que permite prefabricación integral, sino además por su actuación como armadura resistente al margen de su montaje.

Fundamentalmente están constituidas por un conjunto articulado de barras rígidas, totalmente deformable; capaz de plegarse hasta formar un apretado haz con todas ellas y susceptible de extenderse adoptando la forma plana o curva deseada. Este conjunto no pasa a ser estructura rígida, capaz de aguantar todo tipo de esfuerzos, hasta que no se conectan para la fijación. Esto siempre es así, cualquiera que sea el sistema de fuerzas externas que sobre ella actúa. Son estructuras "estrictas" cuyo peso es mínimo.

Su forma de trabajo se asemeja al cuerpo de un vertebrado. Las barras a compresión forman un esqueleto, una auténtica columna vertebral. Las barras de rigidización actúan como el sistema muscular que envuelven y mantiene el esqueleto.

Cualquier carga exterior, al tiempo que las hace variar ligeramente de forma adaptándose a la más conveniente posición de trabajo, origina el que determinadas barras a tracción actúen inmediatamente. Ocurre exactamente lo que al cargar el lomo de un gato.

Pero cuando esta denominación, de "estructura viva", queda plenamente justificada y adquiere todo su valor es cuando las barras desenganchables de rigidización son dotadas de dispositivos adecuados que realizan, desde dentro de la propia estructura, los esfuerzos precisos para el despliegue y bloqueo, una vez enganchadas, de todas las barras.

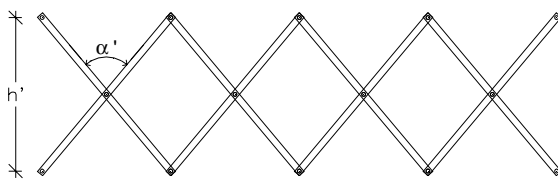
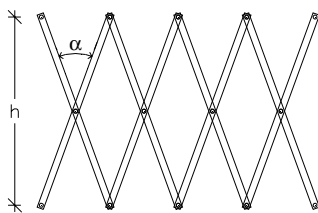
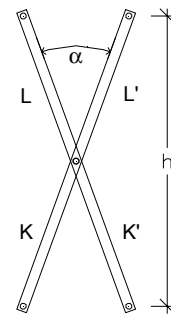
*El paquete inicial, una vez liberado de sus amarras, se expande, crece y se levanta, produciéndose una increíble metamorfosis a expensas de sus propias tensiones internas.*

## 1.2.2 GEOMETRÍA DEL DESPLIEGUE

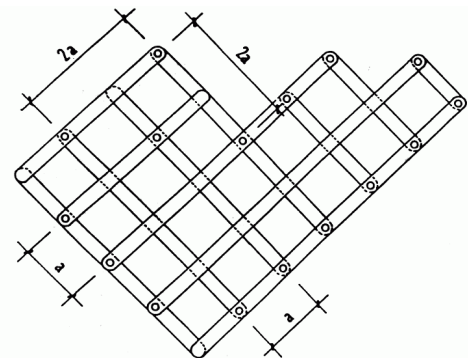
Las mallas que estudiaremos se basan principalmente en un elemento de dos barras que se cruzan en un eje intermedio recordando unas tijeras, por lo que C. Gantes llama a este elemento base: SLE (Scissor-Like-Element).

Si las barras son de igual longitud y están unidas por sus puntos medios, entonces los extremos de las mismas se hallarán, dos a dos, en la misma horizontal y vertical, definiendo un rectángulo ficticio de altura  $h$  y ancho variable en función del ángulo de apertura  $\alpha$ .

Uniendo varios de estos módulos en línea obtendríamos una especie de viga expansible:



Esto se podría hacer más complejo:



Este mecanismo tendrá una forma de movimiento única, o sea, será **consecutivo**, como lo define S. Calatrava [2] si sus grados de libertad se reducen a 1. Esto se puede materializar para estructuras planas con la expresión:

$$G = 3 \cdot (B - 1) - 2 \cdot N_1 - N_2$$

siendo:

G: grados de libertad resultantes.

B: número de barras.

$N_1$ : número de nudos con grado de libertad 1, contados como a continuación se detalla.

$N_2$ : número de nudos con grado de libertad 2.

Se debe tener en cuenta en el cálculo de  $N$ , consta del número de grados de libertad que tiene el tipo de nudo de que se trate, y que impondrá éste a las otras  $i$  barras que quedan en nudo tras descontar una que será fija de modo relativo. Tenemos por tanto  $i = \text{número de barras que convergen en el nudo} - 1 = \text{valencia del nudo}$ .

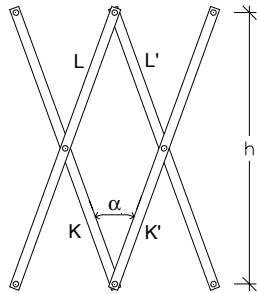
Así para cada conjuntos de nudos con el grado de libertad  $1$ , tendremos el valor de  $N_1$ :

$$N_1 = \sum_{i=1}^n i \cdot (\text{num. nudos con valencia } i)$$

Si como resultado final nos da  $G=1$  y las barras se hallan dispuestas correctamente. La estructura será plegable, y bastará añadir o fijar una barra que se convierta en una estructura isostática.

En cualquier caso nosotros nos ceñiremos a nuestro esquema en aspa, que nos garantiza la condición de despliegue, y nos permite centrar esfuerzos en otras direcciones.

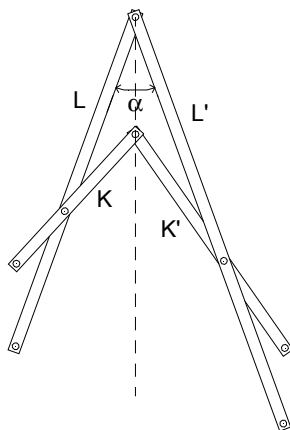
Las condiciones geométricas para que se produzca el plegado y desplegado se pueden obtener a partir del rombo, o romboide, que nos da el encuentro de dos aspas.



La condición de plegado se dará con el alejamiento máximo de los nudo superior e inferior y será:

$$L+K = L'+K'$$

Obsérvese que S. Calatrava [2] plantea otra posible condición de plegado basándose en el acercamiento de los nudo superior e inferior:



La condición de plegado sería en este caso:

$$L-K = L'-K'$$

Planteando ambas condiciones de plegado simultáneamente resultaría:

$$L = L' \quad \text{y} \quad K = K'$$

Lo que implica una simetría respecto de la vertical.

Para el tipo de malla que desarrollaremos y estudiaremos, esta segunda condición de plegado no presenta mayor interés, por lo que nos ceñiremos a la primera.

Respecto al desplegado, si quisiéramos que la malla desplegasen totalmente hasta quedar plana, la condición sería similar a la de plegado, pero con respecto a la horizontal:

$$L + L' = K + K'$$

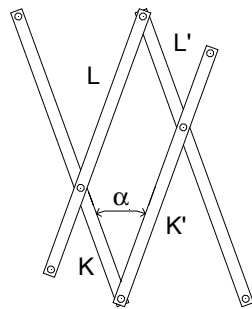
que combinando con la condición de plegado:

$$L + K = L' + K'$$

Resultaría:

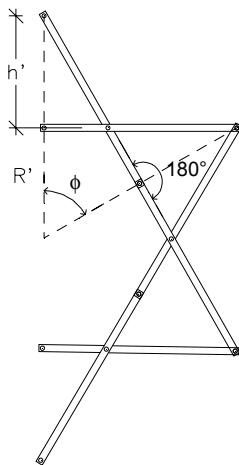
$$L = K' \quad \text{y} \quad L' = K$$

Lo que se nos traduce en un paralelogramo:

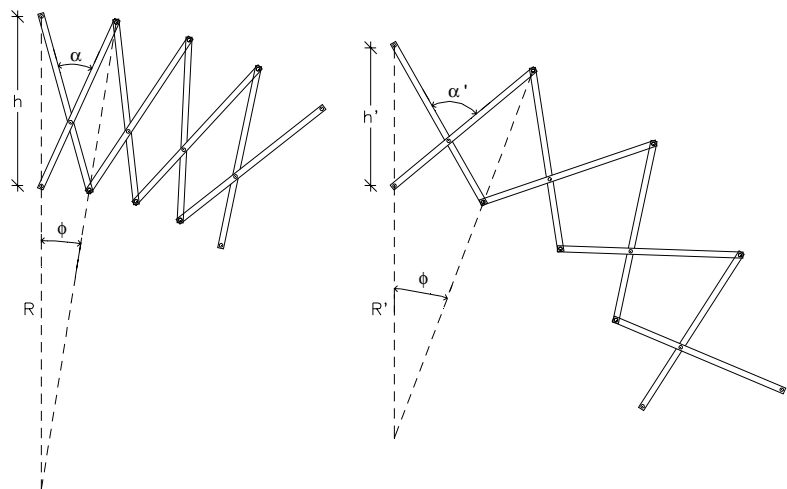


Por otra parte no siempre es preciso la condición de desplegado total. De hecho el arco visto anteriormente no se ajusta a esta regla del paralelogramo y sin embargo sirve bien para nuestros objetivos. En este caso ocurriría que los lados inferiores marcarían el límite de despliegue cuando llegasen a formar un ángulo de 180°, mientras que los superiores formarían un ángulo menor.

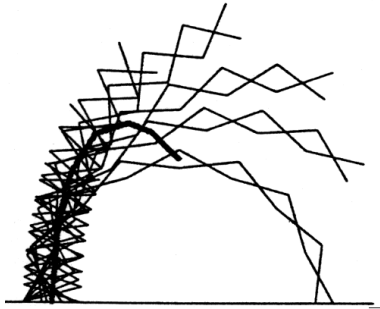
De hecho, en algunos casos se usa esta condición para fijar un límite de desplegado máximo:



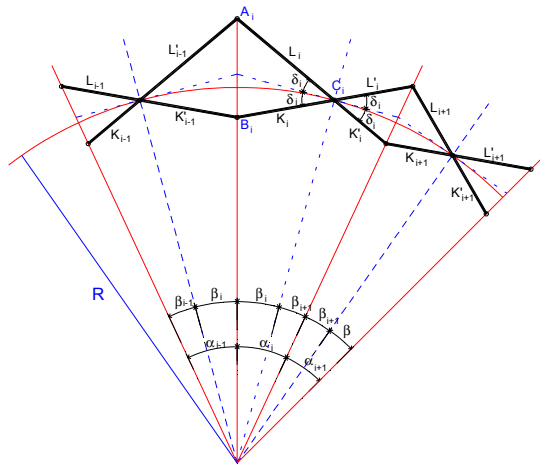
Este último esquema también nos vale como introducción a los despliegues según curvas. El esquema en aspa puede ser bastante versátil, de hecho, simplemente desplazando el punto de cruce de las barras y repitiendo el mismo módulo, obtendríamos figuras en forma de arco:



Se observa que a medida que aumenta el ángulo  $\alpha$ , el ángulo  $\phi$  también aumenta, a la vez que disminuye el radio de curvatura  $R$ . Por tanto la curvatura del conjunto se va pronunciando a medida que avanza el despliegue, como se constata en los estudios de C. H. Hernández [3].



Para el estudio del despliegue en la dirección curva deberemos hacer el estudio que a continuación describimos con las consideraciones que comentamos a continuación [4]:



- Los puntos de cruce de aspas **C**, se hallan sobre la superficie generatriz que queremos cubrir.
- Los nudos superiores e inferiores estarán contenidos en el mismo radio.
- Para que se pueda plegar debe cumplirse:

$$L_{i-1} + K_{i-1} = L_i + K_i$$

y si los puntos **C** deben hallarse sobre la circunferencia, está claro que

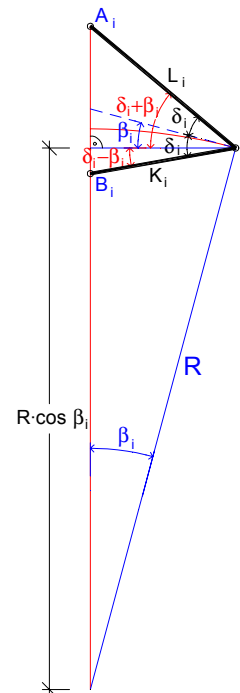
$$L_{i-1} = L_i \quad \text{y} \quad K_{i-1} = K_i$$

Lo que implica que estos cuatro lados deben tener simetría respecto del radio que pasa por  $A_i$  y  $B_i$ .

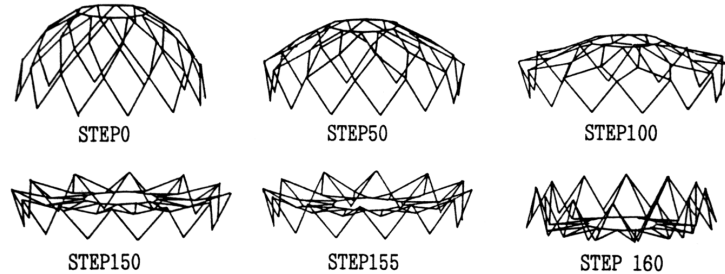
Por ello todos los ángulos  $\delta$  serán iguales, y podemos hallar las longitudes con las expresiones:

$$L_i = L'_{i-1} = \frac{R \cdot \text{sen}(\beta_i)}{\cos(\delta + \beta_i)}$$

$$K_i = K'_{i-1} = \frac{R \cdot \text{sen}(\beta_i)}{\cos(\delta - \beta_i)}$$



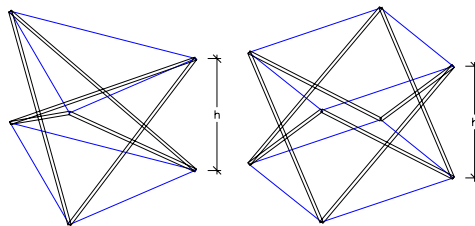
Evidentemente las mallas de aspas no es el único tipo plegable, de hecho Kawaguchi realiza un estudio de procesos de plegado de mallas con barras conectadas sólo en sus extremos [5], aunque como vemos las configuraciones que adoptan son más inestable y deberán confiarse más a la solidez de los nudos o idear algún sistema de estabilidad adicional para que sea operativo:



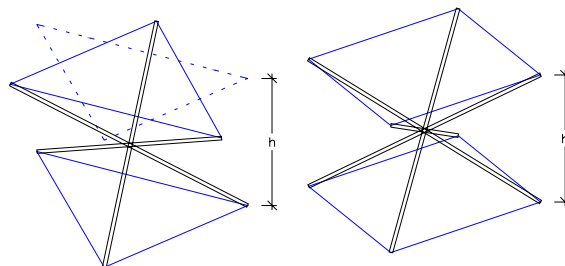
## 1.2.3 CLASIFICACIÓN TIPOLOGICA

Los esquemas anteriores se limitan a un estudio plano. Una configuración espacial se consigue básicamente combinando planos de aspas en el espacio, aunque como veremos, se abren nuevas posibilidades.

En primer lugar, si usamos la combinación de **aspas** en 3 o 4 planos que se intersectan obtendremos módulos de base triangular o cuadrada:

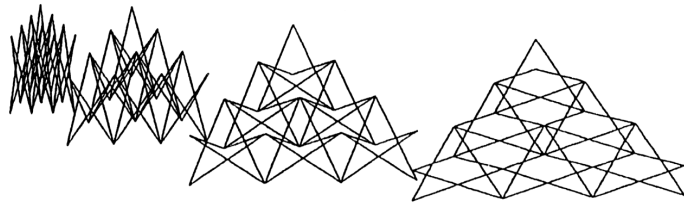


Pero en el espacio tenemos en principio otra alternativa que es el equivalente espacial del aspa: el **haz** de barras.

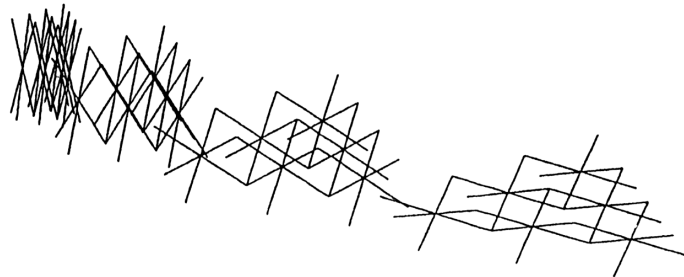


Como vemos, para el mismo módulo reducimos el número de barras a la mitad, pero a cambio va aumentando la complejidad del nudo intermedio.

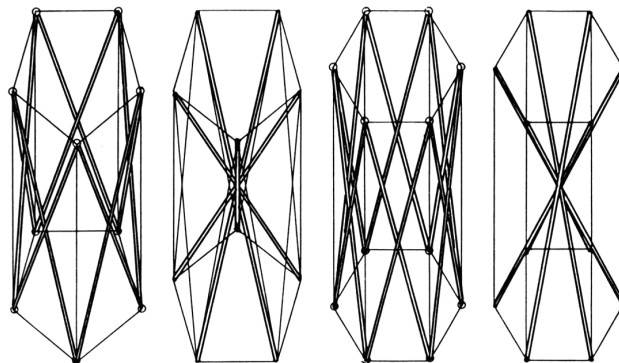
Según que usemos un módulo u otro, durante el despliegue la malla nos ofrecerá un aspecto muy diferente. El despliegue de módulo de aspas sería:



y el de haces resultaría:



Los módulos correspondientes al pentágono y hexágono serían los siguientes:

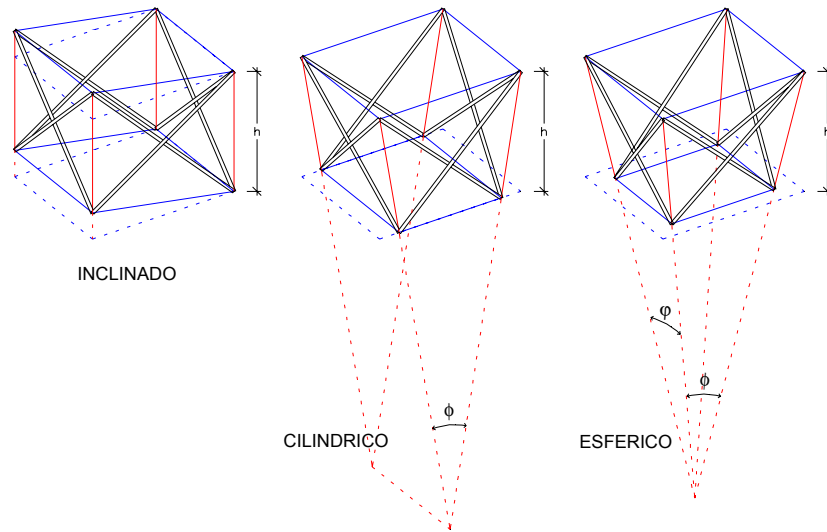


Y así sucesivamente, aunque el módulo pentagonal no nos valdría para cubrir superficies planas por incompatibilidades propias de su geometría. Módulos de mayor número de caras podrían plantearse pero ya no presentan una utilidad práctica ya que serían altamente deformables en planta.

Los cuatro ejemplos nos darían mallas desplegables planas. Los primeros, de base triangular y cuadrada son estudiados por F. Escrig en 1985 con cierto detalle [6], planteando una primera subdivisión de los módulos, clasificándolos como tipos I, II, III y IV. En este estudio ya abre el camino a mallas que se desplieguen sobre superficies distintas al plano.

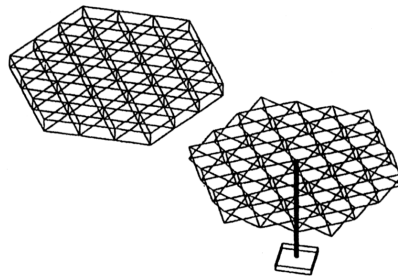


Veamos las posibilidades que se nos abren para el modulo de base cuadrada:

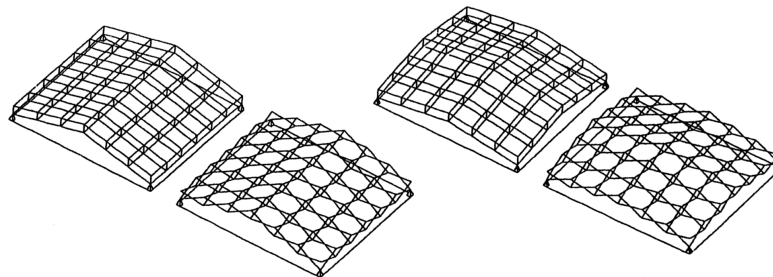


Si mantenemos las bases paralelas, pero inclinamos los planos de las mismas, tenemos un módulo inclinado, que, si mantenemos las aspas verticales, se desarrollará según planos inclinados. Si aproximamos dos de los lados de la base entre sí, tenemos un módulo que se desplegará según superficies cilíndricas sin mayor problema. Si estrechamos los cuatro lados de la base, el módulo desplegará según una superficie esférica, pero en este caso habrá que hacer un estudio más detallado, pues lógicamente, no se puede dividir una superficie esférica con cuadrados, y por tanto los módulos no pueden ser todos iguales.

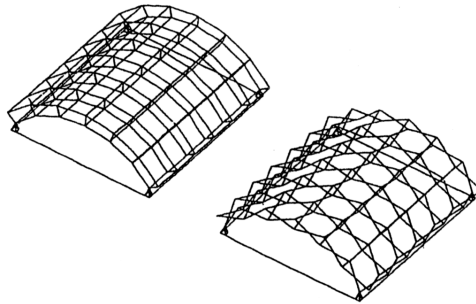
Ejemplo de malla plana:



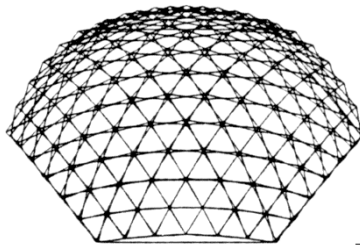
de malla inclinada, a dos y cuatro aguas:



malla cilíndrica:



y malla esférica:



Parece claro entonces que una subdivisión tipológica se formaría clasificando el módulo base según tres parámetros:

**A.- Numero de lados del módulo**

- Triangular
- Cuadrangular
- Pentagonal
- Hexagonal

**B.- Tipo de superficie a cubrir.**

- Plana
- Inclinada
- Cilíndrica
- Esférica.

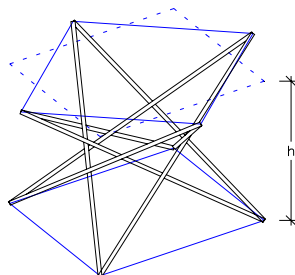
**C.- Tipo de nudo central**

- De aspas.
- De haces.

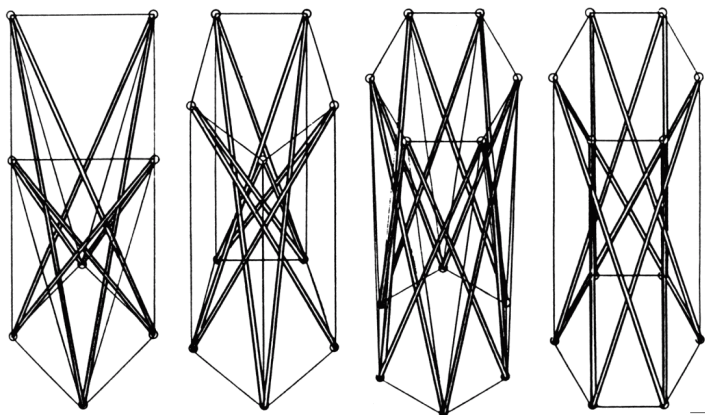
Por tanto un módulo quedaría determinado por los tres parámetros anteriores, por ejemplo:

*"Módulo triangular cilíndrico de haces".*

Parece que con esto ya quedaría cubierta toda la casuística posible, pero no es así. Por ejemplo, si planteamos el antiprisma de base cuadrada podríamos hacer otra distribución de barras que nos ampliarían la clasificación que hicimos en el apartado C (tipo de nudo central):



En 1986 F. Escrig y P. Valcarcel [7] ya nos plantean una variedad de posibilidades para los módulos cuadrado, pentagonal y hexagonal.



Vemos que se nos abre una gran variedad de posibilidades en cuanto a la geometría. Presenta además la ventaja de que cada barra tendría dos uniones intermedias de tipo deslizante, lo que daría mayor estabilidad al conjunto. A cambio presentan el inconveniente de que durante el despliegue las barras giran alrededor de su eje, dificultando la ejecución de las uniones.

## 1.2.4 MALLAS COMPATIBLES E INCOMPATIBLES.

Una característica específica de estas mallas es el hecho de que sean o no compatibles durante el proceso de despliegue. W. Shan define muy bien [8] estos términos, a la vez que acuña el término **Estructuras-P** como forma de hablar de estas estructuras:

*"Estructuras-P pueden ser clasificadas en dos grupos, dependiendo de la forma en como se deforman las aspas durante el proceso de despliegue. Una estructura-p se dice que es 'compatible' siempre que durante el proceso de despliegue no se produzcan deformaciones internas en las barras."*

Parece preciso aclarar un poco los términos, pues toda malla cargada produce un estado tensional en las barras que originan deformaciones en las mismas, por tanto da la impresión de que todas las mallas serían incompatibles. Debemos por tanto realizar una puntualización, especificando que al tipo de deformaciones a que tiene que referir son a las debidas a la flexión de las barras.

Por otra parte parece que a los tres grupos de la clasificación que hemos hecho anteriormente les añade otro más: el de la compatibilidad o incompatibilidad, pero no es así exactamente, sino que será característica intrínseca de cada tipo de módulo.

De este modo, si nos referimos al tipo que usamos como ejemplo unos párrafos atrás:

*"Módulo triangular cilíndrico de haces"*.

No admitiría según esta nueva clasificación tener dos variantes:

*"Módulo triangular cilíndrico de haces compatible"*.

*"Módulo triangular cilíndrico de haces incompatible"*.

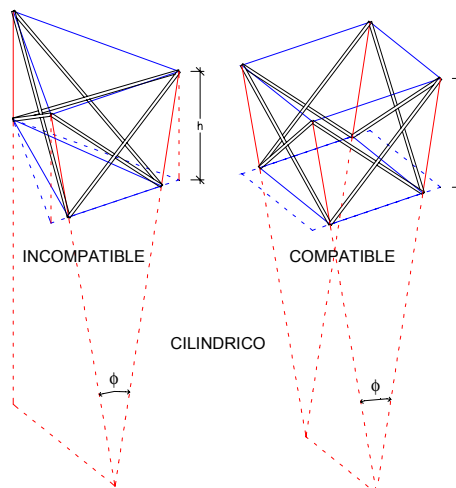
Sino que una vez definido y estudiado el tipo, este devendrá que es o una cosa o la otra. En este ejemplo el estudio nos dice que es:

*"Módulo triangular cilíndrico de haces **incompatible**"*.



## MALLAS CILÍNDRICAS.

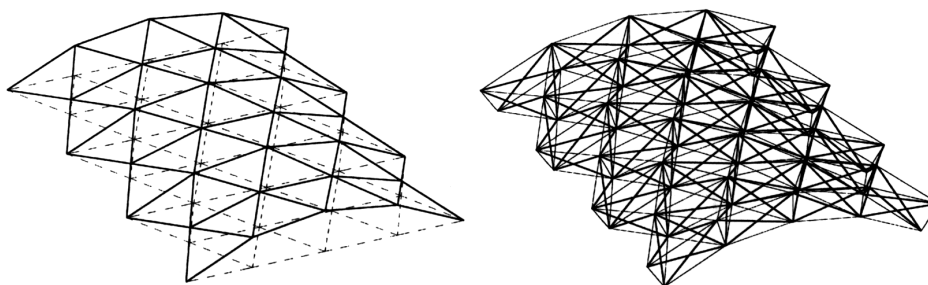
Veremos ahora como podemos saber si una estructura determinada es compatible o no, y para ello podemos empezar estudiando los módulos triangular y cuadrado cilíndricos:



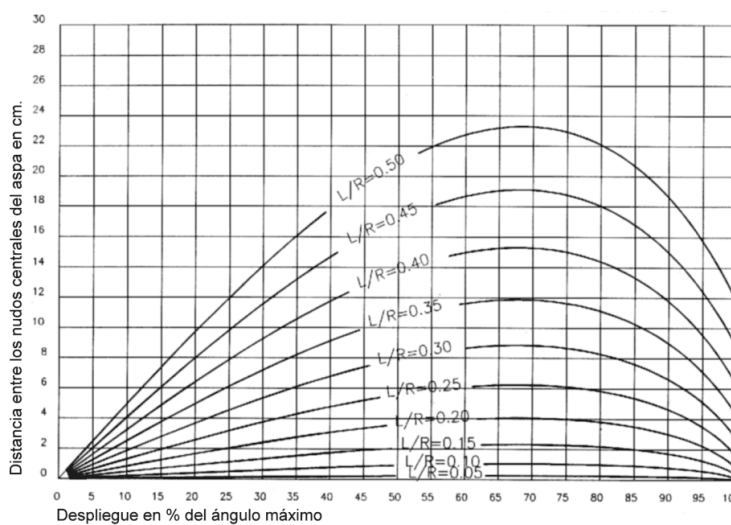
En el módulo cuadrado las aspas están cada una contenida en un plano, y durante el resto del despliegue será así, aunque los planos se vayan inclinando según aumenta el ángulo  $\phi$ .

En el módulo triangular, si bien a la cara frontal le ocurre lo mismo que a la del cuadrado, a las caras laterales derecha e izquierdas les sucede algo más complejo. Si nos fijamos en los cuadriláteros que definen los extremos de las aspas, al pasar del módulo plano al oblicuo, solamente uno de los cuatro vértices se ha desplazado, por lo que ahora dejan de estar los cuatro en el mismo plano y tenemos un cuadrilátero alabeado. Las barras ya no se cortan por tanto en la zona media, sino que se cruzan a una distancia tanto mayor cuanto mayor sea el ángulo de albeo ( $\phi/2$ ) o al disminuir  $h$ , condiciones ambas que se producen al irse desplegando la estructura, lo que significa que si en la posición de plegado los puntos medios estaban unidos mediante un pasador, a medida que se despliegan, las barras deben flexionar cada vez más para permanecer en contacto, por lo que a la energía precisa para el despliegue habrá que añadirle la energía adicional precisa para conseguir la flexión de estas barras. Esto conduce a que estas estructuras tiendan a plegarse espontáneamente, por lo que habrá que fijarlas bien en la posición de despliegue.

En el gráfico siguiente se observa perfectamente este albeo en la sucesión de módulos triangulares.



Ya se han realizado estudios sobre la importancia de este efecto. En el siguiente gráfico, realizado por J.P. Valcarel, se representa la incidencia de la flexión evaluando la distancia total, en cm. que se separan los nudos centrales de las barras si éstas se mantuviesen rectas y se representa en función del porcentaje de despliegue y la relación Luz / Radio.

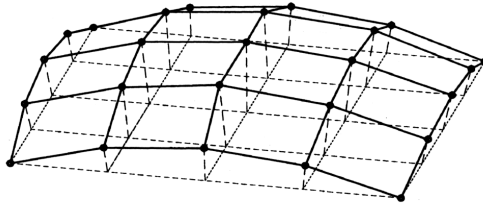


## MALLAS ESFÉRICAS.

Veremos en primer lugar el proceso típico de definición geométrica de estas mallas para luego comentar donde nace en ellas el problema de las incompatibilidades.

El procedimiento habitual sigue los pasos siguientes:

- a.) Definir la malla plana a usar y proyectarla sobre la superficie esférica. El plano donde esté dibujada la malla no es significativo, y solo afectará a la escala de la proyección, de la misma forma que lo haría el variar el tamaño de este módulo base.

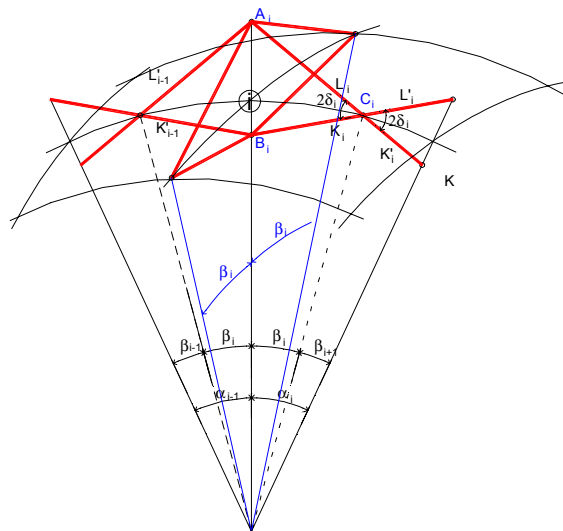


El foco de proyección sí es significativo, pues influirá en que los distintos módulos sean más o menos similares. Si los módulos son muy dispares será más difícil que cumplan las condiciones de plegado y desplegado. Normalmente dan buenos resultados focos situados cerca del polo opuesto a centro de la malla proyectada.

- b.) Una vez que tenemos la malla plana proyectada sobre la esfera debemos hallar los puntos **C**, que estarán en un lugar intermedio de estos segmentos circulares cubiertos por un ángulo que llamamos  $\alpha_i$  que serán datos. Por la condición de plegabilidad sabemos que todos los  $\beta_i$  que concurren en un nudo  $i$  deben ser iguales, de modo que podemos plantear el sistema de ecuaciones:

$$\alpha_i = \beta_i + \beta_{i+1}$$

con tantas ecuaciones como segmentos  $\alpha_i$  tengamos y tantas incógnitas  $\beta_i$  como vértices  $i$  tengamos.



Aplicando las condiciones de simetría eliminaremos todas las ecuaciones idénticas del sistema y nos quedarán las que sean independientes.

Como el número de segmentos (ecuaciones) no tiene por que coincidir con el de vértices (incógnitas) podremos tener un sistema de ecuaciones indeterminado, determinado o incompatible.

Si el sistema es indeterminado, podremos fijar arbitrariamente algunos segmentos  $\beta_i$  hasta convertirlo en determinado. Si es determinado obtendremos directamente las

soluciones, y si es incompatible (más ecuaciones que incógnitas) no será posible obtener valores que cumplan todas las ecuaciones, pero sí se puede optimizar el sistema para obtener soluciones que hagan mínimos los errores. En este tipo de sistemas cobra especial importancia la elección del foco de proyección, que hará que los errores sean mayores o menores.

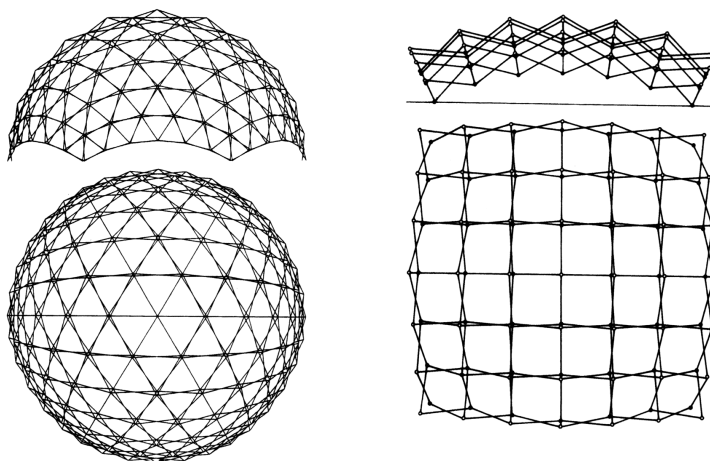
Si por ejemplo hacemos las cuentas para la malla de módulo cuadrado de  $n$  lados, tendremos  $2 \cdot [n \cdot (n+1)]$  segmentos y  $(n+1)^2$  vértices. Si el centro de la malla es un nudo  $i$ , y aplicamos las condiciones de simetría, nos queda un octante de la malla, con  $[n/2 \cdot (n/2+1)]$  segmentos y el mismo número de vértices, por lo que resulta un sistema determinado, con una única solución.

En las mallas triangulares la relación segmentos / vértices es mayor, por lo que los sistemas serán en general incompatibles de mayor o menor grado.

- c.) Por último fijaremos el ángulo de apertura  $2\delta$  que deseemos para las aspas y ya quedará totalmente determinada la geometría. Con este valor  $2\delta$  constante para toda la estructura, y  $\beta_i$  constante para cada nudo, podemos obtener las coordenadas de los nudos superior e inferior y las longitudes  $L_i$  y  $K_i$  de cada barra de la estructura. Hacemos aquí la observación de que los nudos superiores e inferiores no tienen porque estar sobre una superficie esférica ni ser concéntricos.

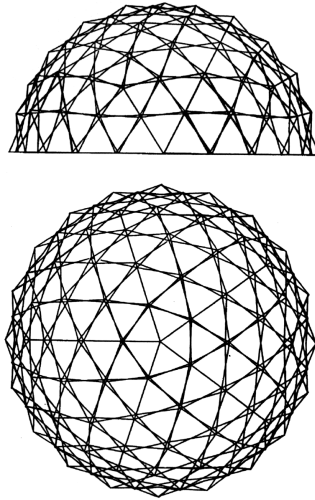
Con el procedimiento descrito obtenemos la geometría en la posición desplegada que cumple las condiciones de plegado y empaquetado cuando el ángulo  $\delta$  se aproxima a los  $90^\circ$ .

Los gráficos siguientes representan mallas de módulo triangular y cuadrado de frecuencia 6, generadas con este método.

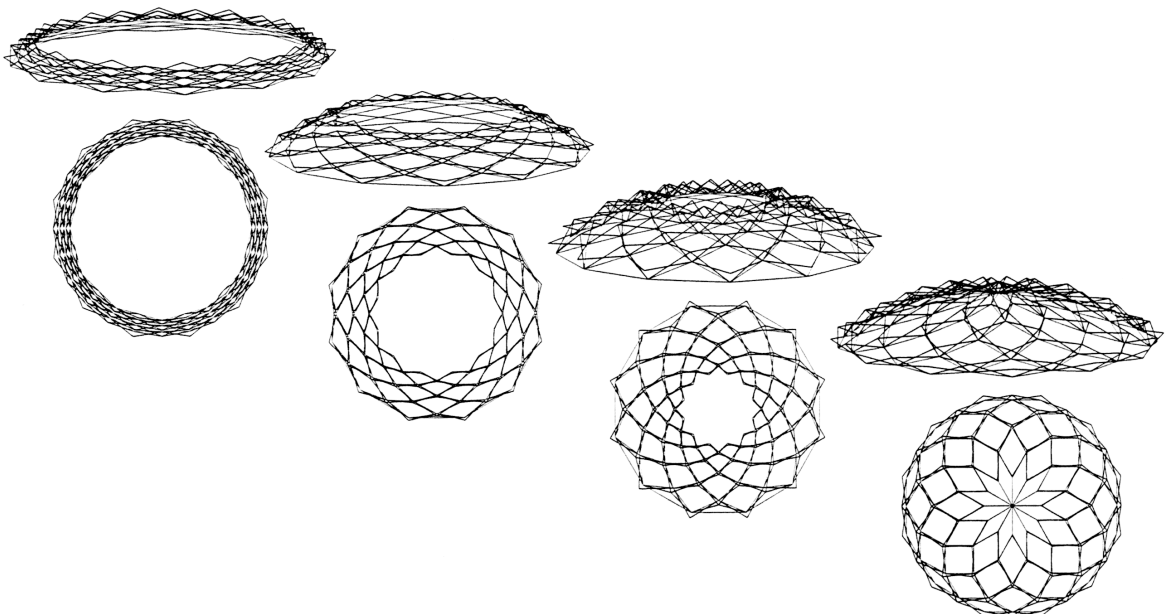


Si se pretende que el ángulo cubierto por la malla desplegada sea superior a  $90^\circ$  o  $100^\circ$ , en este caso puede ser interesante que en vez de usar una malla plana para proyectar, usemos un poliedro regular inscrito en la circunferencia sobre cuyas caras estarán dibujadas las mallas a proyectar y el foco de proyección sería en este caso el centro de la esfera.

Estaríamos hablando entonces de las **mallas geodésicas** y su aspecto sería similar a la figura siguiente.



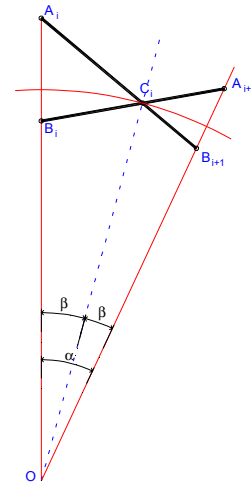
Para el caso particular del módulo cuadrado, también es posible ir trazando la malla directamente sobre la superficie esférica a partir de segmentos de igual longitud, lo que presenta la ventaja de que todas las barras serían de igual longitud simplificando el proceso constructivo. Usando módulos cuadrangulares de lados iguales también podemos construir otro tipo de cúpulas muy interesantes. El estado desplegado se diseña a partir de la cúspide, de donde parten un número determinado de rombos (generalmente 10 o 12) que se unen por su vértice. De estos arranca el anillo siguiente de rombos y así sucesivamente, de modo que cada anillo tiene el mismo número de rombos que el anterior, pero cada vez se ensancharán más al ir correspondiendo con un paralelo de mayor diámetro. Este planteamiento tiene el interés de que se pueden plantear cubiertas retráctiles hacia el borde perimetral, ya que obtenemos una solución muy limpia para despejar el centro, aunque no es tan práctica por el transporte al no reducirse a un paquete compacto.



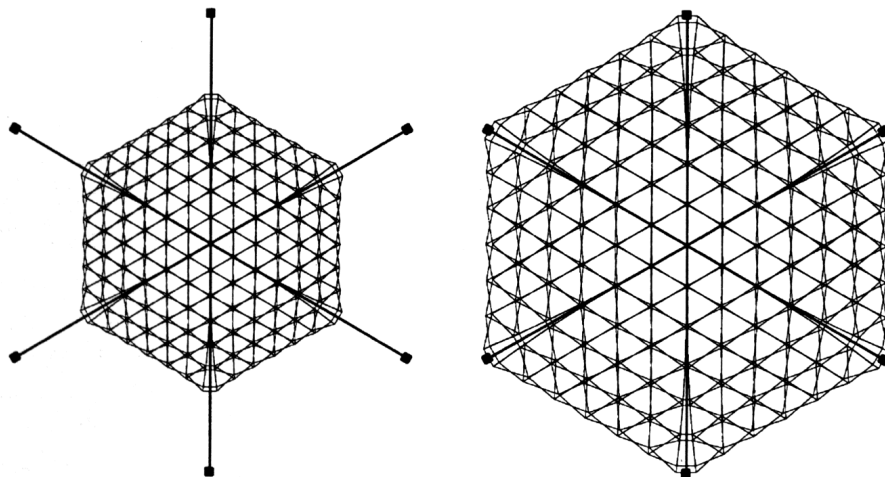


Pero retomamos nuestro caso de estudio. En las mallas esféricas vemos que la incompatibilidad debida al alabeo a que hacíamos referencia en las mallas cilíndricas no se produce, pues la forma de generación nos garantiza que los puntos extremos del aspa  $A_i$  y  $B_i$  son coplanarios con los  $A_{i+1}$  y  $B_{i+1}$  puesto que están sobre dos rectas que se cortan en un punto que es el centro de la esfera  $O$ .

La incompatibilidad en este caso se debe a que en el proceso de generación hemos fijado una geometría compatible en el estado desplegado (o casi, minimizando el error en la medida de lo posible) e impuesto las condiciones que nos aseguran la posición de plegado, pero nada nos garantiza la compatibilidad en las posiciones intermedias, que generalmente presentan unas incompatibilidades más o menos manifiestas. Estas incompatibilidades se resolverán, como ya hemos comentado, con la flexión de las barras, salvo que se diseñe alguna solución constructiva para el nudo intermedio que dé mayor libertad al movimiento de las barras.



Este problema se ha estudiado [9] con un casquete de 60 m. de diámetro y 15 m. de altura. Se ha ido generando la estructura a partir de las longitudes ya conocidas de las barras para los distintos ángulos  $\delta$  de posiciones intermedias y a partir de la cúspide. De esta forma se montaron los 6 'gajos' que forman la cúpula. En las posiciones inicial y final, como era de esperar, encajaban perfectamente, pero en las posiciones intermedias se apreciaban unas holguras significativas entre los gajos, que son indicativo de las deformaciones por flexión y tracción que deberemos producir en las barras para que el conjunto cierre perfectamente.

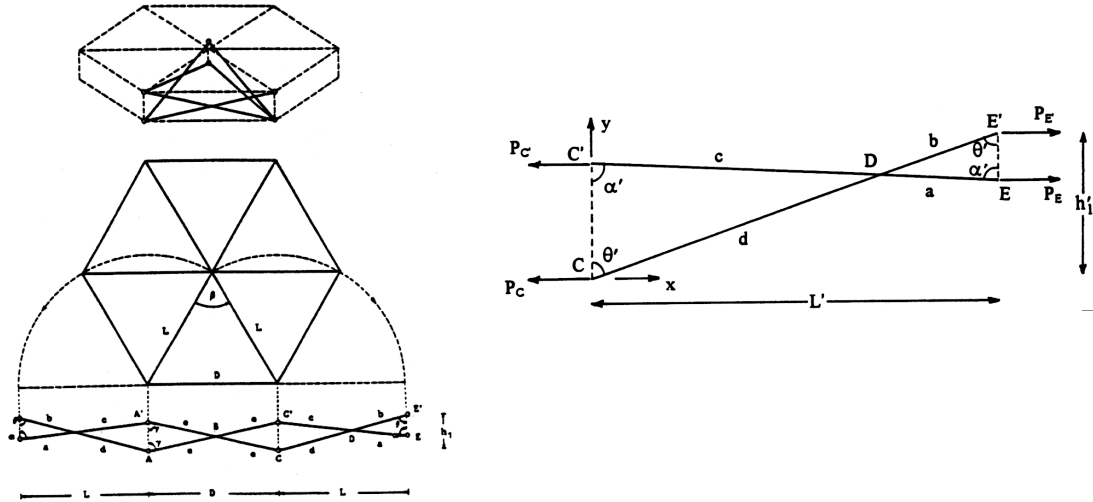


Esto significa que en las posiciones intermedias del despliegue hay que suministrar energía adicional a la malla, que la malla empezará a devolver al ir finalizando el despliegue. Por tanto esta posición final será bastante estable.

Estas incompatibilidades durante el despliegue hacen que una vez pasado el punto de máxima resistencia, se produce una expansión brusca, o **snap-trough** como la define Gantes, que es el

autor de un estudio de la problemática de esta fase, e intenta conseguir un modelo simplificado de cálculo [10] que consiga una evaluación de los esfuerzos ya en los preliminares del diseño.

El esquema de partida es el de una malla de frecuencia 1, como la siguiente, a partir de la cual proyectará sus conclusiones para mallas de cualquier frecuencia.



Realizado un estudio, resuelve que debido a las incompatibilidades geométricas se debe someter a las barras a una compresión aproximada:

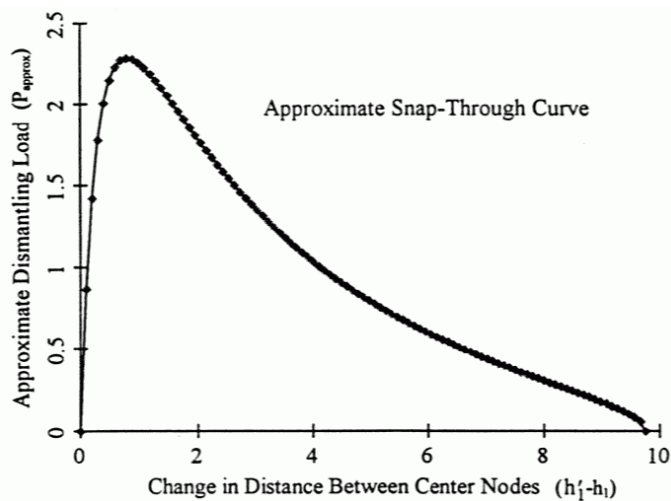
$$P_{\text{aprox}} = E \cdot A_{\text{eq}} \cdot \frac{\Delta L}{L}$$

Siendo:

$A_{\text{eq}}$  - sección equivalente de las barras, y

$$\Delta L = L' - L = (a_c) \cdot \text{sen} \alpha - \frac{\sqrt{e^2 - \frac{(c \cdot \cos \alpha + d \cdot \cos \theta)}{4}}}{\text{sen} \frac{\beta}{2}}$$

Realiza una gráfica que relaciona la posición de despliegue ( $h_1' - h_1$ ) con esta carga estimada, resultando:



Vemos que es en las primeras fases del despliegue cuando es preciso suministrar energía adicional a la estructura, que nos irá devolviendo una vez pasado el punto máximo, el cual dependerá de la geometría concreta de la malla.

## REFERENCIAS:

- [1] F. Escrig  
"Las estructuras de Emilio Pérez Piñero". Textos de arquitectura. E. T. S. A. de Sevilla. 1993
- [2] Calatrava, S.  
"Sobre la plegabilidad de entramados". Textos de arquitectura. E. T. S. A. de Sevilla. 1993
- [3] Hernández, C. H.  
"New ideas on deployable structures"  
Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton 1996.
- [4] Escrig, F.  
"Geometrías de las Estructuras Desplegables de Aspas"  
Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993
- [5] Kawaguchi, K. & Mangai, Y  
"Numerical anlysis for folding of space structures"  
Space Seructures. Parke, Thomas Telford. London. 1993
- [6] F. Escrig  
"Expandable space frame structures". Third International Conference on Space Strutures. Elsevier Applied Science Publishers. H. Nooshin.1985
- [7] Escrig, F. & Valcarcel, J. P.  
"Introducción a la geometría de Estructuras Desplegables de Barras"  
Boletín Académico, nº 3. ETSA Coruña. 1986.
- [8] Shan, W.  
"Configuration Studies of foldable Structures"  
Space Structures. Parke, Thomas Telford. London. 1993
- [9] Sánchez Sánchez, J.  
"Estructuras Desplegables de Aspas para Mallas Polédricas Curvas"  
Tesis Doctoral. Universidad de Sevilla. E.T.S. Arquitectura. Septiembre 1996
- [10] Gantes, C.  
"Analytical predictions of the snap-through characteristics of deployable structures"  
Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Pub. Southampton 1996.

---

## 1.3 CÁLCULO ESTRUCTURAL

---

La mayor parte de la bibliografía que hace referencia al cálculo de estas estructuras (ver referencias [1], [2], [3], [4] y [5]) se ciñen al cálculo de esfuerzos en una posición dada de la cúpula, ya sea la inicial, la final o una intermedia. Se trata por tanto de un cálculo matricial de tipo estático que ya está plenamente resuelto, con grandes refinamientos como vemos los últimos artículos de P. Valcarcel y F. Escrig [5] [6], como pueden ser el estudio no lineal debido a las deformaciones de la estructura, plantear barras inicialmente flexionadas, consideraciones de la excentricidad debida a la curvatura de las barras y las excentricidades de los nudos, etc.

Respecto al cálculo dinámico. Charis Gantes empieza sus estudios con el programa ADINA, que simulen el comportamiento de la estructura real, y estudia el despliegue a través del método de Newton-Raphson.

A continuación expondremos sucintamente una visión del estado actual de este tema, estudiando la formulación matricial de las barras de estas estructuras.

---

### 1.3.1 CÁLCULO MATRICIAL

---

Antes de empezar a hablar más a fondo de este tema, exponemos un breve resumen del método que servirá a la vez para definir la terminología que usaremos para las matrices .

#### 1.- Ley de Hooke.

Se expresan los esfuerzos en las barras en función de los desplazamientos de sus extremos.

$$P = K \cdot Z$$

P - vector fuerzas interiores de las barras.

K - matriz de rigidez en coordenadas locales.

Z - vector de desplazamientos de los nudos.

#### 2.- Ecuaciones de compatibilidad.

Se ponen los movimientos en los extremos de las barras (coordenadas locales) en función de los movimientos de los nudos (coordenadas globales).

$$Z = A \cdot X$$

A - matriz de transformación.

X - desplazamientos de los nudos.

#### 3.- Ecuaciones de equilibrio.

Se aplican las condiciones de equilibrio de fuerzas en los nudos.

$$L = A \cdot P$$

L - vector de fuerzas exteriores.  
A' - traspuesta de a.

#### 4.- Resolución.

$$L = A' \cdot P$$

$$L = A' \cdot K \cdot Z$$

$$L = A' \cdot K \cdot A \cdot X$$

Haciendo:  $A' \cdot K \cdot A = S$  matriz de rigidez en coordenadas globales.

$$L = S \cdot X$$

Teníamos como datos L y como incógnitas X, por tanto podremos despejar X.

Una vez hallada X conoceremos las fuerzas en las barras mediante la ecuación:

$$P = K \cdot A \cdot X$$

Pasaremos ahora al estudio detallado de las estructuras que nos interesan para conocer como está el estado de la cuestión hoy en día.

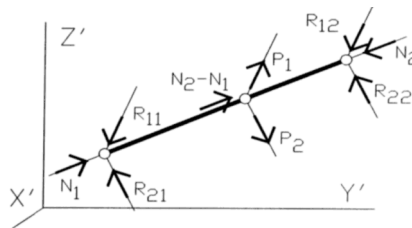
## ■ BARRAS CON ARTICULACIÓN CENTRAL FIJA

Para el desarrollo matricial se ha optado por una formulación muy simple de la matriz de rigidez que conlleva uno más complejo de lo habitual para la matriz de compatibilidad.

### MATRIZ DE RIGIDEZ EN COORDENADAS LOCALES.

A efectos de los axiles, la barra se comportará como dos articuladas alineadas, de axiles  $N_1$  y  $N_2$  y alargamientos  $u_1$  y  $u_2$ .

$$N_1 = \frac{E \cdot A}{L_1} \cdot u_1 \quad ; \quad N_2 = \frac{E \cdot A}{L_2} \cdot u_2$$



Respecto las deformaciones transversales, suponemos unas cargas transversales  $P_1$  y  $P_2$  en el nudo intermedio que provocará unos desplazamientos del mismo  $v$  y  $w$ .

$$P_1 = \frac{3 \cdot E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot w \quad ; \quad P_2 = \frac{3 \cdot E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot w$$

Poniendo esto en forma matricial:

$$\begin{bmatrix} N_1 \\ N_2 \\ P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} \frac{E \cdot A}{L_1} & 0 & 0 & 0 \\ 0 & \frac{E \cdot A}{L_2} & 0 & 0 \\ 0 & 0 & \frac{3 \cdot E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} & 0 \\ 0 & 0 & 0 & \frac{3 \cdot E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix} \quad ; \quad \text{o sea } [P] = [K] \cdot [Z]$$

Tengamos en cuenta aquí que los desplazamientos que empleamos son los significativos de la barra, que derivarían de los desplazamientos locales de los nudos de la barra según las siguientes expresiones:

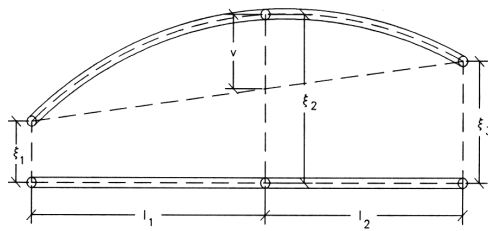
$$u_1 = \psi_2 - \psi_1$$

$$u_2 = \psi_3 - \psi_2$$

$$v = \xi_2 - \frac{L_2}{L} \cdot \xi_1 - \frac{L_1}{L} \cdot \xi_3$$

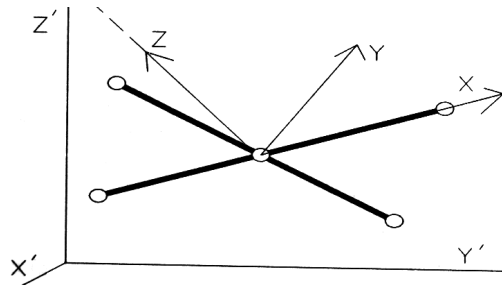
$$w = \eta_2 - \frac{L_2}{L} \cdot \eta_1 - \frac{L_1}{L} \cdot \eta_3$$

siendo  $\psi$ ,  $\xi$ ,  $\eta$  los desplazamientos según los ejes locales  $x$ ,  $y$ ,  $z$  respectivamente. Así por ejemplo, para el cálculo de  $v$  usaríamos el siguiente esquema:



#### MATRIZ DE COMPATIBILIDAD.

Como ejes locales tomaremos el  $X$  coincidente con el de la barra, el  $Y$  como la dirección ortogonal al aspa y el  $Z$  el que resulte ortogonal a los otros dos y nos dé un triedro directo.



Sus cosenos directores serán:

$$\text{eje X: } \cos \alpha_1, \cos \beta_1, \cos \gamma_1$$

$$\text{eje Y: } \cos \alpha_2, \cos \beta_2, \cos \gamma_2$$

$$\text{eje Z: } \cos \alpha_3, \cos \beta_3, \cos \gamma_3$$

Podemos poner entonces los desplazamientos locales de la barra ( $u_1$ ,  $u_2$ ,  $v$ ,  $w$ ) en función de los desplazamientos globales de los nudos ( $x_1$ ,  $y_1$ ,  $z_1$ ,  $x_2$ ,  $y_2$ ,  $z_2$ ,  $x_3$ ,  $y_3$ ,  $z_3$ ) siendo en este caso el nudo 1 el inicial, 2 el intermedio y 3 el final. Nos queda la expresión matricial:

$$\begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix} = \begin{bmatrix} -\cos \alpha_1 & -\cos \beta_1 & -\cos \gamma_1 & \cos \alpha_1 & \cos \beta_1 & \cos \gamma_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\cos \alpha_1 & -\cos \beta_1 & -\cos \gamma_1 & \cos \alpha_1 & \cos \beta_1 & \cos \gamma_1 \\ -\frac{L_2 \cdot \cos \alpha_2}{L} & -\frac{L_2 \cdot \cos \beta_2}{L} & -\frac{L_2 \cdot \cos \gamma_2}{L} & \cos \alpha_2 & \cos \beta_2 & \cos \gamma_2 & -\frac{L_1 \cdot \cos \alpha_2}{L} & -\frac{L_1 \cdot \cos \beta_2}{L} & -\frac{L_1 \cdot \cos \gamma_2}{L} \\ -\frac{L_2 \cdot \cos \alpha_3}{L} & -\frac{L_2 \cdot \cos \beta_3}{L} & -\frac{L_2 \cdot \cos \gamma_3}{L} & \cos \alpha_3 & \cos \beta_3 & \cos \gamma_3 & -\frac{L_1 \cdot \cos \alpha_3}{L} & -\frac{L_1 \cdot \cos \beta_3}{L} & -\frac{L_1 \cdot \cos \gamma_3}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ x_2 \\ y_2 \\ z_2 \\ x_3 \\ y_3 \\ z_3 \end{bmatrix}$$

o sea  $[Z] = [A] \cdot [X]$

## MATRIZ DE RIGIDEZ EN COORDENADAS GLOBALES.

Si nuestra ecuación de equilibrio es:

$$[L] = [A'] \cdot [P]$$

siendo L la matriz de cargas externas y P la de esfuerzos en las barras

en consecuencia:

$$[L] = [A'] \cdot [K] \cdot [A] \cdot [X] = [S] \cdot [X]$$

siendo [S] la matriz de rigidez en coordenadas globales, y resultando de valor:

$$\begin{bmatrix} a+m \cdot k_1 & b+n \cdot k_1 & c+o \cdot k_1 & -a-m \cdot k_3 & -b-n \cdot k_3 & -c-o \cdot k_3 & m \cdot k_5 & n \cdot k_5 & o \cdot k_5 \\ & d+p \cdot k_1 & e+q \cdot k_1 & & -d-p \cdot k_3 & -e-q \cdot k_3 & & p \cdot k_5 & q \cdot k_5 \\ & & f+r \cdot k_1 & \text{simetrica} & & -f-r \cdot k_3 & \text{simetrica} & & r \cdot k_5 \\ \hline & & & a+g+m & b+h+n & c+i+o & -g-m \cdot k_4 & -h-n \cdot k_4 & -i-o \cdot k_4 \\ & & & & d+j+p & e+k+q & & -j-p \cdot k_4 & -k-q \cdot k_4 \\ & & & & & f+l+r & \text{simetrica} & & -l-r \cdot k_4 \\ \hline & \text{simetrica} & & & & & g+m \cdot k_2 & h+n \cdot k_2 & i+o \cdot k_2 \\ & & & & & & & j+p \cdot k_2 & k+q \cdot k_2 \\ & & & & & & & & l+r \cdot k_2 \end{bmatrix}$$

siendo:

$$k_1 = \frac{L_2^2}{L^2} ; k_2 = \frac{L_1^2}{L^2} ; k_3 = \frac{L_2}{L} ; k_4 = \frac{L_1}{L} ; k_5 = \frac{L_1 \cdot L_2}{L}$$

y los coeficientes:

$$\begin{aligned} a &= \frac{E \cdot A}{L_1} \cdot \cos^2 \alpha_1 & g &= \frac{E \cdot A}{L_2} \cdot \cos^2 \alpha_1 & m &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \alpha_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \alpha_3 \\ b &= \frac{E \cdot A}{L_1} \cdot \cos \alpha_1 \cdot \cos \beta_1 & n &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \alpha_2 \cdot \cos \beta_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \alpha_3 \cdot \cos \beta_3 \\ c &= \frac{E \cdot A}{L_1} \cdot \cos \alpha_1 \cdot \cos \gamma_1 & o &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \alpha_2 \cdot \cos \gamma_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \alpha_3 \cdot \cos \gamma_3 \\ d &= \frac{E \cdot A}{L_1} \cdot \cos^2 \beta_1 & p &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \beta_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \beta_3 \\ e &= \frac{E \cdot A}{L_1} \cdot \cos \beta_1 \cdot \cos \gamma_1 & q &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \beta_2 \cdot \cos \gamma_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos \beta_3 \cdot \cos \gamma_3 \\ f &= \frac{E \cdot A}{L_1} \cdot \cos^2 \gamma_1 & r &= \frac{3E \cdot I_1 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \gamma_2 + \frac{3E \cdot I_2 \cdot L}{L_1^2 \cdot L_2^2} \cdot \cos^2 \gamma_3 \end{aligned}$$

## CÁLCULO DE ESFUERZOS

De la ecuación general  $[L] = [S] \cdot [X]$  despejamos las incógnitas  $[X]$ , y podemos convertir los desplazamientos a coordenadas locales de las barras con la ecuación ya vista:  $[Z] = [A] \cdot [X]$

Una vez obtenida  $[Z]$  tenemos los esfuerzos axiles en las barras a partir de la primera ecuación vista:

$$N_1 = \frac{E \cdot A}{L_1} \cdot u_1 \quad ; \quad N_2 = \frac{E \cdot A}{L_2} \cdot u_2$$

Los cortantes y flectores los podemos deducir, ya que tenemos también el valor de:

$$P_1 = \frac{3 \cdot E \cdot I_1 \cdot L}{L_1^2 \cdot L_2} \cdot w \quad ; \quad P_2 = \frac{3 \cdot E \cdot I_2 \cdot L}{L_1^2 \cdot L_2} \cdot w$$

Por tanto Los cortantes valdrán:

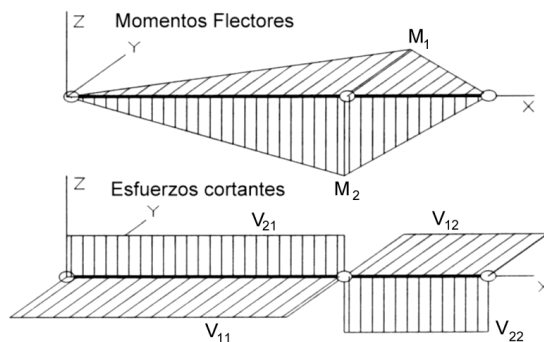
$$V_{11} = \frac{P_1 \cdot L_2}{L} \quad V_{12} = \frac{P_1 \cdot L_1}{L}$$

$$V_{21} = \frac{P_2 \cdot L_2}{L} \quad V_{22} = \frac{P_2 \cdot L_1}{L}$$

Y los momentos flectores:

$$M_1 = \frac{P_1 \cdot L_1 \cdot L_2}{L} \quad ; \quad M_2 = \frac{P_2 \cdot L_1 \cdot L_2}{L}$$

Que podemos representar gráficamente:



## COMPROBACIÓN DE RESULTADOS.

Como verificación de que el proceso ha resultado bien, podemos calcular las fuerzas que las barras ejercen sobre los nudos para verificar que estos se hallen en equilibrio:

$$\text{Fuerzas desequilibradas} = [L] - [S] \cdot [X]$$

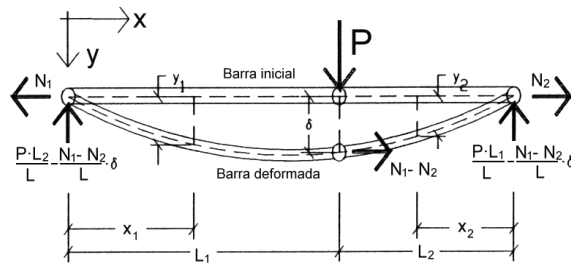
Los cálculos realizados según el esquema son suficientemente satisfactorios para la mayoría de los casos, pero cuando se pretende conseguir grandes luces los efectos de segundo orden son importantes por ello pasaremos ahora a ver que efectos deberemos estudiar en estos casos.



## 1.3.2 CONSIDERACIONES DE SEGUNDO ORDEN

### EFFECTO DEL AXIL EN LA FLEXIÓN.

En la formulación anterior no se tuvo en cuenta que los axiles provocan una variación de la deformación transversal de la barra, ya que si tenemos en cuenta la deformación de la misma, producen una variación de la ley de flectores. Por tanto, si tenemos esto en cuenta, el esquema de fuerzas es más complejo:



y esto afectaría a la matriz de rigidez en coordenadas locales, que pasaría a ser:

$$\begin{bmatrix} N_1 \\ N_2 \\ P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} \frac{E \cdot A}{L_1} & 0 & 0 & 0 \\ 0 & \frac{E \cdot A}{L_2} & 0 & 0 \\ 0 & 0 & \frac{3 \cdot E \cdot I_1 \cdot L}{\phi_1 \cdot L_1^2 \cdot L_2^2} & 0 \\ 0 & 0 & 0 & \frac{3 \cdot E \cdot I_2 \cdot L}{\phi_2 \cdot L_1^2 \cdot L_2^2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix}$$

donde aparecen los nuevos términos  $\phi_1$  y  $\phi_2$ , que son las funciones de estabilidad, que definiremos como la relación entre la flecha en la hipótesis lineal y la no lineal (teniendo en cuenta la influencia del axil en la flexión de la barra). Más adelante definiremos estas funciones.

La matriz de compatibilidad quedaría igual, pero la matriz de rigidez en coordenadas globales también quedaría afectada por este cambio ya que:

$$[S] = [A'] \cdot [K] \cdot [A]$$

### Cálculo de las funciones de estabilidad.

Teniendo en cuenta el gráfico anterior, las nuevas ecuaciones de momentos en los tramos 1 y 2 serán ahora:

$$\begin{aligned} M_1 &= \frac{P \cdot L_2}{L} \cdot x + \frac{N_1 - N_2}{L} \cdot \delta \cdot x + N_1 \cdot y_1 \\ M_2 &= \frac{P \cdot L_2}{L} \cdot (L - x) + \frac{N_1 - N_2}{L} \cdot \delta \cdot (L - x) + N_2 \cdot y_2 \end{aligned}$$

Planteando la ecuación diferencial de la elástica para ambos tramos:

$$E \cdot I \frac{d^2 y_1}{dx^2} = -M_1$$

$$E \cdot I \frac{d^2 y_2}{dx^2} = -M_2$$

Integrando estas ecuaciones diferenciales e imponiendo las condiciones de contorno, se obtiene el valor de la flecha en la articulación central, que será:

$$\delta = \frac{P_1 L_1^2 L_2^2}{E \cdot I L} \frac{-\left(\frac{1}{p_1^2 L_1} + \frac{1}{p_2^2 L_2}\right) + \frac{1}{p_1 \operatorname{th}(p_1) L_1} + \frac{1}{p_2 \operatorname{th}(p_2) L_2}}{L_1 L_2 \left[ \frac{p_1}{\operatorname{th}(p_1) L_1} (1 - q_1 L_1) + \frac{p_2}{\operatorname{th}(p_2) L_2} (1 - q_2 L_2) + (q_1 - q_2) \right]}$$

con:

$$p_1 = \sqrt{\frac{|N_1|}{E \cdot I}} ; \quad p_2 = \sqrt{\frac{|N_2|}{E \cdot I}} ; \quad q_1 = \frac{N_1 - N_2}{N_1} ; \quad q_2 = \frac{N_1 - N_2}{N_2}$$

Definimos por tanto la función de estabilidad como:

$$\phi = \frac{\delta}{v} = \frac{\delta}{\frac{P_1 L_1^2 L_2^2}{3 E \cdot I L}} \quad \text{que dará lugar a } \phi_1 \text{ y } \phi_2 \text{ según que plano coordenado local la estudiemos.}$$

Según los valores de  $N_1$  y  $N_2$ , la función de estabilidad puede derivar en alguno de los cinco casos siguientes:

a.)  $N_1 < 0$  y  $N_2 < 0$

$$\phi = 3 \frac{\left(\frac{1}{p_1^2 L_1} + \frac{1}{p_2^2 L_2}\right) - \frac{1}{p_1 \tan(p_1) L_1} - \frac{1}{p_2 \tan(p_2) L_2}}{L_1 L_2 \left[ \frac{p_1}{\tan(p_1) L_1} (1 - q_1 L_1) + \frac{p_2}{\tan(p_2) L_2} (1 - q_2 L_2) + (q_1 - q_2) \right]}$$

b.)  $N_1 > 0$  y  $N_2 > 0$

$$\phi = 3 \frac{-\left(\frac{1}{p_1^2 L_1} + \frac{1}{p_2^2 L_2}\right) - \frac{1}{p_1 \operatorname{th}(p_1) L_1} - \frac{1}{p_2 \operatorname{th}(p_2) L_2}}{L_1 L_2 \left[ \frac{p_1}{\operatorname{th}(p_1) L_1} (1 - q_1 L_1) + \frac{p_2}{\operatorname{th}(p_2) L_2} (1 + q_2 L_2) + (q_1 - q_2) \right]}$$

c.)  $N_1 < 0$  y  $N_2 > 0$

$$\phi = 3 \frac{\left(\frac{1}{p_1^2 L_1} + \frac{1}{p_2^2 L_2}\right) - \frac{1}{p_1 \tan(p_1) L_1} + \frac{1}{p_1 \operatorname{th}(p_2) L_2}}{L_1 L_2 \left[ \frac{p_1}{\tan(p_1) L_1} (1 - q_1 L_1) + \frac{p_2}{\operatorname{th}(p_2) L_2} (1 + q_2 L_2) + (q_1 - q_2) \right]}$$

d.)  $N_1 < 0$  y  $N_2 = 0$

$$\phi = \frac{\frac{3}{p_1^2 L_1} - \frac{3}{p_1 \tan(p_1) L_1} + L_2}{L_1 L_2 \left[ \frac{p_1}{\tan(p_1) L_1} \frac{L_2}{L} + \frac{1}{L} + \frac{1}{L_2} - \frac{p_1^2 L_2^2}{3L} \right]}$$

e.)  $N_1 < 0$  y  $N_2 = 0$

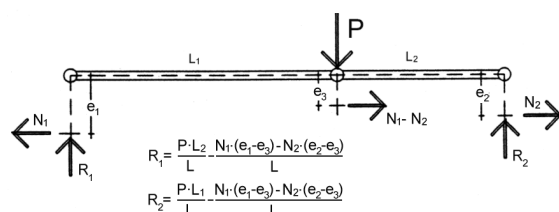
$$\phi = \frac{-\frac{3}{\rho_1^2 \cdot L_1} + \frac{3}{\rho_1 \tan(\rho_1) \cdot L_1} + L_2}{L_1 \cdot L_2 \cdot \left[ \frac{\rho_1}{\tan(\rho_1) \cdot L_1} \cdot \frac{L_2}{L} + \frac{1}{L} + \frac{1}{L_2} + \frac{\rho_1^2 \cdot L_2^2}{3L} \right]}$$

## EFECTO DE LA EXCENTRICIDAD DE LOS NUDOS

Las dimensiones reales de las barras y nudos hace imposible que los ejes de las barras coincidan en un mismo punto. Esto se ha intentado resolver de varias formas, e incluso P. Piñeiro intentó resolver este problema usando barras curvas, para que el cruce de las barras no presentase problemas, pero las dificultades constructivas de esta solución no compensan frente a la posibilidad de efectuar un cálculo más detallado teniendo en cuenta las excentricidades reales de los nudos.

Lo mismo que ocurría con el efecto del axil ya visto, las excentricidades de los nudos conllevan una variación de las leyes de momentos, y por tanto repercutirán en la formación de la matriz de rigidez.

El esquema de cargas que deberemos tomar ahora para una barra será el siguiente:



La energía de deformación de la barra puede calcularse como:

$$W = \frac{1}{2} \int_a^b \left( \frac{M^2}{EI} + \frac{f_c \cdot V^2}{G \cdot A} + \frac{N^2}{E \cdot A} \right) ds$$

Normalmente despreciamos el efecto del cortante en la deformación de la pieza, y por tanto como término de la energía de deformación, pudiendo obtenerse los desplazamientos de los puntos extremos y central como:

$$u_1 = \int_a^b \left( M \cdot \frac{\partial M}{\partial N_1} + N \cdot \frac{\partial N}{\partial N_1} \right) ds \quad ; \quad u_2 = \int_a^b \left( M \cdot \frac{\partial M}{\partial N_2} + N \cdot \frac{\partial N}{\partial N_2} \right) ds$$

$$v = \int_a^b \left( M \cdot \frac{\partial M}{\partial P_1} + N \cdot \frac{\partial N}{\partial P_1} \right) ds \quad ; \quad w = \int_a^b \left( M \cdot \frac{\partial M}{\partial P_2} + N \cdot \frac{\partial N}{\partial P_2} \right) ds$$

Resolviendo estas integrales obtenemos la matriz de flexibilidad de la barra, que resulta:

$$\begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{14} \\ f_{21} & f_{22} & f_{23} & f_{24} \\ f_{31} & f_{32} & f_{33} & f_{34} \\ f_{41} & f_{42} & f_{43} & f_{44} \end{bmatrix} \begin{bmatrix} N_1 \\ N_2 \\ P_1 \\ P_2 \end{bmatrix}$$

con:

$$\begin{aligned}
f_{11} &= \frac{L_1}{EA} + \frac{e_{1y}^2 L_1}{EI_y} - \frac{e_{1y}(e_{1y} - e_{3y})L_1^2}{LEI_y} + \frac{(e_{1y} - e_{3y})^2(L_1^3 + L_2^3)}{3L^2 EI_y} + \frac{e_{1z}^2 L_1}{EI_z} - \frac{e_{1z}(e_{1z} - e_{3z})L_1^2}{LEI_z} + \frac{(e_{1z} - e_{3z})^2(L_1^3 + L_2^3)}{3L^2 EI_z} \\
f_{12} = f_{21} &= \frac{e_{1y}(e_{2y} - e_{3y})L_1^2}{2LEI_y} + \frac{e_{2y}(e_{1y} - e_{3y})L_2^2}{2LEI_y} - \frac{(e_{1y} - e_{3y})(e_{2y} - e_{3y})(L_1^3 + L_2^3)}{3L^2 EI_y} + \frac{e_{1z}(e_{2z} - e_{3z})L_1^2}{2LEI_z} + \frac{e_{2z}(e_{1z} - e_{3z})L_2^2}{2LEI_z} - \frac{(e_{1z} - e_{3z})(e_{2z} - e_{3z})(L_1^3 + L_2^3)}{3L^2 EI_z} \\
f_{13} = f_{31} &= \frac{e_{1y}L_1^2 L_2}{2LEI_y} - \frac{(e_{1y} - e_{3y})(L_1^3 L_2 - L_1 L_2^3)}{3L^2 EI_y} \\
f_{14} = f_{41} &= \frac{e_{1z}L_1^2 L_2}{2LEI_z} - \frac{(e_{1z} - e_{3z})(L_1^3 L_2 - L_1 L_2^3)}{3L^2 EI_z} \\
f_{22} &= \frac{L_2}{EA} + \frac{e_{2y}^2 L_2}{EI_y} - \frac{e_{2y}(e_{2y} - e_{3y})L_2^2}{LEI_y} + \frac{(e_{2y} - e_{3y})^2(L_1^3 + L_2^3)}{3L^2 EI_y} + \frac{e_{2z}^2 L_2}{EI_z} - \frac{e_{2z}(e_{2z} - e_{3z})L_2^2}{LEI_z} + \frac{(e_{2z} - e_{3z})^2(L_1^3 + L_2^3)}{3L^2 EI_z} \\
f_{23} = f_{32} &= \frac{e_{2y}L_1 L_2^2}{2LEI_y} + \frac{(e_{2y} - e_{3y})(L_1^3 L_2 - L_1 L_2^3)}{3L^2 EI_y} \\
f_{24} = f_{42} &= \frac{e_{2z}L_1 L_2^2}{2LEI_z} + \frac{(e_{2z} - e_{3z})(L_1^3 L_2 - L_1 L_2^3)}{3L^2 EI_z} \\
f_{33} &= \frac{L_1^2 L_2^2}{3LEI_y} \\
f_{34} = f_{43} &= 0 \\
f_{44} &= \frac{L_1^2 L_2^2}{3LEI_z}
\end{aligned}$$

Invertiendo esta matriz tendremos la matriz de rigidez de la barra:

$$\begin{bmatrix} N_1 \\ N_2 \\ P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \\ k_{31} & k_{32} & k_{33} & k_{34} \\ k_{41} & k_{42} & k_{43} & k_{44} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix}$$

La matriz de compatibilidad seguirá siendo la misma, pues no se cambia la orientación de los ejes locales.

$$\begin{bmatrix} u_1 \\ u_2 \\ v \\ w \end{bmatrix} = \begin{bmatrix} -\cos \alpha_1 & -\cos \beta_1 & -\cos \gamma_1 & \cos \alpha_1 & \cos \beta_1 & \cos \gamma_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\cos \alpha_1 & -\cos \beta_1 & -\cos \gamma_1 & \cos \alpha_1 & \cos \beta_1 & \cos \gamma_1 \\ -\frac{L_2 \cdot \cos \alpha_2}{L} & -\frac{L_2 \cdot \cos \beta_2}{L} & -\frac{L_2 \cdot \cos \gamma_2}{L} & \cos \alpha_2 & \cos \beta_2 & \cos \gamma_2 & -\frac{L_1 \cdot \cos \alpha_2}{L} & -\frac{L_1 \cdot \cos \beta_2}{L} & -\frac{L_1 \cdot \cos \gamma_2}{L} \\ -\frac{L_2 \cdot \cos \alpha_3}{L} & -\frac{L_2 \cdot \cos \beta_3}{L} & -\frac{L_2 \cdot \cos \gamma_3}{L} & \cos \alpha_3 & \cos \beta_3 & \cos \gamma_3 & -\frac{L_1 \cdot \cos \alpha_3}{L} & -\frac{L_1 \cdot \cos \beta_3}{L} & -\frac{L_1 \cdot \cos \gamma_3}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ x_2 \\ y_2 \\ z_2 \\ x_3 \\ y_3 \\ z_3 \end{bmatrix}$$

La aplicación de estas nuevas ecuaciones nos revela que el efecto de la excentricidad de los nudos no es especialmente relevante, aun sin llegar a ser despreciable.

## CONSIDERACIÓN DE BARRAS FLEXIONADAS INICIALMENTE.

En ciertos casos, debido al empaquetamiento de las barras, o porque la geometría inicial no es totalmente compatible, es preciso que en el comienzo se considere la existencia de algunas barras ya flexadas, y por tanto, ejerciendo esfuerzos sobre la estructura. Si según la geometría recta de las barras, dos nudos  $i, j$  centrales que tienen que hallarse juntos, están a una distancia que definiremos como vector  $d$ , sus componentes serán:

$$\begin{aligned}
d_x &= d \cdot \cos \alpha \\
d_y &= d \cdot \cos \beta \\
d_z &= d \cdot \cos \gamma
\end{aligned}$$

Si se fuerza a los nudos  $i, j$  a juntarse, aparecerán entre ellos unas fuerzas iguales y contrarias  $F$ , que podremos determinar usando la matriz de rigidez global de la estructura. Veamos el ejemplo para la componente  $x$ , o sea,  $F_x$ .

$$\begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & S_{ii} & \dots & S_{ij} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & S_{ji} & \dots & S_{jj} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \\ x_i \\ \vdots \\ x_j \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ \vdots \\ P_i - F_x \\ \vdots \\ P_j + F_x \\ \vdots \end{bmatrix}$$

Sumando ambas filas para eliminar las fuerzas desconocidas  $F_x$  y sustituimos la segunda fila por la ecuación:

$$x_i - x_j = d \cdot \cos \alpha$$

Para conservar la simetría de la matriz de rigidez es preciso que la columna  $j$  tenga todos sus términos nulos, excepto el  $ij$ , que será igual a  $-1$ , y el  $jj$  que será igual a  $1$ .

$$\begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & S_{ii} + 2S_{ij} + S_{jj} + 1 & \dots & 1 & \dots \\ \vdots & \vdots & \vdots & 0 & \vdots \\ 0 & 1 & 0 & -1 & 0 \\ \vdots & \vdots & \vdots & 0 & \vdots \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \\ x_i \\ \vdots \\ x_j \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ P_i + P_j + (S_{ij} + S_{jj} - 1) \cdot d \cdot \cos \alpha \\ \vdots \\ d \cdot \cos \alpha \\ \vdots \end{bmatrix}$$

El resto de términos de la fila  $i$  será:  $S'_{in} = S_{in} + S_{jn}$

El resto de términos de la columna  $i$  será:  $S'_{ni} = S_{ni} + S_{nj}$

Por último, el resto de términos del vector de cargas será:  $P'_n = P_n - S_{nj} \cdot d \cdot \cos \alpha$

## GRANDES DEFORMACIONES.

Para grandes luces, las deformaciones suelen ser importantes, y con la variación de la geometría, los esfuerzos obtenidos en un cálculo lineal pueden diferir significativamente de los reales.

La solución más sencilla en estos casos es un cálculo iterativo hasta que el proceso converja a un resultado determinado. Conviene en este caso definir un parámetro que nos permita fijar el grado de precisión deseado. Normalmente será la comparación de valores entre dos iteraciones consecutivas, tales como desplazamientos, o incluso las funciones de estabilidad.

## PROCESO DE CÁLCULO.

Para considerar la casuística anterior resulta por tanto preciso un método iterativo, que puede basarse en los siguientes pasos:

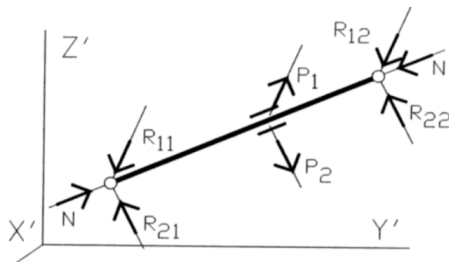
- a.) Cálculo de la estructura suponiéndola lineal, y por tanto las funciones de estabilidad igual a la unidad.
- b.) Con los esfuerzos obtenidos se calculan las funciones de estabilidad. De igual modo se procede a un redimensionado de aquellas barras que no resistan los esfuerzos iniciales.

- c.) Se calculan las funciones de estabilidad y se repite el cálculo hasta que la diferencia de las funciones de estabilidad en dos cálculos consecutivos sea inferior a un límite prefijado (usualmente un 0,1%).
- d.) Una vez que el proceso converge, se calculan las fuerzas desequilibradas en la posición deformada. Con estas fuerzas y con la estructura deformada se calcula nuevamente la misma según el proceso descrito, y los resultados se acumulan a los anteriores. Normalmente las fuerzas desequilibradas son pequeñas, y no es preciso proceder por escalones de carga como sería lo habitual en estos casos.
- e.) Se reitera el proceso hasta que las fuerzas desequilibradas sean inferiores a un límite prefijado.

## OTRAS SOLUCIONES CONSTRUCTIVAS

Otras posibilidades que se estudian en este campo son las barras curvas o el uso de una articulación central deslizante.

Ambas tratan de dar solución al problema de la incompatibilidad geométrica de estas estructuras. Con el nudo central deslizante se dota de un grado más de libertad a la barra, con lo que al ser más mecanismo es más fácil que la malla se adapte a posiciones intermedias sin tener que flexionar en exceso las barras.



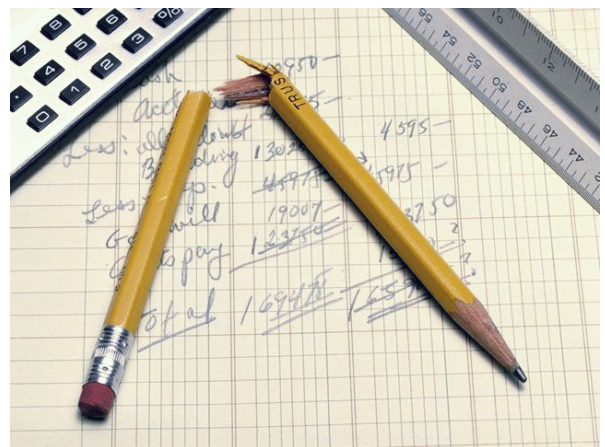
Con las barras curvas se quiere dar solución al problema de que como las barras tendrán un espesor determinado, es imposible para muchas configuraciones que las barras vayan rectas de un extremo a otro, lo que obligaría a una barra a atravesar a otras. Este planteamiento ya lo vemos en las estructuras y patentes de E. P. Piñero.

En cualquier caso estas soluciones son muy específicas y van más allá de los objetivos de este trabajo, por lo que para un estudio detallado de las mismas recomendamos la referencia [4].

## REFERENCIAS

- [1] Escrig, F. & Valcarcel, J. P.  
"Introducción a la geometría de Estructuras Desplegables de Barras"  
Boletín Académico, nº 3. ETSA Coruña. 1986.
- [2] Valcarcel, J.P.  
"Cúpulas de grandes luces con Módulos de aspas"  
I Encuentro Internacional Estructuras ligeras para grandes luces. Fundación Emilio Pérez Piñero.  
Murcia. 1992.
- [3] Shan, W.  
"Computer analysis of foldable Structures"  
Computer and Structures Vol. 42 nº6. 1992
- [4] Valcarcel, J.P.  
"Cálculo de Estructuras Desplegables de Barras"  
Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993.
- [5] Valcarcel, J. P. & Escrig, F.  
"Recent advances in the analysis of expandable structures"  
Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ.  
Southampton 1996.

## 2. Metodología





## 2.1 OBJETIVOS

### CONSIDERACIONES PREVIAS

Podemos ir fijando los objetivos de este estudio la división por fases expuesta por P. Piñero en el punto anterior, y comentando la aproximación que hacemos a cada una de ellas:

#### 1.- Determinación de la forma general.

En este trabajo nos basaremos en mallas desplegables en forma de cúpula apoyadas sobre sus esquinas o vértices de su polígono base, por ser una tipología interesante al ser en general bastante estables, a la vez que permite prescindir de elementos auxiliares como pilares, resolviendo la propia malla toda la estructura. Trabajaremos con modelos que tengan una relación flecha-luz del orden de 1/2 o menor para que la malla proyectada no se deforme demasiado y no tengamos que recurrir a las mallas geodésicas.

#### 2.- Determinación de la retícula, disposición y longitud de las barras.

La retícula que empleamos es de módulo cuadrado o triangular con barras cruzadas en forma de aspa y articulación intermedia, con lo que trataremos con una tipología sencilla, a la vez que bastante probada.

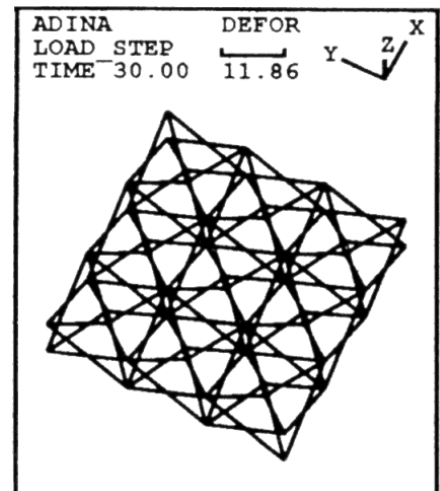
#### 3.- Cálculo mecánico y dimensionamiento de las barras.

En el apartado de cálculo de las barras, el cálculo estático ya está plenamente resuelto como vemos en numerosos artículos de P. Valcarcel [1] y W. Shan [2].

Respecto al cálculo dinámico. Charis Gantes empieza sus estudios con el programa ADINA, que simulan el comportamiento de la estructura real, y estudia el despliegue a través del método de Newton-Raphson [3]. Usa elementos viga isoparamétricos de 2 nodos y un algoritmo para un automático incremento de cargas para conseguir el completo despliegue y simula la fricción de los nudos usando muelles no lineales traslacionales que representan el momento de fricción que se produce cuando hay una rotación relativa entre las dos barras que forman la 'tijera'.

Vemos que en realidad se trata de un tipo de cálculo quasi-dinámico, en el que el proceso de despliegue es lento y el modelo de elementos finitos consume una gran potencia de cálculo. En el ejemplo, el estudio de una malla plana de 3x3 módulos de forma cuadrada, más de una hora de cálculo en un supercomputador CRAY-2.

Es por tanto preciso otro tipo de enfoque que simplifique el cálculo numérico y haga factible el cálculo dinámico de estructuras más o menos grandes con los ordenadores personales actuales.



4.- Resolución constructiva de los nudos.

5.- Ejecución y montaje en su emplazamiento.

Estos dos últimos puntos, aunque básicos en el estudio de las despleables (tengamos en cuenta que es en base al último punto que Piñero define la diferencia sustancial entre las despleables y los otros tipos de mallas espaciales) se centran más en los aspectos constructivos, alejándonos de nuestro objeto de estudio en el presente trabajo.

Nuestros objetivos se centran por tanto en la fase de cálculo, y más concretamente en el cálculo dinámico de estas estructuras. Un objetivo secundario nace de la necesidad de disponer de modelos de estructuras sobre los cuales poder aplicar el cálculo dinámico, es por ello que la definición o generación de la retícula también se plantea, aunque como un objetivo secundario.

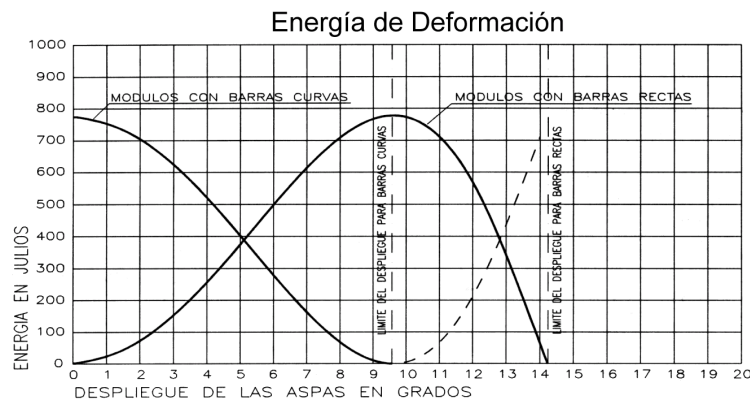
---

## 2.1.1 CÁLCULO DINÁMICO

---

### DESPLIEGUE

Un de despliegue automático de las mallas, más o menos rápido, es interesante para arquitectura de emergencia o para aquellas mallas que se diseñan como autodesplegables, en las que no es sencillo definir a priori cual será la velocidad de despliegue. Este tipo de estructuras ya fueron planteadas por P. Piñero usando muelles, y por P. Valcárcel usando barras dobladas [4].

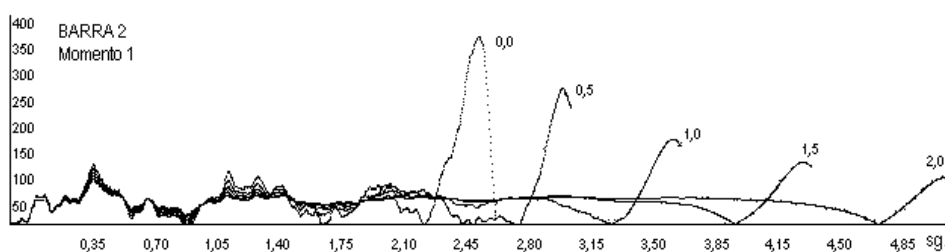


Este gráfico nos muestra que con un diseño adecuado de las barras se puede invertir la tendencia según la cual para efectuar el despliegue de una malla es preciso aplicar energía sobre la misma. Con los módulos de barras curvas, el máximo de su energía elástica, se da cuando se halla totalmente plegada, por lo que es preciso aplicar energía exterior para llegar al mismo, pero a cambio el despliegue sería automático, bastando para ello con liberar al paquete de barras de las sujeciones que se le hallan dispuesto.

Como decíamos anteriormente los métodos de cálculo actuales no permiten, por lo costoso del proceso y lo complejo de las herramientas disponibles, un cálculo dinámico asequible con los ordenadores personales disponibles hoy en día, y mucho menos este cálculo cuando el proceso de despliegue es realmente rápido y los fenómenos inerciales y de amortiguamiento son importantes.

La línea de investigación de este trabajo se centra por tanto en este aspecto, estudiar las mallas que despliegan con una velocidad significativa, ya sea por su energía de deformación propia o porque se le suministra desde el exterior, y comprobar que los fenómenos derivados de las aceleraciones que sufre, originan esfuerzos en las barras que deben ser significativamente distintos de aquellos que se obtienen por medio de un cálculo estático o cuasi-estático.

Estudiaremos asimismo la influencia del amortiguamiento sobre los esfuerzos que soporta la estructura, y comprobaremos que su existencia es básica para que los esfuerzos no se disparen y lleguen a tomar valores excesivos [5].



Por tanto el amortiguamiento se plantea como otra variable de estudio que podría regularse según un diseño constructivo adecuado de los nudos, de modo que debe tener un valor mínimo, pero no debe ser tan alto que dificulte o impida el despliegue de la estructura.



## ANÁLISIS DE RESULTADOS.

Complementariamente a este estudio del despliegue era preciso el manejo de una ingente cantidad de datos que arroja el cálculo dinámico. Para poder sacar algo en limpio entre tantos datos ha sido preciso desarrollar unas herramientas complementarias al cálculo en sí que permitan trabajar cómodamente con los resultados obtenidos.

Un primer análisis es la visión del proceso de despliegue, con datos que nos permitirán identificar en cada momento cuales son las barras más solicitadas de la estructura, y los valores de los esfuerzos alcanzados. La visión misma del proceso de despliegue es interesante pues permite determinar el comportamiento general de la estructura, e incluso determinar el grado de incompatibilidad de cada parte del proceso en función de la variación de la velocidad de despliegue.

Una vez conocidos los elementos más solicitados, es necesario un siguiente análisis, particularizado para cada barra o nudo de la estructura que deseemos estudiar.

Los análisis que efectuaremos serán la visualización de la ecuación de onda o en el dominio del tiempo, y la visualización menos habitual, pero que a veces reporta más información, de su visualización en el dominio de las frecuencias, mediante la aplicación de una transformada de Fourier.

Esta última también la podemos aprovechar para realizar un estudio de vibraciones en mallas. Podremos de esta forma aplicar este estudio a mallas desplegables ya fijas en su posición final como a mallas no desplegables. Una vez realizado el cálculo de una de estas malla, mediante la aplicación de una Transformada Discreta de Fourier, obtenemos rápidamente el valor de las frecuencias a las que es más sensible la estructura, de modo que si va a hallarse sometida a

acciones dinámicas, será fácil evaluar el impacto de las mismas sobre nuestra estructura aplicando una función de transferencia o estudio similar.

---

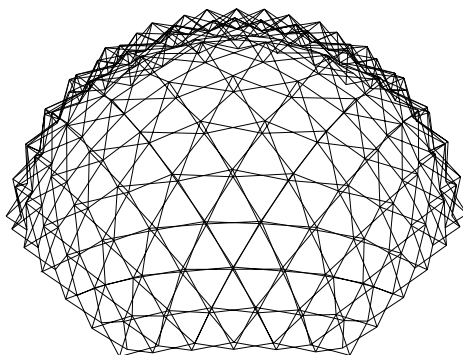
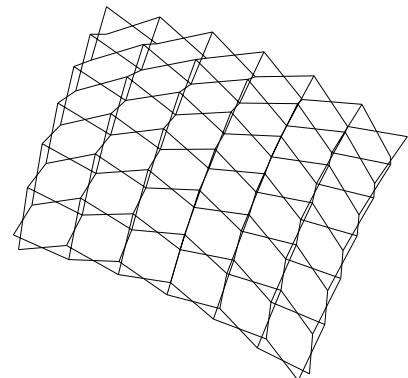
## 2.1.2 GENERACIÓN DE MALLAS.

---

A mayores de nuestro objetivo principal, comentado anteriormente, hemos desarrollaremos unos procesos propios de generación de mallas desplegables, saliéndonos de los caminos habituales, que nos permiten generar de forma rápida y precisa los modelos informáticos de cálculo de las estructuras. Estos modelos los hemos utilizado en la mayoría de los estudios realizados, y se ha comprobando un correcto funcionamiento de los mismos, verificandose por tanto la validez de los métodos de generación empleados.

Para el estudio dinámico es interesante trabajar con dos modelos de estructuras desplegables totalmente diferenciados. Básicamente nos interesaba comparar el comportamiento de mallas compatibles con el de otras con incompatibilidades manifiestas. Es por ello que desarrollaremos el proceso de generación de estas dos tipologías.

Para ambas nos basamos en el esquema de aspas, no en el de haces, que constructivamente resulta más sencillo. Para la primera optamos por una malla de módulos cuadrados, por ser la que presenta menor problema de incompatibilidades, y como en este tipo de mallas disponemos de bastante libertad de maniobra, pudimos añadirle nosotros a mayores, la condición de que todas las barras fuesen iguales.



El segundo tipo elegido fue el de módulos triangulares, nos basamos en el sistema de proyección de una malla plana sobre la esfera, pero con alguna particularidad en el proceso que consigue una posición desplegada altamente estable, merced de pasar por un estado de despliegue con alto grado de incompatibilidades.

En los capítulos siguientes veremos como se materializan estos dos objetivos principales.

## REFERENCIAS:

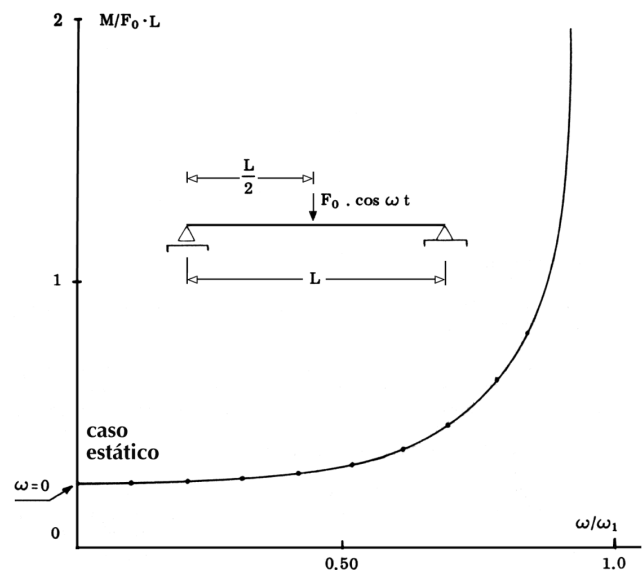
- [1] Escrig, F. & Valcarcel, J. P.  
"Introducción a la geometría de Estructuras Desplegables de Barras"  
Boletín Académico, nº 3. ETSA Coruña. 1986. ISSN 0213-3474 pp.48-57
- [2] Shan, W.  
"Computer analysis of foldable Structures"  
Computer and Structures Vol. 42 nº6. 1992.
- [3] Gantes, C.; Connor, J.J. & Logcher, R.D.  
"Combining Numerical Analysis and Engineering Judgement to Design Deployable Structures"  
Computer & Structures. 1991.
- [4] Valcarcel, J. P.  
"Cálculo de Estructuras Desplegables de Barras"  
Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993.
- [5] Lopez, E. & Muñoz, M.  
Comparative Study of the Geometric Incompatibilities in the Dinamica of Expandable Structures  
IASS-IACM 2000. Fourth International Colloquium on Computation of Shell & Spatial Structures  
June 2000. Chania-Crete, Greece.

## 2.2 METODOLOGÍA

### 2.2.1 ANÁLISIS DINÁMICO

#### INTRODUCCIÓN.

Como introducción a la importancia de un estudio dinámico de las estructuras, baste el gráfico siguiente [1], que muestra el caso de una viga sometida a una carga cíclica. Se ha graficado el valor de los momentos máximos (amplitud) en el centro de la viga ( $M$ ) en función de la relación entre la frecuencia de la carga ( $\omega$ ) y la frecuencia propia fundamental de la viga ( $\omega_1$ ).



Del gráfico se observa que para  $\omega=0$  (caso estático) el valor del momento es mínimo. Para  $\omega$ , de 0 a 0,5 el resultado no varía significativamente, pero es evidente que a partir de este valor, los efectos inerciales no pueden ser ignorados.

Por ello el análisis de estructuras clásicamente se divide en dos campos según el tipo de acciones que se ejerce sobre ellas.

Si la totalidad de las acciones a que la estructura se ve sometida son constantes o tienen una variación muy lenta a lo largo del tiempo, el estudio podrá limitarse a estudiar un instante determinado, y lógicamente elegiremos el más desfavorable.

Si las acciones varían con cierta rapidez, debemos tener en cuenta los efectos inerciales, en cuyo caso el estudio se extenderá en un período de tiempo suficientemente largo como para que la respuesta de la estructura pueda ser conocida de manera adecuada.

Si bien esta bipartición clásica es correcta para las estructuras en general, para nuestro caso concreto de estudio que son las mallas desplegadas, no es totalmente correcta, pues en nuestro

caso aun siendo la intensidad y dirección de las cargas constante en el tiempo, los efectos inerciales serán o no despreciables atendiendo no a las cargas, sino a la velocidad de despliegue.

Por tanto deberemos cambiar el concepto de variación de las cargas más o menos rápida en el tiempo, por el de variación de la geometría más o menos rápida en el tiempo.

De esta forma si la variación de la geometría es muy lenta, podremos realizar un análisis estático en la posición más desfavorable, o mejor en todas las posiciones intermedias y quedarnos con los esfuerzos más desfavorables en el dimensionado de las barras.

Si la variación de la geometría es relativamente rápida, los efectos inerciales será considerables, y es en este campo donde centraremos nuestros estudios.

En realidad, durante el proceso de despliegue no tenemos una estructura propiamente dicha, sino que se trata de un mecanismo por lo que los cálculos estáticos tienen que falsear este estado simulando cuanto menos una estructura isostática, y para ello lo más habitual es coaccionar alguno de los desplazamientos de los nudos de la malla.

Si eligen correctamente los nudos, las reacciones que puedan producirse en ellos serán relativamente pequeñas y el resultado bastante correcto, pero si las reacciones que obtenemos son relativamente importantes, significa que estamos introduciendo en la estructura unas fuerzas que desviarán los resultados de todas las barras de la estructura de modo favorable o desfavorable.

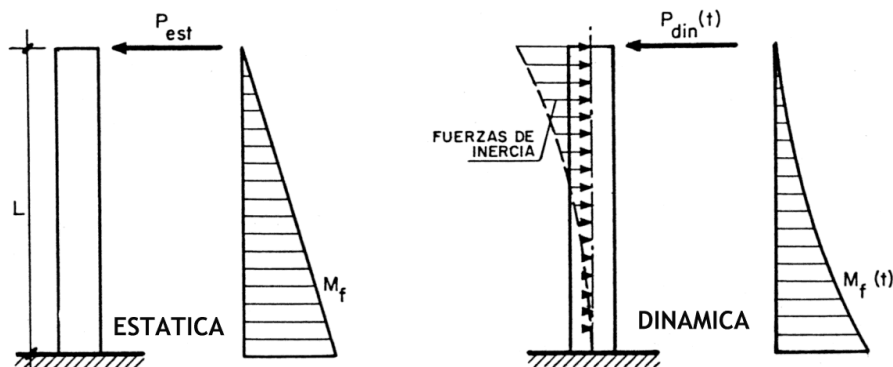
Lo correcto en este caso es elegir aquellos nodos que efectivamente, durante el proceso de despliegue, se sujeten mecánicamente para controlar la velocidad de apertura, y por tanto deberemos conocer perfectamente la ejecución de este proceso.

Realizando el estudio dinámico que proponemos no se nos producirá este problema, a la vez que podremos plantear la ejecución de estructuras de despliegue verdaderamente rápido.

A continuación desarrollaremos un resumen de los tipos básicos de análisis que podemos efectuar sobre una estructura y como se hallan relacionados entre sí. Efectuaremos una visión un poco general para dejar claros los conceptos que manejaremos más adelante, a la vez que nos servirá para dejar claro el por qué del tipo de estudio que finalmente optamos por realizar.

## EQUILIBRIO ESTÁTICO Y EQUILIBRIO DINÁMICO.

La naturaleza propia del problema dinámico ya la podemos poner de manifiesto en las misma ecuaciones de planteamiento de equilibrio del sólido.



Si en la estática cada porción de nuestro sólido debía cumplir:

$$\Sigma F = 0$$

$$\Sigma M = 0$$

ahora debemos recordar la segunda ley de Newton  $F = m \cdot a$ , por tanto

$$\Sigma F = m \cdot a$$

$$\Sigma M = I_0 \cdot \alpha$$

Lo que es lo mismo que considerar que  $m \cdot a$  (ó  $I_0 \cdot \alpha$ , para la rotación) existen como fuerzas inerciales ( $I$ ) y por tanto el sumatorio de fuerzas puede quedar de forma genérica:

$$\Sigma F + \Sigma I = 0$$

Que es el conocido principio de D'Alembert. Por supuesto ahora el resultado de las expresiones serán ecuaciones diferenciales en lugar de las ecuaciones algebraicas que resultan en el caso estático.

## ANÁLISIS ESTÁTICO

El análisis estático ya lo hemos comentado anteriormente, así que aquí expondremos únicamente el esquema matricial resultante que nos sirva de comparativa con las ecuaciones que iremos obteniendo en el cálculo dinámico.

El cálculo estático de una estructura involucra la solución del sistema de ecuaciones lineares representadas por :

$$K = u \cdot r$$

donde  $K$  es la matriz de rigidez,  $r$  es el vector de cargas aplicadas, y  $u$  es el vector de desplazamientos resultantes.

## ANÁLISIS ARMÓNICO ESTACIONARIO

Un tipo muy común de cargas es de la forma  $r(t) = p \cdot \cos(\omega t)$ , donde  $\omega$  es la frecuencia circular de la excitación, así  $r$  varía con respecto al tiempo. Sin embargo, la situación en el espacio de carga  $p$  no varía como una función de tiempo. Para el caso de un sistema no amortiguado, la ecuación de equilibrio del sistema estructural es de la forma:

$$K \cdot u(t) + M \cdot \ddot{u}(t) = r(t) = p \cdot \cos(\omega t)$$

donde  $K$  es la matriz de rigidez y  $M$  es la matriz de masa diagonal. La solución de esta ecuación requiere que los desplazamientos  $u$  y aceleraciones  $\ddot{u}$  sea de la forma siguiente:

$$u(t) = a \cdot \cos(\omega t)$$

$$\ddot{u}(t) = -\omega^2 \cdot a \cdot \cos(\omega t)$$

Por tanto, la amplitud de respuesta  $a$  se obtiene de la solución de las siguientes ecuaciones lineales:

$$[K - \omega^2 \cdot M] \cdot a = p$$

Podemos resaltar que la solución para cargas estáticas es simplemente una solución de esta ecuación con frecuencia cero:  $K \cdot a = p$

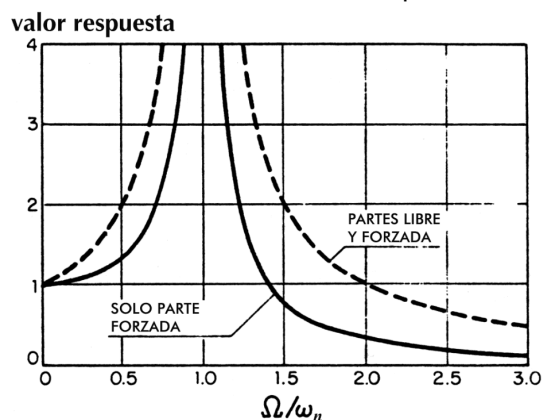
Los desplazamientos  $a$  que obtenemos de la resolución del sistema, y las tensiones derivadas de los mismos son valores también máximos, los cuales varía como  $\cos(\omega t)$ .

Las limitaciones del análisis armónicos estables son:

- El amortiguamiento estructural se asume que es cero.
- La estructura puede ser analizada para cualquier número de distribución espacial cargas. Sin embargo, la frecuencia de excitación de todas las cargas debe ser la mismo.



- Si la frecuencia de excitación corresponde a una frecuencia natural de la estructura, el sistema entrará en resonancia, y por tanto resultará en una respuesta infinita. En tal caso la solución del sistema fallará pues la matriz  $[ \mathbf{K} - \omega^2 \cdot \mathbf{M} ]$  será singular.



## ANÁLISIS MODAL.

Este análisis determina las formas de vibración libres de la estructura y sus frecuencias. Estas formas de vibración o **Modos Propios** de la estructura representan cómo vibra la estructura (autovectores) y con qué frecuencias (autovectores asociados). Estos modos naturales proveen una percepción excelente del comportamiento de la estructura.

Luego podrán usarse como base para un análisis posterior mediante el espectro de respuesta, que comentamos más adelante, sin embargo los vectores de Ritz son más recomendables para este último propósito.

El análisis de autovectores involucra la solución del problema generalizado de autovalores:

$$\mathbf{K} \cdot \mathbf{u} = \omega^2 \cdot \mathbf{M} \cdot \mathbf{a}$$

de donde deriva una ecuación similar a la del caso anterior, pero con la matriz de cargas igual a cero:

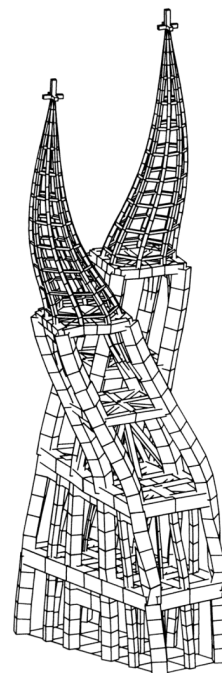
$$[ \mathbf{K} - \omega^2 \cdot \mathbf{M} ] \cdot \mathbf{a} = \mathbf{0}$$

donde  $\mathbf{K}$  es la matriz de rigidez,  $\mathbf{M}$  es la matriz de masa diagonal,  $\omega^2$  es la matriz diagonal de autovalores (que son el cuadrado de las frecuencias circulares  $\omega$ , para cada modo), y  $\mathbf{a}$  es la matriz de los correspondientes autovectores (formas modales).

$$\omega^2 = \begin{bmatrix} \omega_1^2 & 0 & \dots & 0 \\ 0 & \omega_2^2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \omega_n^2 \end{bmatrix}$$

Cada pareja de autovalor-autovector se llama **Modo de Vibración natural de la estructura**. La frecuencia cíclica  $f$ , y el período  $T$ , del Modo están relacionados por:

$$T = 1 / f \quad \text{y} \quad f = \omega / 2\pi$$



El número de modos lógicamente estará limitado por el número de grados de libertad de las masas del modelo. Si el modelo fuese continuo, tendríamos infinitos grados de libertad.

Una estructura estable tendrá todas frecuencias naturales positivas. Los modos de frecuencia más bajos son usualmente los de mayor interés. Una estructura que es inestable, cuando se halla descargada tendrá algunos modos con frecuencia cero. Estos modos podrían corresponder al movimiento como sólido rígido de una estructura inadecuadamente apoyada, o a mecanismos que estén presentes dentro la estructura. No es posible un cálculo estático de tal estructura, sin embargo, los modos propios sí pueden ser hallados.

## ANÁLISIS DE VECTORES DE RITZ

La investigación nos ha mostrado que los modos normales o naturales de vibración no son el mejor fundamento para un análisis por superposición de modos de sujeto de estructuras a cargas dinámicas. Ha sido demostrado (Wilson, Yuan, y Dickens, 1982) que el análisis basado en un conjunto especial de vectores de Ritz, dependientes de las cargas, consigue resultados más precisos que el uso del mismo número de modos naturales.

La razón del excelente resultado de los vectores de Ritz es que estos están generados teniendo en cuenta la distribución espacial de las cargas dinámicas. Información esta muy importante que el uso directo de los modos naturales desprecia.

La distribución espacial del vector de carga dinámica sirve como vector de partida para iniciar el procedimiento. El primer vector Ritz es el vector de desplazamientos estáticos correspondiente al vector de carga inicial. Los vectores restantes están generados por un proceso recursivo en cual la matriz de masa es multiplicada por el vector Ritz previamente obtenido y el resultado usado como el vector de carga para la próxima solución. Cada solución estática se llama **ciclo de generación**.

Las técnicas estándares de resolución de problemas de autovectores sirven para resolver el problema y obtener un conjunto de modos de vectores Ritz. Cada modo de vector Ritz consiste en una forma modal y una frecuencia. El juego completo de los modos de vectores Ritz pueden ser usado como un base para representar el desplazamiento dinámico de la estructura. Una vez la matriz de rigidez es triangularizada solamente es necesario resolver estáticamente un vector de carga por cada vector Ritz requerido. Así resulta un algoritmo extremadamente eficiente.

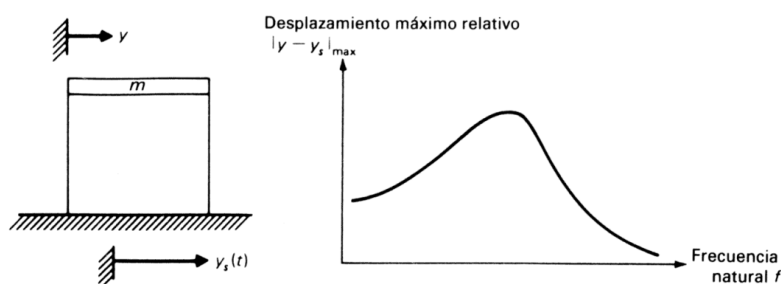
## ANÁLISIS DEL ESPECTRO DE RESPUESTA

En los últimos años este análisis ha tenido una amplia aceptación en la práctica de la dinámica estructural, pero más bien en el campo del diseño antisísmico.

Básicamente el espectro de respuesta en un diagrama de la máxima respuesta (desplazamiento, velocidad, aceleración u otra) a una función específica de excitación, para todos los sistemas posible con un grado de libertad.

Un choque representa la acción de una fuerza sobre un sistema. El valor máximo de la respuesta es una buena medida de la severidad del choque, y será dependiente de las características del sistema.

La abcisa del espectro es la frecuencia natural del sistema y la ordenada la respuesta máxima.



Tipos diferentes de excitación de choque, conducirán a diferentes espectros de respuesta, pero como el espectro de respuesta está determinado por un punto singular en la curva de respuesta temporal que es a su vez una porción incompleta de la información, puede ocurrir que dos excitaciones de choque diferentes tengan espectros de respuesta similares. A pesar de ello, el espectro de respuesta es un concepto útil, ampliamente utilizado.

Las ecuaciones de equilibrio dinámico asociadas con la respuesta de una estructura al movimiento del suelo están dadas por :

$$\mathbf{K} \cdot \mathbf{u}(t) + \mathbf{C} \cdot \dot{\mathbf{u}}(t) + \mathbf{M} \cdot \ddot{\mathbf{u}}(t) = m_x \cdot \ddot{u}_{gx}(t) + m_y \cdot \ddot{u}_{gy}(t) + m_z \cdot \ddot{u}_{gz}(t)$$

donde  $\mathbf{K}$  es la matriz de rigidez;  $\mathbf{C}$  es la matriz de amortiguamiento proporcional;  $\mathbf{M}$  es la matriz de masa diagonal;  $\mathbf{u}$ ,  $\dot{\mathbf{u}}$ , y  $\ddot{\mathbf{u}}$  son los desplazamientos relativos, velocidades y aceleraciones con respecto a la suelo;  $m_x$ ,  $m_y$  y  $m_z$  son las cargas por la aceleración; y  $\ddot{u}_{gx}$ ,  $\ddot{u}_{gy}$  y  $\ddot{u}_{gz}$  son los componentes de aceleración del suelo.

El análisis de espectro de Respuesta busca la respuesta probable máxima a estas ecuaciones en vez de la historia de tiempo completa. La aceleración del suelo de un terremoto en cada dirección se da como una curva de espectro de respuesta de la aceleración. Sin embargo aunque las aceleraciones se especifiquen en tres direcciones, solamente se da un resultado positivo por respuesta.

## ANÁLISIS EN EL TIEMPO.

El análisis en el tiempo es usado para determinar la respuesta dinámica de una estructura sometida un estado de cargas arbitrario. El sistema de ecuaciones que se deben resolver para que se produzca el equilibrio dinámico es :

$$\mathbf{K} \cdot \mathbf{u}(t) + \mathbf{C} \cdot \dot{\mathbf{u}}(t) + \mathbf{M} \cdot \ddot{\mathbf{u}}(t) = \mathbf{r}(t)$$

donde  $\mathbf{K}$  es la matriz de rigidez ;  $\mathbf{C}$  es la matriz de amortiguamiento proporcional;  $\mathbf{M}$  es la matriz de masa diagonal;  $\mathbf{u}$ ,  $\dot{\mathbf{u}}$ , y  $\ddot{\mathbf{u}}$  son los desplazamientos, velocidades y aceleraciones y  $\mathbf{r}$  es la carga aplicada.

Para este tipo de problemas suele requerir un proceso iterativo en el tiempo. La precisión del resultados suele depender del tamaño del incremento de tiempo  $dt$ , y normalmente mejorarán al disminuir  $dt$ .

## ANÁLISIS NO LINEAL EN EL TIEMPO

El sistema de ecuaciones que nos garantizan el equilibrio dinámico de una estructura lineal que contiene algún elementos no lineal, y sometida un conjunto de cargas arbitrarias se puede escribir como :

$$\mathbf{K}_L \cdot \mathbf{u}(t) + \mathbf{C} \cdot \dot{\mathbf{u}}(t) + \mathbf{M} \cdot \ddot{\mathbf{u}}(t) + \mathbf{r}_N(t) = \mathbf{r}(t)$$

donde  $\mathbf{K}_L$  es la matriz rigidez para los elementos lineales ;  $\mathbf{C}$  es la matriz de amortiguamiento proporcional;  $\mathbf{M}$  es la matriz de masa diagonal ;  $\mathbf{r}_N$  es el vector de fuerzas de los elementos no lineales;  $\mathbf{u}$ ,  $\dot{\mathbf{u}}$ , y  $\ddot{\mathbf{u}}$  son los desplazamientos, velocidades, y aceleraciones, y  $\mathbf{r}$  es el vector de cargas aplicadas.

Esta ecuaciones debe ser resueltas iterativamente en el tiempo. Se puede asumir que la parte derecha de la ecuación varía linealmente durante cada paso de tiempo paso, y resolver por integración exacta esta ecuación en cada iteración. Las iteraciones se repiten en cada intervalo de tiempo hasta que la solución converge. Si no converge, se puede dividir el intervalo de tiempo en subpasos más pequeños y probar otra vez.

Como vemos de todas las posibilidades que tenemos abiertas, la alta no linealidad geométrica del proceso de despliegue nos obliga a recurrir a los procesos de análisis en el tiempo. Con la estructura una vez desplegada y estabilizada podríamos plantear un análisis modal de la misma, pero no es este nuestro objetivo. En cualquier caso veremos que con los resultados de nuestro análisis también podemos hallar las frecuencias fundamentales de vibración de la estructura.

A continuación pasaremos a detallar el modelo matemático elegido y su implementación en el ordenador.

---

## 2.2.2 MODELO MATEMÁTICO

---

### ■ ECUACIONES DE COMPORTAMIENTO

#### SISTEMA DE ECUACIONES DIFERENCIALES

Hemos visto que las ecuaciones que vamos a obtener son del tipo:

$$K \cdot u(t) + C \cdot \dot{u}(t) + M \cdot \ddot{u}(t) = r(t)$$

ecuación que también podemos expresar como.

$$k \cdot x + c \cdot \frac{dx}{dt} + m \cdot \frac{d^2x}{dt^2} = f$$

Vemos que tendremos que tratar con una ecuación diferencial de 2º orden. En algunos casos la ecuación que resulta puede ser integrada de forma analítica, pero para la magnitud del problema que vamos a tratar, este tipo de intento es a todas luces imposible.

Para cada nudo de la estructura tendremos cuanto menos (dependerá de la modelización elegida) tres ecuaciones como la anterior, una para cada dirección del espacio, por lo que vemos que el problema rápidamente se escapa a cualquier intento de tratamiento manual y debemos proceder a su implementación informática para que sea útil.

#### DIFERENCIAS FINITAS

Cuando realizamos el paso del sistema de ecuaciones diferenciales a los métodos numéricos, la variable continua  $t$  debe ser remplazada por una variable discreta  $t_i$ , y la ecuación diferencial se resuelve progresivamente en incrementos de tiempo  $h = \Delta t$ , arrancando de las condiciones iniciales conocidas. La solución es aproximada, pero con un incremento de tiempo lo suficientemente pequeño, se obtienen soluciones de precisión aceptable.

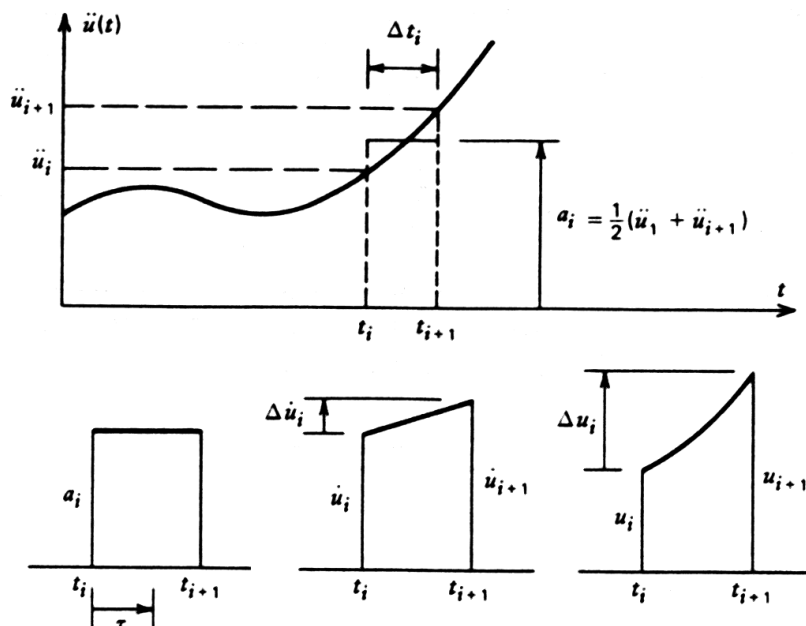
En la elección del método de integración son importantes dos aspectos: la estabilidad y la precisión. Decimos que un método es **estable incondicionalmente** si siempre da soluciones finitas con independencia del valor de  $\Delta t$  elegido. Un método es **condicionalmente estable** si da soluciones finitas sólo para pequeños valores de  $\Delta t$ . Pese a esta desventaja, los métodos condicionalmente estables son generalmente más precisos.

Otra clasificación de los algoritmos de integración es que pueden ser de paso simple o paso múltiple. Los de **paso simple** predicen las condiciones de un sistema en un tiempo  $t_{i+1}$  a partir de las condiciones existentes en el paso anterior  $t_i$ . Los de **paso múltiple** recurren a la situación varios pasos atrás, en los tiempos  $t_i, t_{i-1}, t_{i-2}$ , etc. Evidentemente los métodos multipaso son más precisos, pero no pueden arrancar el cálculo por sí solos, precisarán la realización de unos cálculos iniciales por un método de paso simple.

Hablaremos también de métodos **explícitos**, como aquellos que obtienen las soluciones en el tiempo  $t_{i+1}$  directamente, mientras que los **implícitos** las incógnitas aparecen en ambos lados de las ecuaciones, y se obtendrán sus valores resolviendo un sistema de ecuaciones lineales.

### Método de aceleración media.

A continuación exponemos el primer método que hemos incorporado en el programa de cálculo. Básicamente es el método de la aceleración media [2] (también descrito en algunos libros como método de Newmark de  $\beta=1/4$  [3], o método de la regla del trapecio, o método de la aceleración media constante), pero ligeramente modificado.



Para resolver la ecuación general aprovechamos en primer lugar el hecho de que **M** es diagonal con determinante distinto de cero de manera que se puede despejar la matriz de aceleración y expresarla en la forma:

$$\ddot{\mathbf{U}} = \mathbf{M}^{-1} \cdot [\mathbf{R} - \mathbf{C} - \mathbf{K} \cdot \mathbf{U}] \quad (1)$$

Para no estar condicionados por el comportamiento más o menos lineal de la estructura, la ecuación anterior se resuelve mediante un método iterativo con un esquema de integración en el tiempo de tipo implícito de dos pasos. En efecto, dadas las posiciones  $U_t$  y las velocidades  $\dot{U}_p$  de los nudos en un instante  $t$ , las posiciones y velocidades en un instante  $t+\Delta t$  se calculan según los pasos siguientes:

1º.- Cálculo de las posiciones  $U_p$ , velocidades  $\dot{U}_p$  provisionales en un instante  $t+\Delta t$  suponiendo que la aceleración de los nudos durante el intervalo  $\Delta t$  se mantiene constante e igual a la aceleración en el instante  $t$ .

$$\dot{U}_p^{t+\Delta t} = \dot{U}^t + \ddot{U}^t \cdot \Delta t \quad (2)$$

$$U_p^{t+\Delta t} = U^t + \left( \dot{U}^t + \dot{U}_p^{t+\Delta t} \right) \cdot \frac{\Delta t}{2} \quad (3)$$

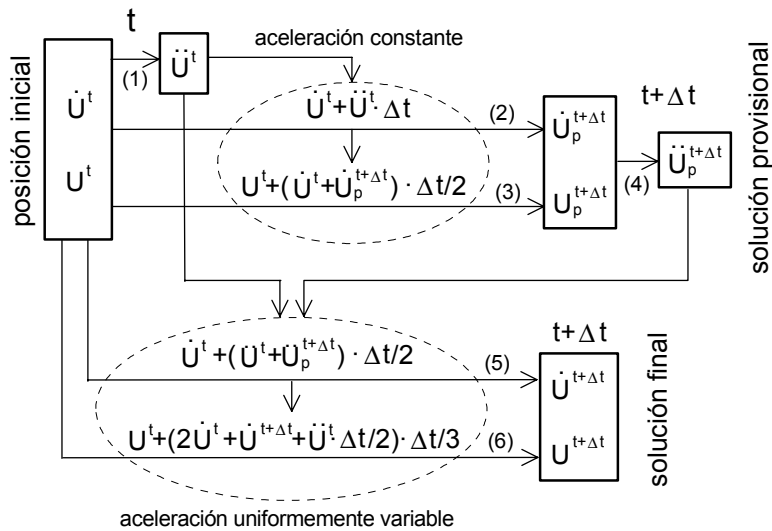
2º.- Cálculo de las posiciones y velocidades finales en el instante  $t+\Delta t$  suponiendo las aceleraciones uniformemente variables entre el valor en el instante  $t$  y el valor provisional en el instante  $t+\Delta t$  deducido de la ecuación (7) para las posiciones y velocidades provisionales (8) y (9).

$$\ddot{U}_p^{t+\Delta t} = \mathbf{M}^{-1} \cdot [\mathbf{R} - \mathbf{C} - \mathbf{K} \cdot U_p^{t+\Delta t}] \quad (4)$$

$$\dot{U}^{t+\Delta t} = \dot{U}^t + \left( \ddot{U}^t + \ddot{U}_p^{t+\Delta t} \right) \cdot \frac{\Delta t}{2} \quad (5)$$

$$U^{t+\Delta t} = U^t + \left( 2 \cdot \dot{U}^t + \dot{U}^{t+\Delta t} + \ddot{U}^t \cdot \frac{\Delta t}{2} \right) \cdot \frac{\Delta t}{3} \quad (6)$$

En el diagrama adjunto se representa el esquema de integración elegido



Si bien este algoritmo funciona, no era lo suficientemente preciso para nuestros propósitos, ya que presentaba el inconveniente de que exigía intervalos de tiempo muy pequeños o sino el proceso podía diverger muy rápidamente.

A continuación optamos por implementar el conocido método de Runge-Kutta.

### Método de Runge-Kutta de cuarto orden.

Los métodos de Runge-Kutta logran la exactitud del procedimiento de una serie de Taylor sin requerir el cálculo de derivadas superiores. Existen muchas variaciones, pero todas se pueden denotar en la forma generalizada de la ecuación:

$$y_{i+1} = y_i + \phi(y_i, t_i) \cdot \Delta t$$

donde  $\phi$  es conocida como **función incremento**, la cual puede interpretarse como una pendiente representativa sobre el intervalo. La función incremento se escribe por lo general como:

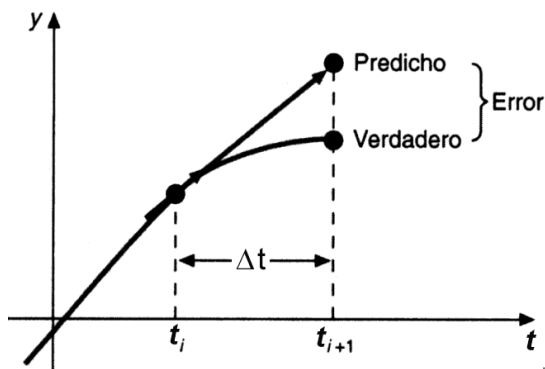
$$\phi = a_1 \cdot k_1 + a_2 \cdot k_2 + \dots + a_n \cdot k_n$$

siendo las **a** constantes y las **k** valores que se obtienen de:

$$\begin{aligned} k_1 &= f(t, y) \\ k_2 &= f(t + \theta_1 \cdot \Delta t, y + (w_{11} \cdot k_1) \cdot \Delta t) \\ k_3 &= f(t + \theta_2 \cdot \Delta t, y + (w_{21} \cdot k_1 + w_{22} \cdot k_2) \cdot \Delta t) \\ &\vdots \\ k_n &= f(t + \theta_{n-1} \cdot \Delta t, y + (w_{n-1,1} \cdot k_1 + w_{n-1,2} \cdot k_2 + \dots + w_{n-1, n-1} \cdot k_{n-1}) \cdot \Delta t) \end{aligned}$$

Vemos que las **k** son relaciones de recurrencia, o sea, que para calcular  $k_2$  hace falta  $k_1$ , y para  $k_3$  hace falta  $k_2$ , etc. Como cada **k** es una evaluación funcional, esta recurrencia hace que los métodos de Runge-Kutta sean eficientes para cálculos en computadora.

Existen varios métodos, en función de hasta que orden **n** usemos en los cálculos. Observemos que el método de primer orden ( $n=1$ ) es el conocido método de Euler:



El número del orden que usemos representará el valor de la aproximación al valor real. Los coeficientes  $\theta$  y  $w$  se obtienen de igualar la ecuación a un desarrollo en serie de Taylor. Los errores de truncamiento local son del orden de  $\Delta t^n$  por lo que para minimizar el error se deberán usar intervalos de tiempo muy pequeños y un orden de desarrollo elevado.

Para nuestros propósitos hemos usado el método de Runge-Kutta de cuarto orden, por ser el más popular y tener una buena relación entre la precisión obtenida y el esfuerzo de cálculo preciso.

Hay infinitud de versiones de métodos de cuarto orden, hemos optado por la más común o **método clásico de cuarto orden** [4]. En éste la función de incremento resulta como sigue:

$$\phi = \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4$$

con :

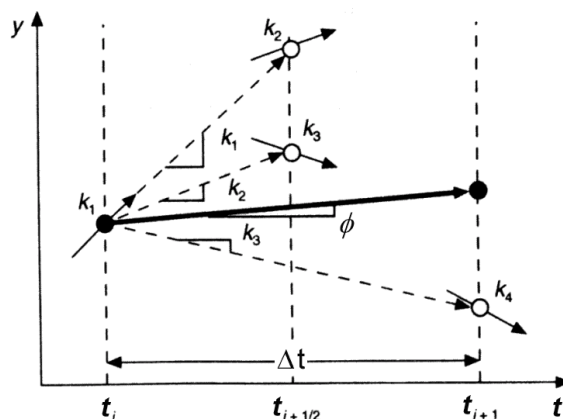
$$k_1 = f(t, y)$$

$$k_2 = f\left(t + \frac{1}{2}\Delta t, y + \frac{1}{2}k_1\Delta t\right)$$

$$k_3 = f\left(t + \frac{1}{2}\Delta t, y + \frac{1}{2}k_2\Delta t\right)$$

$$k_4 = f(t + \Delta t, y + k_3\Delta t)$$

El esquema de funcionamiento de este método lo podemos ver más claramente con el gráfico siguiente:



Obsérvese que en primer lugar desarrollamos las pendientes para todas las variables en el valor inicial. Estas pendientes (un conjunto de las  $k_1$ ) se usarán entonces para hacer predicciones de la variable dependiente en el punto medio del intervalo. Dichos valores de punto medio se utilizan a su vez para calcular un conjunto de pendientes en el punto medio (las  $k_2$ ). Estas nuevas pendientes se toman como punto de inicio para hacer otro conjunto de predicciones de punto medio que nos



llevan a unas nuevas predicciones de pendiente en el punto medio (las k3). Usaremos estas para hacer predicciones al final del intervalo y sus correspondientes pendientes (k4). Por último, las k se combinan en un conjunto de funciones incremento y son llevadas de nuevo al inicio para hacer la predicción final.

La implementación informática de este proceso no tiene especial dificultad y la exponemos a continuación.

En primer lugar, el cálculo de los coeficientes k, se realiza en las siguientes líneas:

```

○ 'CALCULO DE K1,K2,K3,K4.....
○ For k = 1 To 4
○   Select Case k
○     Case 1: ul = 0
○     Case 2: ul = 0.5
○     Case 3: ul = 0.5
○     Case 4: ul = 1
○   End Select
○   tiempo = dt * iterac + ul * dt
○
○   For i = 1 To np2
○     For j = 1 To 3
○       If k > 1 Then AUXa = ul * AKa(i, j, k - 1): AUXb = ul * _
○         AKb(i, j, k - 1) Else AUXa = 0: AUXb = 0
○       ck(i, j) = C(i, j) + AUXa
○       vk(i, j) = v(i, j) + AUXb
○     Next j
○   Next i
○   For i = 1 To ne
○     For j = 1 To 2
○       If k > 1 Then AUXba = ul * AKba(i, j, k - 1) Else AUXaa = 0: AUXba = 0
○       cka(i, j) = CA(i, j) + AUXaa
○       vka(i, j) = va(i, j) + AUXba
○     Next j
○   Next i
○   Call A3n(ck, cka, vk, vka, ac, aca, tiempo, fueraxil, fuercort, flec)
○   For j = 1 To 3
○     For i = 1 To np2
○       AKa(i, j, k) = vk(i, j) * dt
○       AKb(i, j, k) = ac(i, j) * dt
○     Next i
○   Next j
○   For j = 1 To 2
○     For i = 1 To ne
○       AKaa(i, j, k) = vka(i, j) * dt
○       AKba(i, j, k) = aca(i, j) * dt
○     Next i
○   Next j
○ Next k

```

y a continuación empleamos estos valores para el cálculo de las nuevas posiciones de los nudos según el esquema que exponemos a continuación:

```

○ 'CALCULO DE LOS DESPLAZAMIENTOS por Runge-Kutta.....
○ For j = 1 To 3
○   For i = 1 To np2
○     desp = (AKa(i, j, 1) + 2 * AKa(i, j, 2) + 2 * AKa(i, j, 3) + AKa(i, j, 4)) / 6
○     C(i, j) = C(i, j) + desp
○     v(i, j) = v(i, j) + (AKb(i, j, 1) + 2 * AKb(i, j, 2) + 2 * AKb(i, j, 3) + _
○       AKb(i, j, 4)) / 6
○     vabs = Abs(v(i, j))
○   Next i
○ Next j
○ For j = 1 To 2
○   For i = 1 To ne 'nudos intermedios
○     CA(i, j) = CA(i, j) + (AKaa(i, j, 1) + 2 * AKaa(i, j, 2) + 2 * AKaa(i, j, 3) + _
○       AKaa(i, j, 4)) / 6
○     va(i, j) = va(i, j) + (AKba(i, j, 1) + 2 * AKba(i, j, 2) + 2 * AKba(i, j, 3) + _
○       AKba(i, j, 4)) / 6
○   Next i
○ Next j

```

Si bien el resultado es satisfactorio, sigue siendo en realidad un método de paso o intervalo simple, o sea la predicción de la situación en un intervalo se basa únicamente en los datos del intervalo inmediatamente anterior.

En la búsqueda de mayor precisión decidimos dar el paso a los métodos de intervalo múltiple

### Método de Adams-Moulton.

En el método de intervalo sencillo de Runge-Kutta, el valor de la función en el intervalo siguiente, sólo depende del valor de la función en el intervalo anterior. En un método de intervalo múltiple la función se aproxima por un polinomio que pasa por varios puntos previos y muy posiblemente por el siguiente punto buscado  $y_{i+1}$ .

Si en el proceso alcanzamos un punto con solución aproximada  $y_i \cong y(t_i)$ , usualmente habrá varias soluciones aproximadas:  $y_{i+1-j} \cong y(t_{i+1-j})$ ;  $j = 2, 3, \dots, p$

De la propia ecuación diferencial se pueden obtener aproximaciones a la derivada e la forma:

$$\dot{y}_{i+1-j} = f(t_{i+1-j}, y(t_{i+1-j})) \cong f(t_{i+1-j}, y_{i+1-j}) = f_{i+1-j}$$

Esta información se puede extrapolar para obtener el siguiente valor de la función:

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} \dot{y}(t) dt \cong y(t_i) + \int_{t_i}^{t_{i+1}} f(t, y(t)) dt$$

Si integramos el polinomio  $P(t)$  que interpola  $f_{i+1-j}$  en  $t_{i+1-j}$  ( $j=1, 2, \dots, p$ ) en lugar de  $f$ , entonces obtenemos la fórmula de Adams-Bashforth:

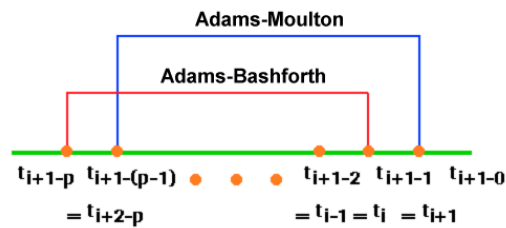
$$y_{i+1} = y_i + \Delta t \sum_{j=1}^p \alpha_{p,j} \cdot f_{i+1-j}$$

Las fórmulas implícitas de Adams-Moulton surgen cuando el polinomio  $P(t)$  interpola  $f_{i+1-j}$  para  $j=0, 1, \dots, p-1$

$$y_{i+1} = y_i + \Delta t \sum_{j=0}^{p-1} \hat{\alpha}_{p,j} \cdot f_{i+1-j}$$

cuando  $j=p-1$ , la parte derecha contiene el término  $f_{i+1} = f(t_{i+1}, y_{i+1})$ , y vemos que  $y_{i+1}$  está definida solo implícitamente en esta fórmula. La solución se completa primero 'prediciendo' el resultado usando la fórmula explícita de Adams-Bashforth y después 'corrigiendo' la misma usando la fórmula implícita.

Si  $L$  es un límite de  $\left| \frac{\partial f}{\partial y} \right|$  y el intervalo  $\Delta t$  es suficientemente pequeño, entonces para una constante  $\rho$ , tal que  $|\Delta t \cdot \hat{\alpha}_{k,0}| L \leq \rho < 1$ , la fórmula de Adams-Moulton tiene una única solución  $y_{i+1}$  y el error decrece como un factor de  $\rho$  en cada iteración. Para valores de  $\Delta t$  pequeños, la iteración converge rápidamente.



Hagamos notar que las fórmulas de Adams-Bashforth y Adams-Moulton interpolan la función  $f$  en el mismo número de puntos, pero están desplazadas un intervalo, siendo más precisa y estable la formulación de Adams-Moulton para un orden dado.

El resultado de los cálculos anteriores será una expresión del tipo:

$$y_{i+1} = y_i + \Delta t (\beta_0 \dot{y}_{i+1} + \beta_1 \dot{y}_i + \beta_2 \dot{y}_{i-1} + \beta_3 \dot{y}_{i-2} + \dots)$$

Resultando métodos explícitos aquellos en que  $\beta_0=0$ , e implícitos en caso contrario. El orden del método depende de cuantos pasos previos tomemos para alcanzar el siguiente valor de  $y$ .

La implementación de este tipo de métodos presenta habitualmente un par de problemas:

- Puesto que las fórmulas requieren resultados igualmente espaciados, autoajustar el tamaño del intervalo es difícil
- El arranque también presenta problemas, pues se precisan aparte de los valores iniciales, varios puntos previos a los iniciales, de los cuales, lógicamente, no dispondremos.

Para nuestro caso hemos optado por el método de cuarto orden, siendo para este paso el **predictor**:

$$y_{i+1} = y_i + \frac{\Delta t}{24} (55 \dot{y}_i - 59 \dot{y}_{i-1} + 37 \dot{y}_{i-2} - 9 \dot{y}_{i-3})$$

y el **corrector**:

$$y_{i+1} = y_i + \frac{\Delta t}{24} (9 \dot{y}_{i+1} + 19 \dot{y}_i - 5 \dot{y}_{i-1} + \dot{y}_{i-2})$$

Para solucionar el problema del arranque, éste lo realizamos con el anterior método de Runge-Kutta hasta el paso 4, y a partir de aquí tomará el relevo el método de Adams-Moulton.

La implementación en el programa informático de este método se muestra en las líneas siguientes:

```

○ 'PREDICCIÓN.....
○ For j = 1 To 3
○   For i = 1 To np2
○     ck(i, j) = cp(i, j, nCi) + dt * (55 * vp(i, j, nCi) - 59 * vp(i, j, nci1) + 37 * _
○     vp(i, j, nci2) - 9 * vp(i, j, nci3)) / 24
○     vk(i, j) = vp(i, j, nCi) + dt * (55 * ap(i, j, nCi) - 59 * ap(i, j, nci1) + 37 * _
○     ap(i, j, nci2) - 9 * ap(i, j, nci3)) / 24
○   Next i
○ Next j
○ For i = 1 To np
○   For j = 1 To 3
○     If nx(i, j) <> 1234 Then ck(i, j) = C0(i, j): vk(i, j) = 0
○   Next j
○ Next i
○
○ For j = 1 To 2

```

```

O   For i = 1 To ne
O     cka(i, j) = cpa(i, j, nCi) + dt * (55 * vpa(i, j, nCi) - 59 * vpa(i, j, nci1) + _
O       37 * vpa(i, j, nci2) - 9 * vpa(i, j, nci3)) / 24
O     vka(i, j) = vpa(i, j, nCi) + dt * (55 * apa(i, j, nCi) - 59 * apa(i, j, nci1) + _
O       37 * apa(i, j, nci2) - 9 * apa(i, j, nci3)) / 24
O   Next i
O Next j
O 'calcula aceleración...
O Call A3n(ck, cka, vk, vka, ac, aca, tiempo, fueraxil, fuercort, flec)

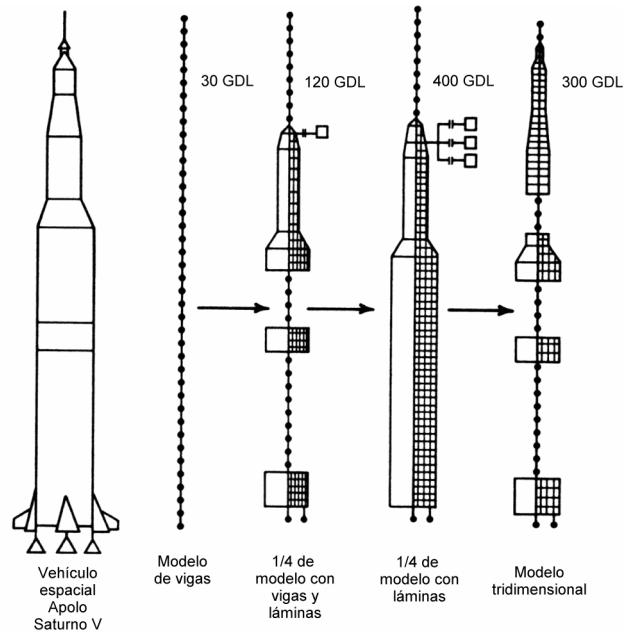
O 'CORRECCION.....
O For j = 1 To 3
O   For i = 1 To np2
O     v(i, j) = vp(i, j, nCi) + dt * (9 * ac(i, j) + 19 * ap(i, j, nCi) - _
O       5 * ap(i, j, nci1) + ap(i, j, nci2)) / 24
O     vabs = Abs(v(i, j))
O     desp = dt * (9 * v(i, j) + 19 * vp(i, j, nCi) - 5 * vp(i, j, nci1) + _
O       vp(i, j, nci2)) / 24
O     C(i, j) = cp(i, j, nCi) + desp
O   Next i
O Next j
O
O For i = 1 To np
O   For j = 1 To 3
O     If nx(i, j) <> 1234 Then C(i, j) = C0(i, j): v(i, j) = 0
O   Next j
O Next i
O
O For i = 1 To ne
O   For j = 1 To 2
O     va(i, j) = vpa(i, j, nCi) + dt * (9 * aca(i, j) + 19 * apa(i, j, nCi) - _
O       5 * apa(i, j, nci1) + apa(i, j, nci2)) / 24
O     CA(i, j) = cpa(i, j, nCi) + dt * (9 * va(i, j) + 19 * vpa(i, j, nCi) - _
O       5 * vpa(i, j, nci1) + vpa(i, j, nci2)) / 24
O     cpa(i, j, ncim1) = CA(i, j)
O     vpa(i, j, ncim1) = va(i, j)
O   Next j
O   If (CA(i, 2) > 2 * pi) Then
O     cpa(i, 2, ncim1) = cpa(i, 2, ncim1) - 2 * pi
O     cpa(i, 2, nCi) = cpa(i, 2, nCi) - 2 * pi
O     cpa(i, 2, nci1) = cpa(i, 2, nci1) - 2 * pi
O     cpa(i, 2, nci2) = cpa(i, 2, nci2) - 2 * pi
O   End If
O   If (CA(i, 2) < 0) Then
O     cpa(i, 2, ncim1) = cpa(i, 2, ncim1) + 2 * pi
O     cpa(i, 2, nCi) = cpa(i, 2, nCi) + 2 * pi
O     cpa(i, 2, nci1) = cpa(i, 2, nci1) + 2 * pi
O     cpa(i, 2, nci2) = cpa(i, 2, nci2) + 2 * pi
O   End If
O Next i
O 'calcula aceleración...
O Call A3n(C, CA, v, va, ac, aca, tiempo, fueraxil, fuercort, flec)
O For i = 1 To np2
O   For j = 1 To 3
O     cp(i, j, ncim1) = C(i, j)
O     vp(i, j, ncim1) = v(i, j)
O     ap(i, j, ncim1) = ac(i, j)
O   Next j
O Next i
O For i = 1 To ne
O   For j = 1 To 2
O     apa(i, j, ncim1) = aca(i, j)
O   Next j
O Next i

```

## MODELIZACIÓN DE LAS BARRAS.

Para pasar de la estructura real al modelo matemático debemos decidir de qué modo efectuamos la idealización. Una misma estructura admitirá múltiples idealizaciones, y todas serán correctas dependiendo de los resultados que esperemos encontrar en cada una de ellas.

Por ejemplo, a continuación vemos los modelos de análisis que se fueron creando para estudiar el comportamiento de la nave espacial Apolo Saturno V.



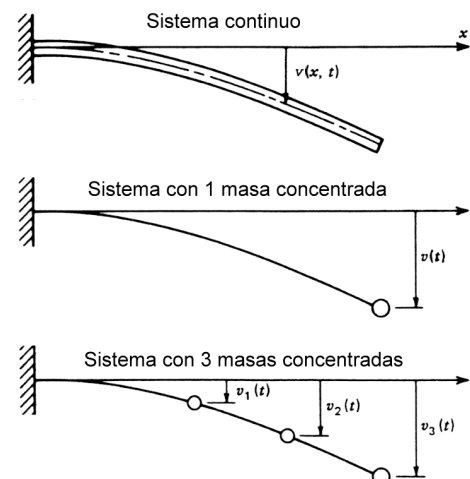
Las dos categorías básicas de los modelos para análisis son:

- modelos continuos
- modelos discretos

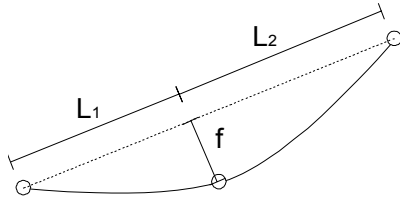
El número de desplazamientos que deben ser considerados para que queden representadas todas las fuerzas de inercia significativas se llama **Número de Grados de Libertad**.

Así un modelo continuo será un sistema con infinitos grados de libertad, mientras que un sistema discreto tendrá más o menos dependiendo del tipo de discretización elegido.

Así para el caso más sencillo, como puede ser una viga en voladizo, en primer lugar tenemos el **sistema continuo**, y a continuación dos posibles **sistemas de masas concentradas**, llamados así porque se representa la masa del sistema por medio de un reducido número de masas puntuales o partículas.

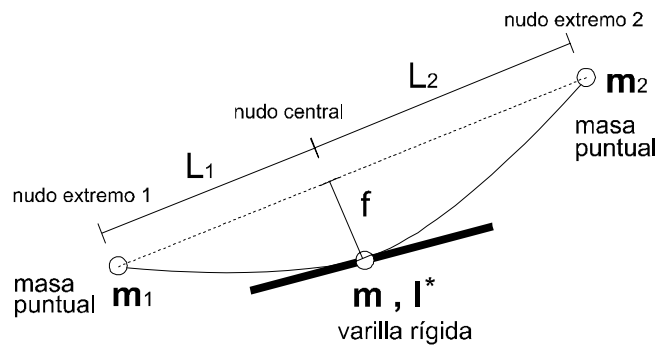


Para el caso de las mallas despegables que estamos estudiando, parece que el planteamiento inicial más claro es representar cada barra por medio de un modelo de tres masas concentrado en los puntos de conexión con las otras barras.



Este modelo de tres nudos, con tres grados de libertad por nudo, parece que cumple bien los objetivos de ser lo suficientemente completo como para representar bien el movimiento de cada barra, a la vez que no es excesivamente complejo y por tanto no ralentizará demasiado los cálculos.

Sin embargo, en las pruebas preliminares, este modelo parecía no funcionar tan bien como debiera, sobre todo cuando la estructura tenía barras de tamaños muy dispares. Por ello decidimos cambiar a un modelo un poco más complejo, sustituyendo el nudo central por una varilla unidimensional rígida, de modo que a las coordenadas del nudo (3 variables) le añadíamos dos grados de libertad más que serían los ángulos que definen la orientación de la varilla.



Con este tipo de modelo se puede simular bien los nudos articulados en los extremos y la continuidad del nudo intermedio.

### Grados de libertad.

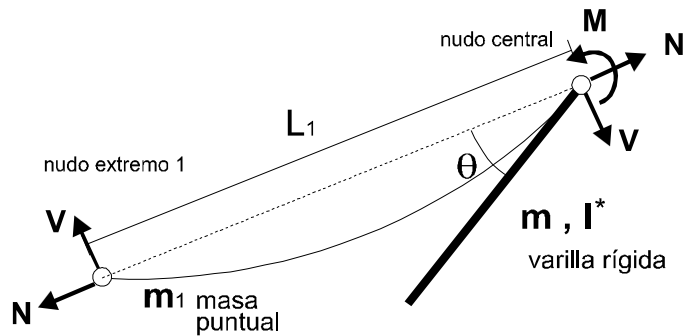
De esta forma nuestro modelo de barra estará definido por 3 coordenadas por cada nudo extremo y 5 coordenadas correspondientes a los tres desplazamientos y dos rotaciones de la varilla central.

Por tanto los grados de libertad a calcular para toda la estructura serán 11 por cada barra, menos 3 por cada cruce de barras (los dos nudos centrales deberán tener los mismos desplazamientos), menos  $3 \cdot (n-1)$  por cada enlace de nudos extremos donde confluyen  $n$  barras, y menos los grados de libertad que estén impedidos por coacciones externas.

## ESFUERZOS EN LA BARRA.

Para el estudio de los esfuerzos podemos considerar cada semibarra independientemente, conforme la figura siguiente.

Para el cálculo de los esfuerzos  $N$ ,  $V$  y  $M$ , asumimos una aproximación cuasi estática, obteniendo estos valores de las condiciones de equilibrio, y suponiendo que son todavía válidas.



Resulta entonces:

$$N = \frac{E \cdot A}{L_1} \cdot \Delta L_1$$

$$V = 3 \frac{E \cdot I}{L^2} \cdot \theta$$

$$M = 3 \frac{E \cdot I}{L} \cdot \theta$$

donde:

**E** es el módulo de elasticidad del material

**A** la sección transversal de la barra

**$L_1$**  la longitud de la semibarra

**$\Delta L_1$**  la variación de longitud de la misma

**I** el momento de inercia de la sección de la barra

**$\theta$**  el ángulo en radianes.

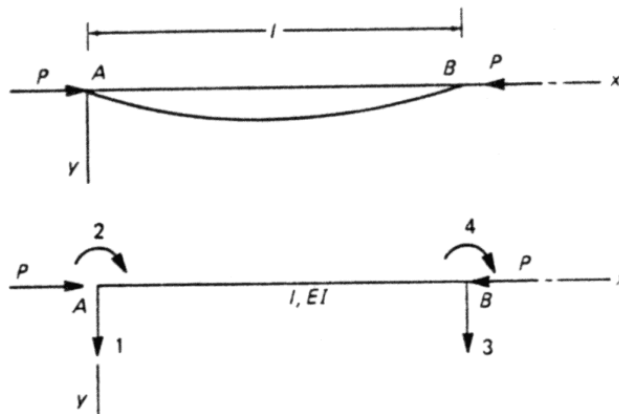
De momento hemos despreciado el efecto de los esfuerzos de torsión.

Después del cálculo de ambas semibarras, obtenemos, debido a las fuerzas internas, 6 esfuerzos en el nudo central y 2 en cada nudo extremo. Repetimos este proceso para cada semibarra y añadiendo las fuerzas externas y las de tipo gravitatorio, obtenemos las fuerzas resultantes sobre cada nudo central o extremo.

## VARIACIÓN DE LA RIGIDEZ POR EFECTO DEL AXIL.

Un punto interesante a tener en cuenta en estas barras son los efectos de segundo orden que se originan cuando el axil adquiere cierta importancia y provoca variaciones en la rigidez a flexión de las barras.

Este tipo de consideraciones introducen un alto grado de no linealidad en los cálculos, por lo que los programas que realizan cálculo dinámico ya ni siquiera plantean esta posibilidad. En nuestro caso concreto, gracias a la idealización realizada de las barras, nos es factible tenerlo en cuenta, aunque debemos considerar que, lógicamente, su inclusión penalizará ligeramente la rapidez del proceso de cálculo.



Para su estudio partimos del gráfico anterior, donde se plantea una viga con un axil P y unos momentos y cortantes en los extremos. La ecuación diferencial que gobierna la desviación  $y$  de un miembro prismático **AB** sometido a una fuerza de compresión **P** y a cualquier restricción de extremo es:

$$\frac{d^4 y}{dx^4} + \frac{P}{EI} \frac{d^2 y}{dx^2} = \frac{q}{EI}$$

donde  $q$  es la intensidad de la carga transversal. Cuando  $q=0$ , la solución general adquiere la forma [5]:

$$y = A_1 \cdot \operatorname{senu} \frac{x}{L} + A_2 \cdot \operatorname{cosu} \frac{x}{L} + A_3 \cdot x + A_4$$

con :

$$u = L \cdot \sqrt{\frac{P}{EI}}$$

siendo  $A_1, A_2, A_3, A_4$  las constantes de integración que se deben determinar de las condiciones de borde.



Si consideramos el vector de desplazamientos  $\{D\}$  definido de la forma siguiente:

$$\begin{aligned} D_1 &= (y)_{x=0} \\ D_2 &= \left(\frac{dy}{dx}\right)_{x=0} \\ D_3 &= (y)_{x=L} \\ D_4 &= \left(\frac{dy}{dx}\right)_{x=L} \end{aligned}$$

Este vector se relaciona con las constantes  $\{A\}$  mediante la ecuación:

$$\begin{Bmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ \frac{u}{L} & 0 & 1 & 0 \\ s & c & L & 1 \\ \frac{u}{L}c & -\frac{u}{L}s & 1 & 0 \end{bmatrix} \cdot \begin{Bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{Bmatrix} \Rightarrow \{D\} = [B]\{A\}$$

con

$$s = \text{sen } u$$

$$c = \text{cos } u$$

Las fuerzas  $\{F\}$  en los extremos son el esfuerzo cortante y el momento flector, por tanto:

$$\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{Bmatrix} = \begin{Bmatrix} (-V)_{x=0} \\ (M)_{x=0} \\ (V)_{x=L} \\ (-M)_{x=L} \end{Bmatrix} = E \cdot I \cdot \begin{Bmatrix} \left(\frac{d^3y}{dx^3} + \frac{u^2}{L^2} \frac{dy}{dx}\right)_{x=0} \\ \left(\frac{d^2y}{dx^2}\right)_{x=0} \\ \left(\frac{d^3y}{dx^3} + \frac{u^2}{L^2} \frac{dy}{dx}\right)_{x=L} \\ \left(\frac{d^2y}{dx^2}\right)_{x=L} \end{Bmatrix}$$

Si diferenciamos la solución inicial y sustituimos en esta última, tendremos:

$$\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{Bmatrix} = E \cdot I \cdot \begin{bmatrix} 0 & 0 & \frac{u^2}{L^2} & 0 \\ 0 & \frac{u^2}{L^2} & 0 & 0 \\ 0 & 0 & -\frac{u^2}{L^2} & 0 \\ -\frac{s \cdot u^2}{L^2} & -\frac{c \cdot u^2}{L^2} & 0 & 0 \end{bmatrix} \cdot \begin{Bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{Bmatrix} \Rightarrow \{F\} = [C]\{A\}$$

por tanto:

$$\{F\} = [C][B]^{-1}\{D\}$$

$$\{F\} = [S]\{D\}$$

siendo

$$[B]^{-1} = \frac{1}{2-2c-us} \begin{bmatrix} -s & \frac{L}{u}(1-c-us) & s & -\frac{L}{u}(1-c) \\ (L-c) & \frac{L}{u}(s-uc) & -(L-c) & \frac{L}{u}(u-s) \\ \frac{u}{L}s & (L-c) & -\frac{u}{L}s & (1-c) \\ (1-c-us) & -\frac{L}{u}(s-uc) & (L-c) & -\frac{L}{u}(u-s) \end{bmatrix}$$

y

$$[S] = \frac{EI}{(2-2c-us)} \begin{bmatrix} S_1 & S_2 & S_4 & S_2 \\ S_2 & S_3 & S_2 & S_5 \\ S_4 & S_2 & S_1 & S_2 \\ S_2 & S_5 & S_2 & S_3 \end{bmatrix}$$

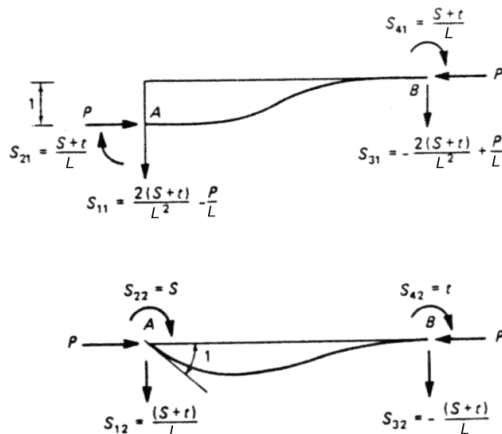
con

$$\begin{aligned} S_1 &= \frac{u^3 s}{L^3} & S_4 &= -\frac{u^3 s}{L^3} \\ S_2 &= \frac{u^2(L-c)}{L^2} & S_5 &= \frac{u(u-s)}{L} \\ S_3 &= \frac{u(s-uc)}{L} \end{aligned}$$

Cuando la fuerza axial  $P$  tiende a cero, la matriz de rigidez anterior se hará idéntica a la matriz de rigidez habitual de una barra:

$$\lim_{u \rightarrow 0} [S] = EI \cdot \begin{bmatrix} \frac{12}{L^3} & \frac{6}{L^2} & -\frac{12}{L^3} & \frac{6}{L^2} \\ \frac{6}{L^2} & \frac{4}{L} & -\frac{6}{L^2} & \frac{2}{L} \\ -\frac{12}{L^3} & -\frac{6}{L^2} & \frac{12}{L^3} & -\frac{6}{L^2} \\ \frac{6}{L^2} & \frac{2}{L} & -\frac{6}{L^2} & \frac{4}{L} \end{bmatrix}$$

Por tanto las rigideces a desplazamiento o giro de uno de los extremos quedan expresadas en el gráfico siguiente



siendo:

$$S = \frac{u(s - u \cdot c)}{(2 - 2c - u \cdot s)} \cdot \frac{E \cdot I}{L}$$

$$t = \frac{u(u - s)}{(2 - 2c - u \cdot s)} \cdot \frac{E \cdot I}{L}$$

Cuando la fuerza P alcanza el valor crítico de pandeo, S se convierte en cero, lo que significa que el desplazamiento D2 se puede producir con un valor infinitamente pequeño de la fuerza F2. El valor de S es cero cuando  $\tan(u) = u$ . El valor más pequeño de u que satisface esta ecuación es:

$$u = L \cdot \sqrt{\frac{P_{crit}}{E \cdot I}} = 4,49$$

Si solo se plantea el giro de un extremo, la carga crítica será:

$$P_{crit} = 20,19 \cdot \frac{E \cdot I}{L^2}$$

### Barra traccionada.

Si pasamos de compresión a tracción, sustituiremos P por -P, por lo tanto el valor de u pasará a ser:

$$u = L \cdot \sqrt{\frac{-P}{E \cdot I}} = i \cdot u \quad \text{con: } i = \sqrt{-1}$$

y puesto que:

$$-i \cdot \text{sen } iu = \text{senh } u$$

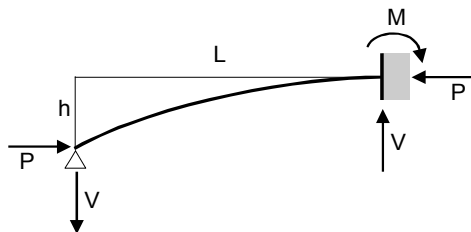
$$\text{cos } iu = \text{cosh } u$$

entonces resultará

$$S = \frac{u(u \cdot \text{cosh } u - \text{senh } u)}{(2 - 2 \cdot \text{cosh } u - u \cdot \text{senh } u)} \cdot \frac{E \cdot I}{L}$$

$$t = \frac{u(\text{senh } u - u)}{(2 - 2 \cdot \text{cosh } u - u \cdot \text{senh } u)} \cdot \frac{E \cdot I}{L}$$

Particularizando para nuestro caso concreto de estudio de una semibarra, obtendríamos la idealización siguiente:



siendo el momento:

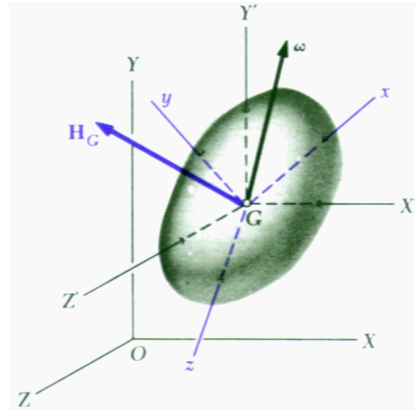
$$M = S - \left( \frac{t^2}{S} \right) \cdot \frac{h}{L}$$

## MOVIMIENTO CINÉTICO DEL SÓLIDO RÍGIDO.

Si bien para los nudos extremos de la estructura son suficientes las ecuaciones de movimiento de una masa puntual, para el nudo central, idealizado como una varilla rígida, deberemos usar las ecuaciones generales de movimiento del sólido rígido [6] y [7]:

$$\Sigma F = m \cdot \bar{a}$$

$$\Sigma M_G = \dot{H}_G = \left( \dot{H}_G \right)_{Gxyz} + \Omega \times H_G$$



Siendo:

$H_G$  momento angular del sólido con respecto al sistema  $Gx'y'z'$  de orientación fija

$\left( \dot{H}_G \right)_{Gxyz}$  la derivada temporal de  $H_G$  con respecto al sistema móvil  $Gxyz$

$\Omega$  velocidad angular del sistema móvil  $Gxyz$

Si el sistema móvil está solidariamente unida al sólido, como vamos a suponer, su velocidad angular  $\Omega$  será idéntica a la velocidad angular  $\omega$  del sólido.

El momento angular o cinético de un sistema depende, como sabemos, del punto con respecto al cual se define. En la mecánica del sólido rígido, lo más racional es escoger este punto en el origen del sistema de coordenadas móvil. La relación entre momento angular y el vector de velocidad angular viene dada por el **tensor de inercia del sólido** respecto su centro de masas.

$$\begin{pmatrix} H_x \\ H_y \\ H_z \end{pmatrix} = \begin{pmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}$$

Si los ejes del sistema móvil se hacen coincidentes con los ejes principales, el tensor de inercia del sólido se simplifica quedando:

$$\begin{pmatrix} I_{x'} & 0 & 0 \\ 0 & I_{y'} & 0 \\ 0 & 0 & I_{z'} \end{pmatrix}$$

Entonces sustituyendo en la ecuación vectorial de equilibrio de momentos, obtenemos las tres ecuaciones escalares siguientes:

$$\Sigma M_x = I_x \cdot \dot{\omega}_x - (I_y - I_z) \cdot \omega_y \cdot \omega_z$$

$$\Sigma M_y = I_y \cdot \dot{\omega}_y - (I_z - I_x) \cdot \omega_z \cdot \omega_x$$

$$\Sigma M_z = I_z \cdot \dot{\omega}_z - (I_x - I_y) \cdot \omega_x \cdot \omega_y$$

Estas ecuaciones se llaman ecuaciones de Euler del movimiento en honor al matemático suizo Leonhard Euler, y que junto con las ecuaciones de desplazamiento:

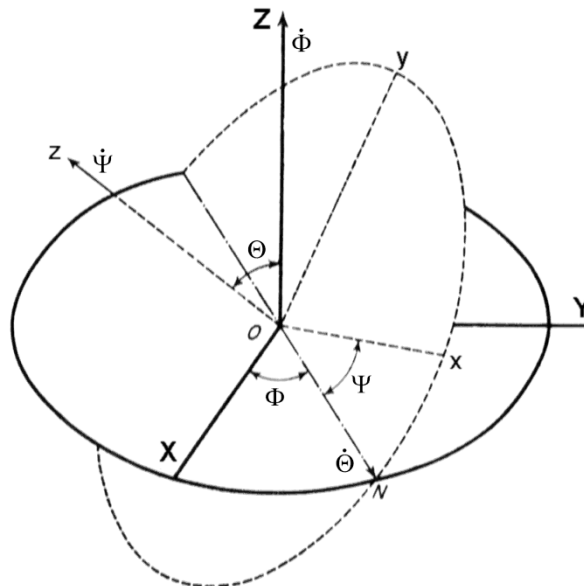
$$\begin{aligned}\Sigma F_x &= m \cdot a_x \\ \Sigma F_y &= m \cdot a_y \\ \Sigma F_z &= m \cdot a_z\end{aligned}$$

forman un sistema de seis ecuaciones diferenciales, que dando las condiciones iniciales adecuadas, tendrán una solución única. son las que posteriormente desarrollaremos para analizar el movimiento de la estructura.

### Ángulos de Euler.

Para describir el movimiento del cuerpo rígido basta con dar las tres coordenadas de su centro de masa y los tres ángulos que definen la orientación de sus ejes móviles (x,y,z) con respecto a los ejes fijos (X,Y,Z). Habitualmente conviene tomar estos ángulos como los **ángulos de Euler**.

Como aquí solo nos interesa detallar los ángulos entre ejes coordenados, dibujaremos ambos sistemas situados en el mismo origen.



El plano móvil **xy** corta al plano fijo **XY** según la recta **ON** llamada **línea nodal**. Esta línea es evidentemente perpendicular simultáneamente al eje **Z** y al eje **z**; se tomará como sentido positivo aquel que corresponda al del producto vectorial  $\underline{z} \times \underline{x}$  (siendo  $\underline{z}$  y  $\underline{x}$  los vectores unitarios según los ejes **Z** y **z** respectivamente).

Tomaremos como magnitudes que determinan la posición de los ejes x,y,z con respecto a los X,Y,Z los siguientes ángulos: el ángulo  $\Theta$  entre los ejes **Z** y **z**, el ángulo  $\Phi$  entre el eje **X** y **ON**, y el ángulo  $\Psi$  entre el eje **x** y **ON**. Los ángulos  $\Phi$  y  $\Psi$  se miden en el sentido definido por la regla del sacacorchos, respectivamente alrededor de los ejes **Z** y **z**. El ángulo  $\Theta$  toma valores de cero a  $\pi$  y los ángulos  $\Phi$  y  $\Psi$  de cero a  $2\pi$ .

Expresaremos ahora las componentes del vector velocidad angular  $\Omega$  sobre los ejes móviles x,y,z en función de los ángulos de Euler y de sus derivadas.

Para ello debemos hallar las componentes según esos ejes de las velocidades angulares  $\dot{\Theta}, \dot{\Phi}, \dot{\Psi}$ . La velocidad angular  $\dot{\Theta}$  está dirigida según la línea nodal **ON** y sus componentes sobre los ejes x,y,z son:

$$\begin{aligned}\dot{\Theta}_1 &= \dot{\Theta} \cdot \cos \Psi \\ \dot{\Theta}_2 &= -\dot{\Theta} \cdot \sin \Psi \\ \dot{\Theta}_3 &= 0\end{aligned}$$

La velocidad angular  $\dot{\Phi}$  está dirigida según el eje **Z**; su componente según **z** es  $\dot{\Phi}_z = \dot{\Phi} \cdot \cos \Theta$ , y su proyección en el plano x,y es  $\dot{\Phi} \cdot \sin \Theta$ . Descomponiendo esta última sobre los ejes x,y se obtiene:

$$\begin{aligned}\dot{\Phi}_1 &= \dot{\Phi} \cdot \sin \Theta \cdot \sin \Psi \\ \dot{\Phi}_2 &= \dot{\Phi} \cdot \sin \Theta \cdot \cos \Psi\end{aligned}$$

Finalmente, la velocidad angular  $\dot{\Psi}$  está dirigida según el eje **z**. Reuniendo las componentes a lo largo de cada eje se obtiene:

$$\begin{aligned}\dot{\Phi}_1 &= \dot{\Phi} \cdot \sin \Theta \cdot \sin \Psi + \dot{\Theta} \cdot \cos \Psi \\ \dot{\Phi}_2 &= \dot{\Phi} \cdot \sin \Theta \cdot \cos \Psi - \dot{\Theta} \cdot \sin \Psi \\ \dot{\Phi}_3 &= \dot{\Phi} \cdot \cos \Theta + \dot{\Psi}\end{aligned}$$

## Amortiguamiento

En la realidad todos los movimientos sufren amortiguamientos hasta cierto grado debido a las fuerzas de rozamiento. Este rozamiento puede ser entre sólidos rígidos, con fluidos o por rozamiento interno entre las moléculas de un sólido que parece elástico.

Un tipo de amortiguamiento de especial interés es el **amortiguamiento viscoso**, causado por el rozamiento a velocidades bajas. Está caracterizado por el hecho de que la fuerza de fricción es directamente proporcional a la velocidad del cuerpo en movimiento, o sea:  $F = -\lambda v$

Entonces las ecuaciones del movimiento serán:

$$\begin{aligned}\Sigma F &= m \cdot a - \lambda \cdot v \\ \Sigma M_G &= \left( \dot{H}_G \right)_{Gxyz} + \Omega \times H_G - \lambda \cdot \Omega\end{aligned}$$

Otro tipo de **rozamiento** que presenta interés es el de rozamiento por contacto. La misma depende de que el cuerpo esté en reposo o en movimiento. El valor de la fuerza será:

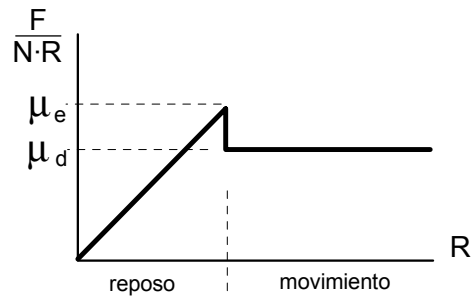
$$\begin{aligned}F_e &\leq -\mu_e \cdot N \cdot R \\ F_d &= -\mu_d \cdot N \cdot R\end{aligned}$$

Siendo:

- F** la fuerza de rozamiento
- $\mu$**  el coeficiente de rozamiento
- N** la fuerza normal de contacto entre superficies
- R** la fuerza que mueve o intenta mover el cuerpo

los subíndices **e** y **d** hacen referencia a los casos estático y dinámico respectivamente.

En general  $\mu_d < \mu_e$ . Si representamos el valor del rozamiento en función de la fuerza aplicada sobre el cuerpo, tendríamos la gráfica:



Existen otras formas de considerar el amortiguamiento de un sistema, pero estas son las que consideramos más apropiadas para aplicar en las estructuras de estudio.

## ECUACIONES DE MOVIMIENTO APLICADAS A UNA BARRA.

En este apartado expondremos como resultan las ecuaciones particulares de movimiento del cuerpo rígido cuando se particularizan para la idealización que hemos hecho de los nudos de nuestra estructura.

Una vez calculadas las fuerzas resultantes aplicadas sobre cada nudo, las ecuaciones del movimiento de un nudo  $i$  extremo, serán:

$$a_{ix} = \frac{R_{ix}}{m_i} - \lambda \cdot v_{ix}$$

$$a_{iy} = \frac{R_{iy}}{m_i} - \lambda \cdot v_{iy}$$

$$a_{iz} = \frac{R_{iz}}{m_i} - \lambda \cdot v_{iz}$$

donde:

$a_i$  es la aceleración del nudo  $i$

$v_i$  es la velocidad del nudo  $i$

$R_i$  es la resultante de las fuerzas del nudo  $i$

$\lambda$  es el factor de amortiguamiento viscoso

y si asumimos, como hemos hecho, que el nudo central se comportará como un cuerpo rígido unidimensional (varilla rígida), en este caso las ecuaciones del movimiento para un nudo medio  $k$  serán [8]:

$$a_{kx} = \frac{R_{kx}}{m_k} - \lambda \cdot v_{kx}$$

$$a_{ky} = \frac{R_{ky}}{m_k} - \lambda \cdot v_{ky}$$

$$a_{kz} = \frac{R_{kz}}{m_k} - \lambda \cdot v_{kz}$$

$$\frac{d^2\Theta_k}{dt^2} = \frac{Ma_k}{I^*} + \left(\frac{d\Phi_k}{dt}\right)^2 \cdot \cos\Theta_k \cdot \sin\Theta_k - \lambda \cdot \frac{d\Theta_k}{dt}$$

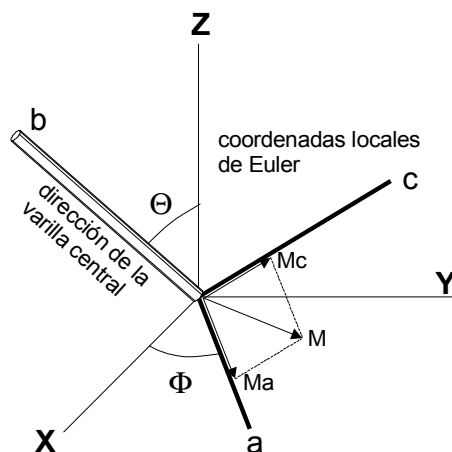
$$\frac{d^2\Phi_k}{dt^2} = \frac{\left(\frac{Mc_k}{I^*} - 2\left(\frac{d\Theta_k}{dt}\right) \cdot \left(\frac{d\Phi_k}{dt}\right) \cdot \cos\Theta_k\right)}{\sin\Theta} - \lambda \cdot \frac{d\Phi_k}{dt}$$

donde:

$I^*$  es el momento de inercia de la varilla central

$Ma$  y  $Mc$  son las proyecciones sobre los ejes  $a$  y  $c$  del momento resultante  $M$

$\Theta$  y  $\Phi$  son las coordenadas de Euler de la varilla  $k$ , según el esquema siguiente

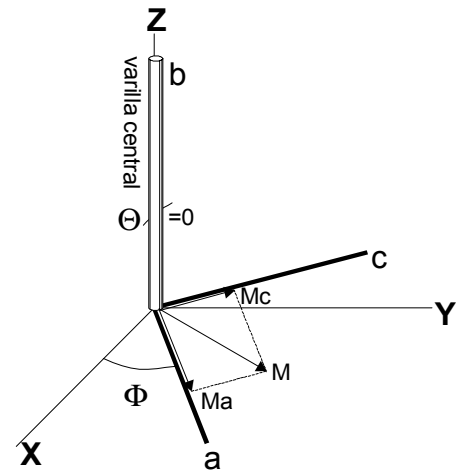


Como problemática particular, vemos que en la última ecuación, con la que obtenemos la aceleración angular, existe un seno en el denominador, por lo que se nos presenta una singularidad cuando el ángulo  $\Theta$  valga cero, o valores muy próximos a cero. Cuando esto ocurre, en un cálculo numérico se produce un error de desbordamiento, y se da cuando la barra alcanza la posición vertical, debido a lo cual lo hemos denominado '**error de verticalidad**'. Como no disponemos de las ecuaciones, no podemos intentar llegar al límite de la función de forma analítica, por ello un primer intento de solución de este problema fue evitando que ninguna barra pase por una posición vertical mediante un giro de toda la estructura un ángulo arbitrario antes de empezar el cálculo, y al final del proceso realizamos el proceso inverso de giro de los resultados para que todo resulte según las coordenadas iniciales.

Generalmente este proceso funciona, y en caso de que dé problemas de desbordamiento la solución será reiniciar con otro ángulo aleatorio. Esta es una solución, aunque no es un método muy elegante y existe todo un trabajo numérico de girar / desgirar, que tiende a ralentizar el proceso.



Si observamos con más detalle la posición problemática, vemos que la aceleración angular que no podemos calcular es la  $\Phi$ , o sea, la que habitualmente representa la aceleración y velocidad angular de la varilla rígida alrededor del eje Z global. En este caso concreto coincidiría con la velocidad angular de la propia varilla alrededor de su eje. Como hemos supuesto la varilla como un sólido unidimensional, este parámetro, efectivamente no tiene ningún sentido, pues no influye en la posición de la varilla cuando esta se halla vertical, es decir su posición será la misma con independencia del valor de  $\Phi$ . Por tanto, a medida que nos acercamos a la vertical se nos complica hallar los parámetros de  $\Phi$ , pero por otra tienen cada vez una influencia o importancia menor en el cálculo.



Como solución a esta problemática singular podemos suponer que cuando estamos lo suficientemente cerca de la vertical como para empezar a producirse errores de desbordamiento, esta velocidad angular puede considerarse constante, con lo que podremos prescindir del valor de la aceleración problemática  $\Phi$ . Con ello le damos solución al problema, sin pérdida de precisión en los cálculos.

Probada esta solución, se verifica un correcto funcionamiento de la misma, y no se aprecian diferencias significativas con respecto la anterior solución.

### RESOLUCIÓN DE LAS ECUACIONES DE MOVIMIENTO.

Como hemos comentado, para la resolución de estas ecuaciones implementamos el método semi-implícito de Adams-Moulton de cuarto orden, que nos da un balance adecuado entre precisión y tiempo de cálculo. Las dos partes de este método, ya adaptadas a nuestro problema concretos, son en primer lugar la predicción aproximada de coordenadas y velocidades de la forma siguiente:

$$u^*(t+dt) = u(t) + \frac{dt}{24} [55 \cdot v(t) - 59 \cdot v(t-dt) + 37 \cdot v(t-2dt) - 9 \cdot v(t+3dt)]$$

$$v^*(t+dt) = v(t) + \frac{dt}{24} [55 \cdot a(t) - 59 \cdot a(t-dt) + 37 \cdot a(t-2dt) - 9 \cdot a(t+3dt)]$$

en este paso evaluamos la aceleración provisional en función de las fuerzas internas y externas del sistema en la posición de la predicción:

$$a^*(t+dt)$$

Conocida esta podemos seguir con el paso de corrección para obtener los valores definitivos:

$$v(t+dt) = v(t) + \frac{dt}{24} [9 \cdot a(t+dt) + 19 \cdot a(t) - 5 \cdot a(t-dt) + a(t+2dt)]$$

$$u(t+dt) = u(t) + \frac{dt}{24} [9 \cdot v(t+dt) + 19 \cdot v(t) - 5 \cdot v(t-dt) + v(t+2dt)]$$

y nuevamente calculamos los valores de las aceleraciones en este punto, pero ahora ya obtenemos los valores que consideramos definitivos para estas iteración:

$$a(t+dt)$$

Y a partir de aquí reiniciamos el proceso para la iteración siguiente, siendo  $dt$  el intervalo de tiempo entre dos iteraciones, valores usuales para el cálculo de estas mallas son del orden de  $10^{-4}$  y  $10^{-5}$  segundos.

## DETERMINACIÓN DE LAS MASAS E INERCIA.

Un punto importante es determinar el valor a aplicar a las masas puntuales y a la masa e inercia de la barra central.

Obviamente las masas deben cumplir:

$$m_1 + m_2 + m = \text{Masa de la barra completa}$$

Y para la inercia de la varilla central, de forma similar debe cumplir:

$$m_1 \cdot L_1^2 + m_2 \cdot L_2^2 + I^* = \text{Inercia total de la barra.}$$

Tenemos un sistema de 2 ecuaciones con 4 incógnitas (las masas e inercia), aunque para simplificar podemos considerar el punto medio totalmente centrado y entonces:

$$m_1 = m_2$$

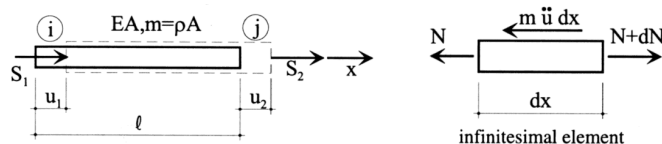
$$L_1 = L_2$$

Si además consideramos El tramo central, como una varilla, con su masa e inercias correspondientes con su longitud, solo tendremos una variable, la longitud del tramo central.


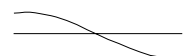

Veremos ahora por tanto los parámetros que rigen el comportamiento dinámico de una barra tomada con sistema continuo y comparándola con un sistema discreto, para darles valores de modo que se ajusten lo mejor posible al comportamiento de la barra real.

## Vibraciones longitudinales de una barra.

Para el estudio del sistema continuo se supone una barra de espesor despreciable frente a su longitud, por lo que se desprecian todas las componentes de los corrimientos que no sean paralelas al eje longitudinal de la varilla.



Este comportamiento no es real pues se desprecian la contracción o dilatación transversal, pero es lo suficientemente buena para varillas largas. Las frecuencias naturales de vibración por flexión para son [9] :

CASO	$k_1$	$k_2$
libre 	1 	2 

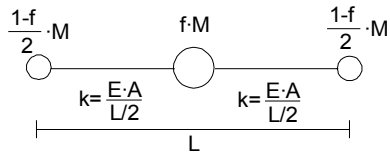
Siendo las frecuencias circulares naturales:

$$\omega_i = k_i \cdot \frac{\pi}{L} \cdot \sqrt{\frac{E}{\rho}} \quad [\text{rad/seg.}]$$

con:

- $\rho$  - masa por unidad de volumen
- $E$  - modulo de Young
- $L$  - longitud de la barra

Para el caso de un sistema discreto, tenemos el esquema típico siguiente, donde llamamos  $f$  al factor de distribución de masas entre la central y las extremos y  $M$  la masa total de la barra. Suponemos para simplificar que el punto central se halla en este caso perfectamente centrado.



Para el modo 1 tenemos un sistema discreto fácil de calcular, puesto que en este modo el punto central permanece quieto y son los extremos los que vibran simétricamente, con lo que mirando la mitad de la barra tenemos en realidad un sistema masa-resorte, cuya frecuencia fundamental es la conocida expresión:

$$\omega = \sqrt{\frac{K}{M}}$$

que desarrollando y llamando  $m$  a la masa por unidad de longitud ( $M=m \cdot L$ ):

$$\omega = \sqrt{\frac{\frac{E \cdot A}{L/2}}{\frac{1-f}{2} \cdot M}} = \sqrt{\frac{4 \cdot E \cdot A}{(1-f) \cdot M \cdot L}} = \sqrt{\frac{4 \cdot E \cdot A}{(1-f) \cdot m \cdot L^2}}$$

Igualando la frecuencia angular de este caso con el de la barra continua:

$$\frac{\pi}{L} \sqrt{\frac{E}{\rho}} = \sqrt{\frac{4 \cdot E \cdot A}{(1-f) \cdot m \cdot L^2}}$$

y resolvemos:

$$\left(\frac{\pi}{L}\right)^2 \cdot \frac{E}{\rho} = \frac{4 \cdot E \cdot A}{(1-f) \cdot m \cdot L^2}$$

$$\frac{\pi^2}{\rho} = \frac{4 \cdot A}{m - f \cdot m}$$

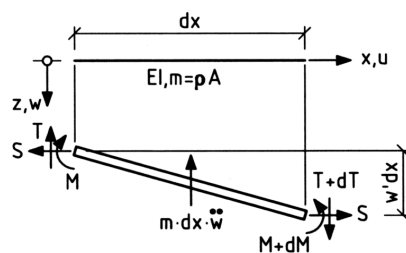
$$m \cdot \pi^2 - f \cdot m \cdot \pi^2 = 4 \cdot A \cdot \rho = 4 \cdot m$$

$$f = \frac{\pi^2 - 4}{\pi^2} = 0,5947$$


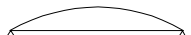
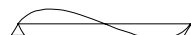
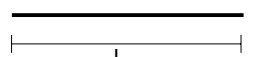


### Vibraciones transversales de una barra.

Las frecuencias naturales de vibración por flexión para la barra clásica, según la teoría de Euler-Bernouilli, en la cual se desprecian las deformaciones por corte y la inercia rotacional, son [10] y [11]:

esquema esfuerzos



Valores de los modos propios:

CASO	$k_1$	$k_2$
biapoyada 	$\pi^2$ 	$4\pi^2$ 
libre 	22.3733 	61.6728 

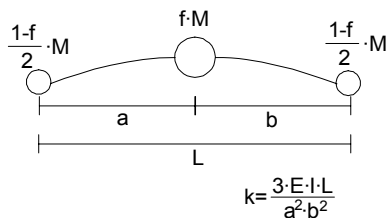
Siendo las frecuencias circulares naturales:

$$\omega_i = k_i \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} \quad [\text{rad/seg.}]$$

con:

- $m$  - masa por unidad de longitud
- $E$  - modulo de Young
- $I$  - Inercia de la sección transversal
- $L$  - longitud de la barra

El sistema discreto será similar al que hemos visto antes, con la diferencia que la constante elástica que nos interesa ahora será la debida a la flexión. También en este caso es fácil de calcular la frecuencia fundamental del modo 1 para una viga biapoyada, pues solo se mueve la masa central, con una fuerza de restitución igual a la rigidez a flexión de la barra:



volvemos por tanto al esquema masa-resorte:

$$\omega = \sqrt{\frac{k}{M}} = \sqrt{\frac{\frac{3 \cdot E \cdot I}{a^2 \cdot b^2}}{f \cdot M}} = \sqrt{\frac{48 \cdot E \cdot I}{f \cdot m \cdot L^4}}$$

igualando con la ecuación correspondiente del sistema continuo y despejando  $f$ :

$$\pi^2 \sqrt{\frac{E \cdot I}{m \cdot L^4}} = \sqrt{\frac{48 \cdot E \cdot I}{f \cdot m \cdot L^4}}$$

$$\pi^4 \cdot \frac{E \cdot I}{m \cdot L^4} = \frac{48 \cdot E \cdot I}{f \cdot m \cdot L^4}$$

$$\pi^4 = \frac{48}{f}$$

$$f = \frac{48}{\pi^4} = 0,49276$$

Vemos que para el caso anterior el valor de  $f$  debería ser del orden de 0,6 y en este caso 0,5. De entrada parece que un valor de **0,55** podría ser adecuado, pero podemos hacer un análisis más a fondo.

### Inercia de la barra en rotación como sólido rígido.

Teníamos que la inercia de la varilla central debería cumplir:

$$m_1 \cdot L_1^2 + m_2 \cdot L_2^2 + I^* = \text{Inercia total de la barra para giro como sólido rígido.}$$

Desarrollando y suponiendo el nudo medio justo en el centro:

$$\begin{aligned} m_1 \cdot L_1^2 + m_2 \cdot L_2^2 + \frac{m_3 \cdot L_c^2}{12} &= \frac{m_T \cdot L^2}{12} \\ 2 \cdot (m_1 \cdot L_1^2) + \frac{m_3 \cdot L_c^2}{12} &= \frac{m_T \cdot L^2}{12} \\ 2 \cdot \left[ \frac{1-f}{2} \cdot L \cdot m \cdot \left(\frac{L}{2}\right)^2 \right] + \frac{f \cdot L \cdot m \cdot (f \cdot L)^2}{12} &= \frac{L \cdot m \cdot L^2}{12} \\ (1-f) \frac{L^3}{4} + \frac{f^2 \cdot L^3}{12} &= \frac{L^3}{12} \\ 3 - 3f + f^2 &= 1 \\ f^2 - 3f + 2 &= 0 \\ f = \frac{3 \pm \sqrt{3^2 - 4 \cdot 2}}{2} = \frac{3 \pm 1}{2} &= \begin{cases} 2 \\ 1 \end{cases} \end{aligned}$$

La solución 1 es el caso de la barra continua, y la solución 2 implica suponer masas negativas en los extremos, pero en cualquier caso ninguna se aproxima a los valores estimados en el punto anterior. Si comparamos la inercia que tenemos ahora con la de la barra real obtenemos:

$$\frac{\text{inercia que tenemos}}{\text{inercia barra real}} = \frac{2 \cdot (m_1 \cdot L_1^2) + \frac{m_3 \cdot L_c^2}{12}}{\frac{m_T \cdot L^2}{12}} = f^2 - 3f + 3 = 1,6525$$

Vemos que es superior. La opción sería usar un factor  $f'$  distinto para inercias del usado para masas, lo que implicaría que la varilla central tendría una longitud  $f' \cdot L$  y una masa por unidad de longitud  $m \cdot f'$ , que daría como resultado que mantiene la misma masa pero varía su inercia. Debiendo entonces valer:

$$\begin{aligned} 2 \cdot (m_1 \cdot L_1^2) + \frac{m_3 \cdot L_c^2}{12} &= \frac{m_T \cdot L^2}{12} \\ 2 \cdot \left[ \frac{1-f}{2} \cdot L \cdot m \cdot \left(\frac{L}{2}\right)^2 \right] + \frac{f \cdot L \cdot m \cdot (f' \cdot L)^2}{12} &= \frac{L \cdot m \cdot L^2}{12} \\ (1-f) \frac{L^3}{4} + \frac{f \cdot f' \cdot L^3}{12} &= \frac{L^3}{12} \\ 3 - 3f + f \cdot f' &= 1 \\ f' = \frac{1 + 3f - 3}{f} &\quad \text{y si } f=0,55 \dots \\ f' &= -0,63 \end{aligned}$$

Resulta que deberíamos tener una inercia negativa, lo cual no parece muy plausible.


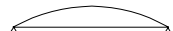
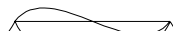
Para el caso límite de  $f=0,666666$   $f'$  nos daría cero, lo que nos conduciría a un modelo de tres nudos.

Para valores mayores de  $f$ , como 0,7 sí tendríamos soluciones positivas, del orden de 0,14.

### Vibraciones transversales de una barra ( 2º modo ).

La otra opción para fijar la inercia del tramo central es mediante el estudio nuevamente de los modos principales de vibración.

El primer modo en que entraría en juego sería en el 2º modo de las vibraciones transversales, que en el caso de una viga biapoyada era:

CASO	$k_1$	$k_2$
biapoyada 	$\pi^2$ 	$4\pi^2$ 

con

$$\omega_i = k_i \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} \quad [\text{rad/seg.}]$$

En este caso, para el modelo discreto las masas no entran en juego, ya que el punto intermedio no se desplaza, sino únicamente la inercia y la rigidez angular. La frecuencia de vibración para el sistema discreto sería:

$$\omega = \sqrt{\frac{k}{I}} = \sqrt{\frac{\frac{12 \cdot E \cdot I}{L}}{(m \cdot L \cdot f) \cdot (L \cdot f'')^2}} = \sqrt{\frac{144 \cdot E \cdot I}{m \cdot L^4 \cdot f \cdot f'^2}}$$

igualando con el anterior:

$$4 \cdot \pi^2 \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} = \sqrt{\frac{144 \cdot E \cdot I}{m \cdot L^4 \cdot f \cdot f'^2}}$$

$$16 \cdot \pi^4 \cdot \frac{E \cdot I}{m \cdot L^4} = \frac{144 \cdot E \cdot I}{m \cdot L^4 \cdot f \cdot f'^2}$$

$$f' = \sqrt{\frac{9}{f \cdot \pi^4}} = \frac{3}{\pi} \cdot \sqrt{\frac{1}{f}} = 0,4099$$

Vemos que este puede ser un buen valor de referencia, pues es como reacciona la barra en un primer instante ante los intentos de giro debidos a los momentos de la barra que actúan sobre el nudo medio.

En una barra elástica, las inercias debidas a los extremos se movilizarán posteriormente hasta alcanzar la inercia como sólido rígido cuando todos los puntos de la barra giren con la misma aceleración angular. Como en los casos habituales de estudio no nos vamos a encontrar con esta casuística, quizás no debería preocuparnos demasiado que la inercia como sólido rígido resulte más alta de lo debido. Posiblemente esto se traduzca en un pequeño ralentizamiento del movimiento de la estructura, pero esto ya lo estudiaremos cuando verifiquemos el comportamiento del modelo.

Por tanto, resumiendo, podemos tomar:

$f = 0,55$ $f' = 0,41$
---------------------------

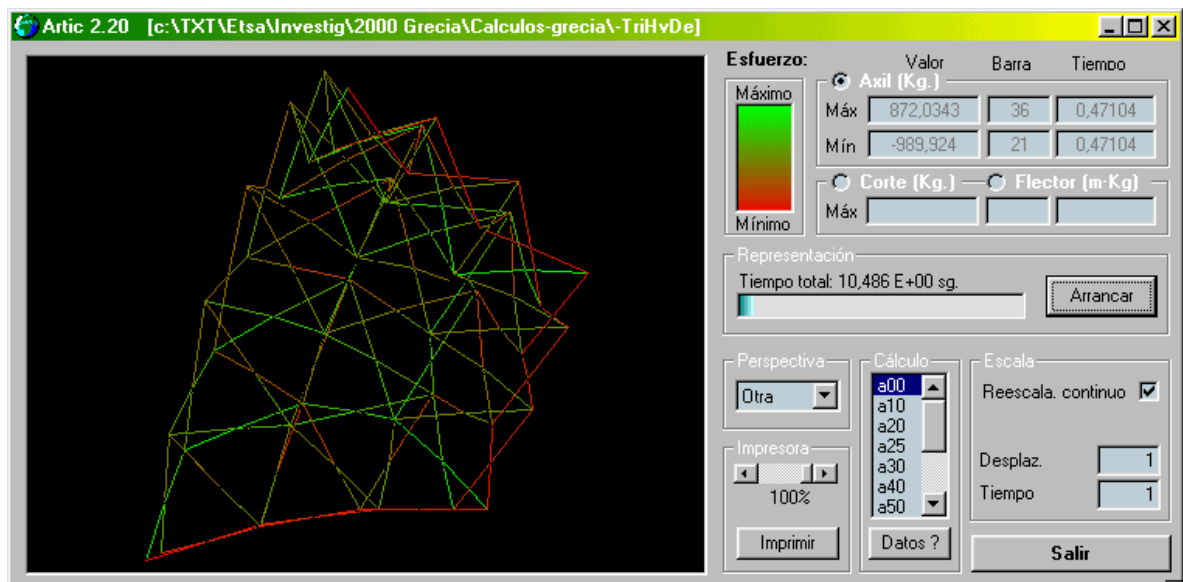
## 2.2.3 ANÁLISIS DE RESULTADOS

Una vez efectuado el cálculo por los procedimientos descritos, es preciso un estudio de los resultados arrojados para obtener algunas conclusiones.

Normalmente este tipo de cálculos derivan en cantidades ingentes de datos que luego debemos filtra para obtener exactamente lo que necesitamos en cada caso. En principio nos llegarán dos tipos de análisis.

### ■ ANÁLISIS GLOBAL.

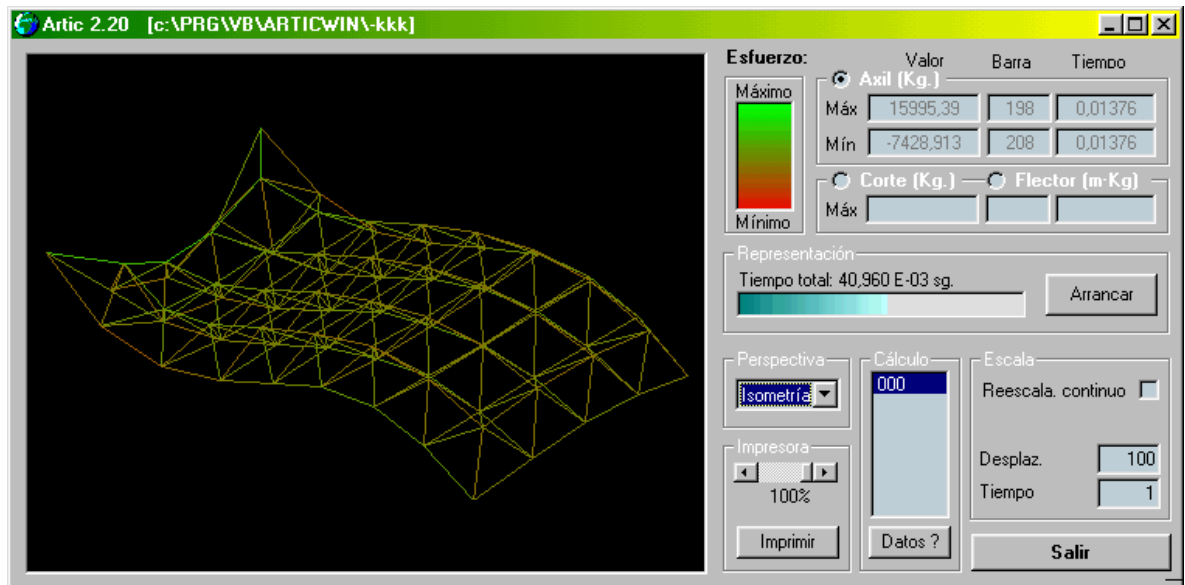
Es el tipo de análisis que más útil nos resultará para el estudio del despliegue de las mallas.



Consiste básicamente en una visión en perspectiva del despliegue de la estructura, en la cual se podrán apreciar, mediante un código de colores, que barras de la estructura se hallan más solicitadas en cada instante.

También es interesante conocer en cada período de tiempo de estudio, qué barra es la más solicitada y cuanto vale el esfuerzo que se produce en ella. Para simplificar el proceso se estudiará cada esfuerzo por separado.

Si en vez del estudio de un despliegue, estamos estudiando un tema de vibraciones, en el cual los desplazamientos serán inapreciables, nos interesará poder ver ampliados los desplazamientos, aplicando el factor de escala que nos interese a los mismos.



Vemos en la figura que precede, la forma de vibración de una malla plana (para poderlo apreciar mejor) apoyada en sus cuatro esquinas y con cargas excéntricas.

Si estamos en un caso sin amortiguamiento obtendremos un movimiento perfectamente periódico, Pero aún así no se tratará, obviamente, de ninguno de los modos fundamentales de vibración de la malla, ya que estos no dependían de la distribución de fuerzas aplicadas, sino de la forma real de vibración de la estructura.

De esta forma hemos visto un poco por encima la potencia y posibilidades de este tipo de análisis.



## ANÁLISIS LOCAL.

Definimos de esta forma el estudio concreto de algún punto o barra de la estructura. Este análisis será normalmente complementario y continuación del anterior.

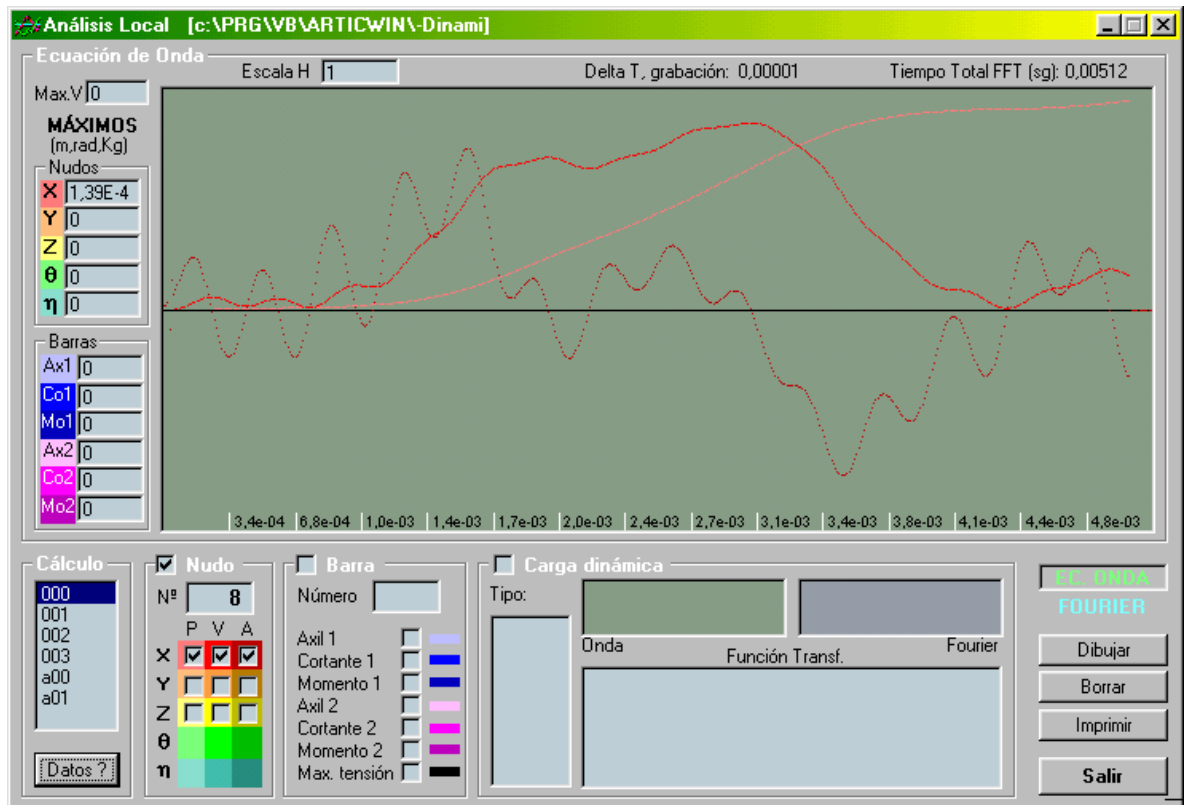
Decíamos antes, que con el análisis global podíamos saber cuales eran las barras más solicitadas de la estructura en un determinado período de tiempo. Ahora podemos analizar cada una de estas barras con todo detalle. Bastará con indicar cual es el número de barra y que esfuerzo queremos estudiar. De igual modo podemos estudiar el comportamiento de los nudos extremos y central de la barra a lo largo del tiempo. De estos la información con la que podemos trabajar será la de desplazamientos, velocidades y aceleraciones.

En este análisis tenemos dos posibilidades.

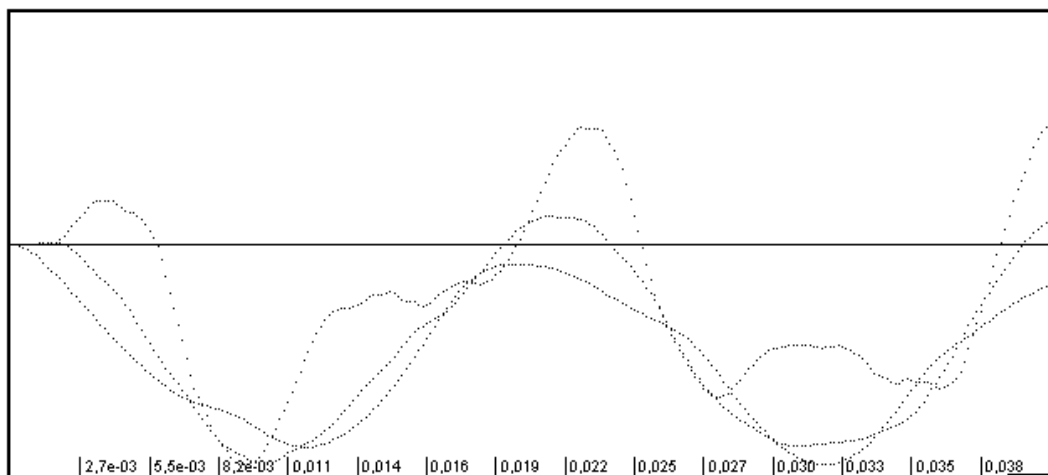
### DOMINIO DEL TIEMPO.

En primer lugar podemos ver la evolución del esfuerzo o de los desplazamientos a lo largo del tiempo, es lo que definimos como ecuación de onda, y refleja claramente la evolución del elemento de estudio a lo largo del tiempo.





Este tipo de representación no tiene mayor complicación y se reduce a la representación gráfica de una serie de valores que se han ido almacenando durante el proceso de cálculo. Tiene la ventaja de que podemos ir dibujando sucesivamente el mismo dato para sucesivas barras o nudos, con lo que se superponen los gráficos y resulta una comparativa muy cómoda si el número de gráficos que se superpone no es excesivo.

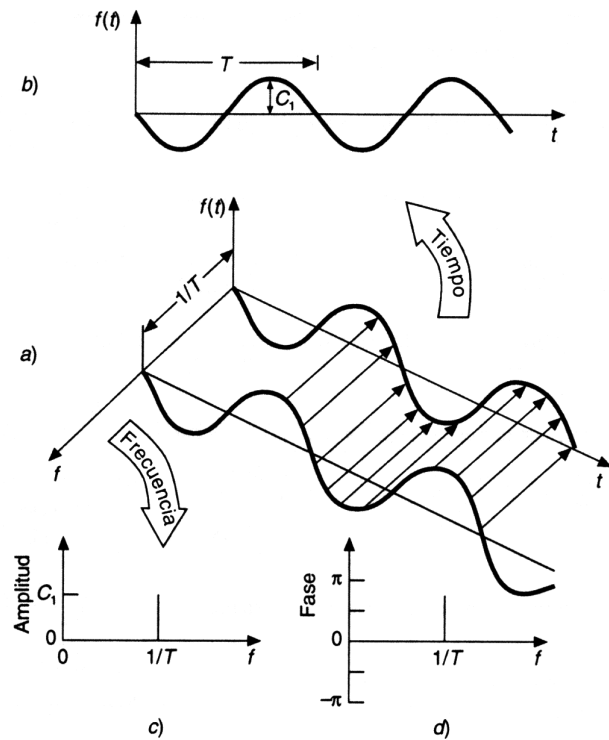


## DOMINIO DE LAS FRECUENCIAS.

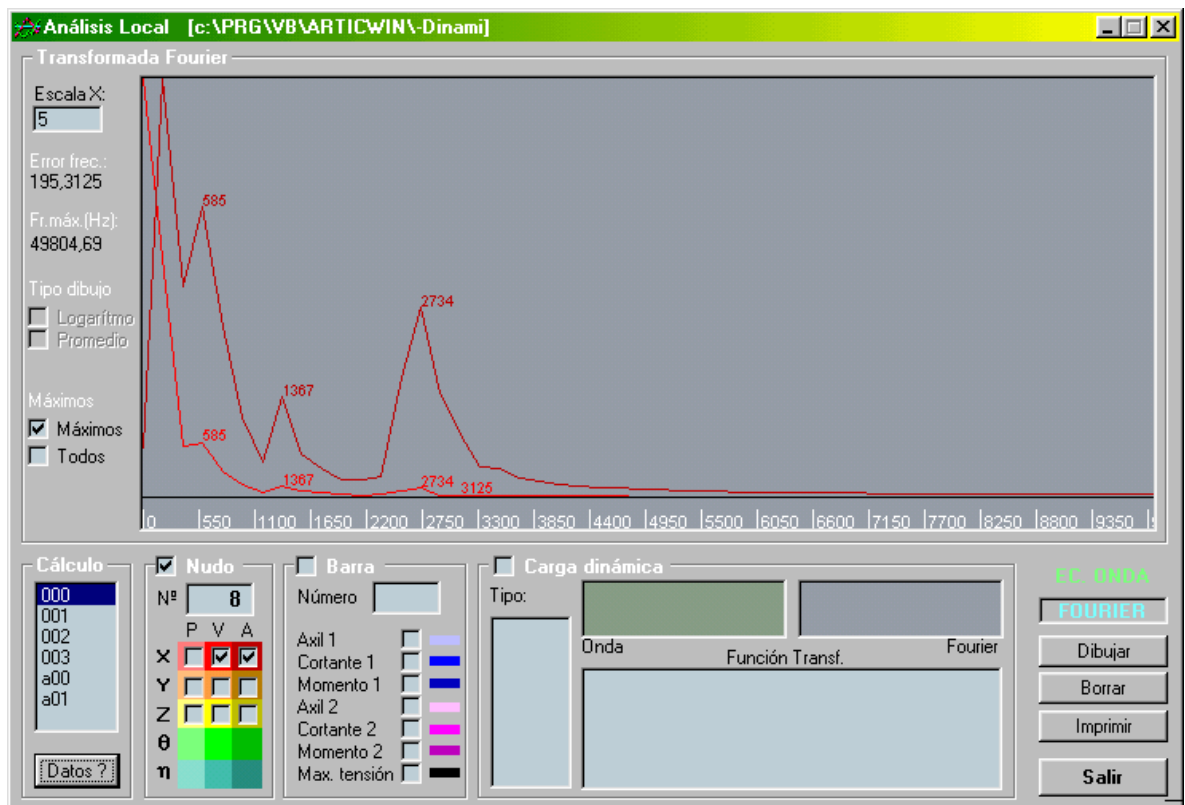
Hasta ahora hemos realizado los análisis en el dominio del tiempo sin mayores complicaciones. Ello se debe a que conceptualizar de esta forma el comportamiento de una función es cómodo y nos resulta muy intuitivo. El dominio de la frecuencia proporciona una perspectiva alternativa para caracterizar el comportamiento de funciones aleatorias.

En el gráfico adjunto vemos un ejemplo de cómo se puede dibujar un senoide en los dominios del tiempo y la frecuencia.

La proyección del tiempo se reproduce en b), la proyección de la amplitud-frecuencia se reproduce en c) y la proyección fase-frecuencia en d).



El programa desarrollado realiza este análisis en el dominio de las frecuencias mediante la Transformada Rápida de Fourier (TRF). De la que daremos unas breves nociones a continuación para su mejor entendimiento.



### Series de Fourier.

Ya es conocido, y fue demostrado por Fourier, que si tenemos una función periódica  $f(t)$ , tal que

$$f(t) = f(t+T)$$

donde  $T$  es una constante llamada período, que es el valor más pequeño para el cual es válida la expresión anterior, entonces esta función se puede representar por medio de una serie infinita de sinusoides de frecuencias relacionadas de manera armónica, de la forma [4]:

$$f(t) = a_0 + a_1 \cos(\omega t) + b_1 \text{sen}(\omega t) + a_2 \cos(2\omega t) + b_2 \text{sen}(2\omega t) + \dots \\ \dots + a_n \cos(n\omega t) + b_n \text{sen}(n\omega t) + \dots$$

o de manera más concisa

$$f(t) = a_0 + \sum_{k=1}^{\infty} [a_k \cos(k\omega t) + b_k \text{sen}(k\omega t)]$$

donde  $\omega = 2\pi / T$  es la frecuencia angular

Pudiéndose calcular los coeficientes mediante las expresiones:

$$a_0 = \frac{1}{T} \int_{t_1}^{t_1+T} f(t) dt \\ a_k = \frac{2}{T} \int_{t_1}^{t_1+T} f(t) \cdot \cos(k\omega t) dt \\ b_k = \frac{2}{T} \int_{t_1}^{t_1+T} f(t) \cdot \text{sen}(k\omega t) dt \\ \text{para } k = 1, 2, \dots$$

donde  $t_1$  se suele tomar igual a cero y  $a_0$  representa el valor medio de la función.

La forma exponencial de la serie de Fourier resulta:

$$f(t) = \sum_{k=-\infty}^{\infty} C_k \cdot e^{ik\omega t} \\ \text{con} \\ C_k = \frac{1}{T} \int_0^T f(t) \cdot e^{-ik\omega t} dt \\ \text{donde } \omega = \frac{2\pi}{T} \text{ y } k = 0, 1, 2, \dots$$

### Transformada de Fourier.

Aunque la serie de Fourier es útil para estudiar el espectro de una función periódica, existen muchas formas de onda, entre las que se hallan el despliegue de mallas, que no se autorrepiten de manera regular. Sería por tanto valioso analizar dichas formas de onda no periódica.

La transición de una función de periódica a no periódica se puede efectuar el permitir que el período tienda a infinito. En otras palabras, como  $T$  se vuelve infinito, la función misma nunca se repetirá y así se volverá no periódica. Si se permite que ocurra esto, se puede demostrar que la serie de Fourier se reduce a:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(i\omega) \cdot e^{i\omega t} \cdot dt \quad (\text{conocida como la Inversa de la Transformada de Fourier})$$

y los coeficientes se vuelven una función continua de la variable frecuencia  $\omega$ , como:

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k \cdot e^{-i\omega n} \quad (\text{función conocida como Transformada de Fourier})$$

La distinción entre serie y transformada de Fourier debe ser ahora muy clara. La serie se aplica a formas periódicas y la transformada a las no periódicas. Además la serie de Fourier convierte una función continua, de periódica en el dominio del tiempo a magnitudes en el dominio de la frecuencia con frecuencias discretas. En contraste, la transformada de Fourier convierte una función continua en el dominio del tiempo en una función continua en el dominio de la frecuencia.

### Transformada Discreta de Fourier (TDF).

En nuestro caso de estudio por diferencias finitas, como resultado no tendremos una función continua, sino un conjunto de valores discretos finitos.

Tendremos generalmente un intervalo de  $0$  a  $t$  dividido en  $N$  subintervalos iguales, de ancho  $\Delta t = T/N$ , y designamos  $f_n$  como el valor de la función en el tiempo  $t_n$ . Los datos se especificarán en  $n=0,1,2,\dots,N-1$ .

Para este caso las ecuaciones anteriores se convierten en:

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k \cdot e^{-i\omega n} \quad \text{para } n = 0, \dots, N-1$$

y

$$F(i\omega) = \int_{-\infty}^{\infty} f(t) \cdot e^{-i\omega t} \cdot dt \quad \text{para } k = 0, \dots, N-1$$

$$\text{siendo: } \omega = \frac{2\pi}{N}$$

Siendo la primera la inversa y la segunda la **Transformada Discreta de Fourier**. Aunque los cálculos podrían realizarse a mano, es una tarea laboriosa, sobre todo teniendo en cuenta el gran número de datos que manejaremos habitualmente. Por ello se implementa un algoritmo de cómputo para la TDF, que resulta relativamente sencillo:

```

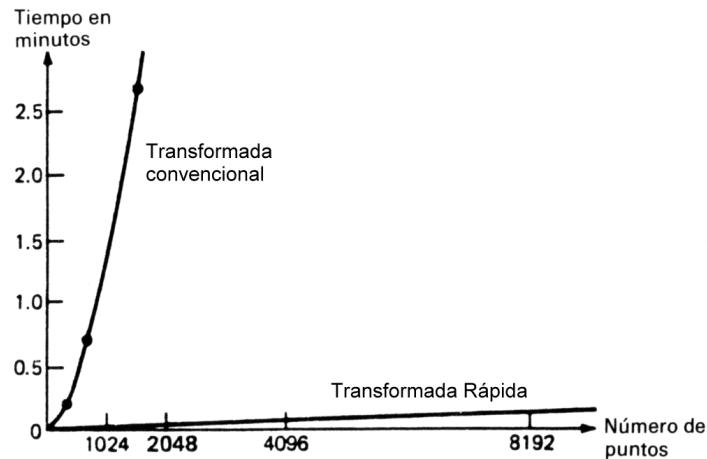
o
o For k=0 to N-1
o   For n=0 to N-1
o     angulo = k * ω * n
o     real_k = real_k + f_n * cos(angulo) / N
o     imaginario_k = imaginario_k - f_n * sen(angulo) / N
o   Next n
o Next k
o

```

### Transformada Rápida de Fourier (TRF).

Aunque el algoritmo descrito anteriormente calcula de manera adecuada la **TDF**, es laborioso por su trabajo de cómputo debido a que se requieren  $N^2$  operaciones. En consecuencia, para los datos de las muestras de tamaño moderado o alto, la determinación directa de la **TDF** puede ser en extremo lenta.

La Transformada Rápida de Fourier no es un nuevo tipo de transformación, sino un algoritmo numérico muy eficiente para evaluar la **TDF**. Su velocidad surge por el hecho de utilizar los resultados de los cálculos previos para reducir el número de operaciones. En particular, explota la periodicidad y simetría de funciones trigonométricas, reduciendo el número de operaciones a aproximadamente  $N \cdot \log_2 N$ . Puede verse la diferencia de tiempos en el gráfico siguiente.



El primer algoritmo TRF fue desarrollado por Gauss en los inicios del s.XIX. En 1965 Cooley y Tukey publicaron un artículo clave, en el cual delineaban un algoritmo para el cálculo de la TRF.

La idea básica es que una TDF de longitud  $N$  se puede 'cortar' en varias TDF sucesivamente más pequeñas. Así un cálculo de  $N$  puntos se puede dividir en dos cálculos de  $N/2$  puntos. Ya que cada uno de estos últimos requiere aproximadamente  $(N/2)^2$  resulta que ahora necesitamos  $N/2$  cálculos frente a los  $N^2$ , reduciendo el tiempo preciso a la mitad. Vemos que podemos repetir este procedimiento para cada una de las dos mitades, y así sucesivamente, hasta que no podamos efectuar más divisiones. La implementación del algoritmo puede resumirse en:

```

○
○ m = Log(N) / Log(2)
○ N2 = N
○ For k = 1 To m
○   N1 = N2
○   N2 = N2 / 2
○   angulo = 0
○   arg = 2 * pi / N1
○   For j = 0 To N2 - 1
○     c = Cos(angulo)
○     s = -Sin(angulo)
○     For i = j To N - 1 Step N1
○       kk = i + N2
○       xt = x(i) - x(kk)
○       x(i) = x(i) + x(kk)
○       yt = y(i) - y(kk)
○       y(i) = y(i) + y(kk)
○       x(kk) = xt * c - yt * s
○       y(kk) = yt * c + xt * s
○     Next i
○     angulo = (j + 1) * arg
○   Next j
○ Next k
○
○
○
○
○ j = 0
○ For i = 0 To N - 2
○   If (i < j) Then
○     xt = xj
○     xj = xi

```

```

○      xi = xt
○      yt = yj
○      yj = yi
○      yi = yt
○      End If
○      k = N / 2
○      Do
○          If (k >= j + 1) Then Exit Do
○          j = j - k
○          k = k / 2
○      Loop
○      j = j + k
○  Next i
○  For i = 0 To N - 1
○      x(i) = x(i) / N
○      y(i) = y(i) / N
○  Next i
○

```

Obsérvese que este código está compuesto de dos partes. La primera parte es básicamente la TRF, y la segunda es una rutina de fragmentos inversa para que no queden desordenados los coeficientes resultantes de Fourier.

---

## 2.2.4 GENERACIÓN DE MALLAS

---

Como decíamos, otra parte complementaria de este trabajo surgía de la necesidad disponer de modelos de mallas desplegadas con los que poder efectuar posteriormente los cálculos.

Frente a la posibilidad de usar métodos estándares, surge la idea de crear dos métodos propios, que serán más adaptados a nuestras necesidades particulares, y servirán de aportación al estudio a los numerosos métodos existentes.

La elección de los tipos a generar no es arbitraria, sino que surge del enfoque de los trabajos que hemos realizado previamente sobre este tema, optando, como criterio general por mallas en forma de cúpula por ser la tipología que vemos con más posibilidades.

Así hemos creado un modelo de generación para mallas de módulos cuadrados con la finalidad de que las incompatibilidades durante el proceso de despliegue fuesen inexistentes. Por contra, para conseguir modelos con un grado alto de incompatibilidad, optamos por el módulo triangular, del que ya hemos comentado el origen y magnitud de sus incompatibilidades.

### ■ CÚPULA DE MÓDULOS CUADRADOS.

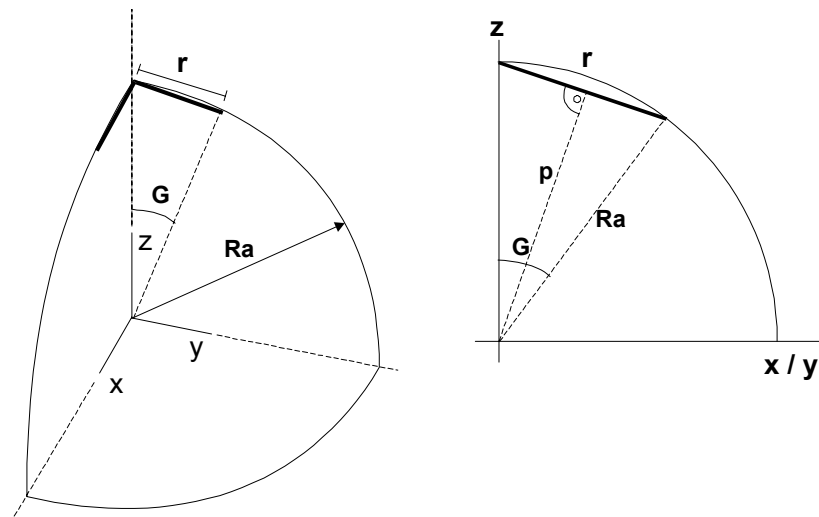
El proceso genérico para esta malla y para la siguiente es similar a los ya comentados: se define el módulo a usar y se crea una retícula base sobre la esfera a cubrir con dicho módulo.

Para esta primera cúpula varía la forma de definir la retícula base. En vez de los típicos métodos de generar la retícula en un plano y proyectarla, se optó por un sistema que resultase muy sencillo constructivamente, que es en base a que todas las barras de la estructura sean exactamente iguales.

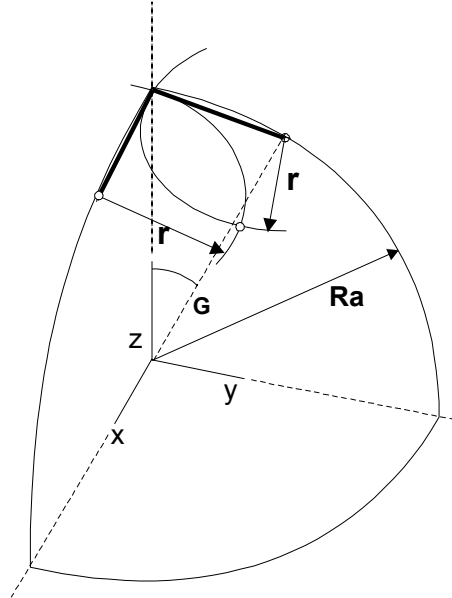
Esto es factible y relativamente sencillo, basta con conseguir que la retícula base trazada sobre la superficie esférica también tenga todos los lados de los módulos iguales.

El proceso consiste en partir de un punto, que en nuestro caso es el polo y a partir de ahí ir disponiendo las barras de lados iguales hasta alcanzar el número de módulos deseado. Veamos el proceso para un cuarto de la cúpula, luego por simetría será fácil trazar el resto de la cúpula.

Si llamamos **Ra** al radio de la cúpula y **r** al lado constante de la malla base, a partir de éstos es fácil determinar el ángulo de cobertura correspondiente **G**. Repitiendo éste sobre los círculos máximos contenidos en los planos coordenados **yz** y **xz** obtenemos de modo inmediato los vértices de la retícula para estos planos.

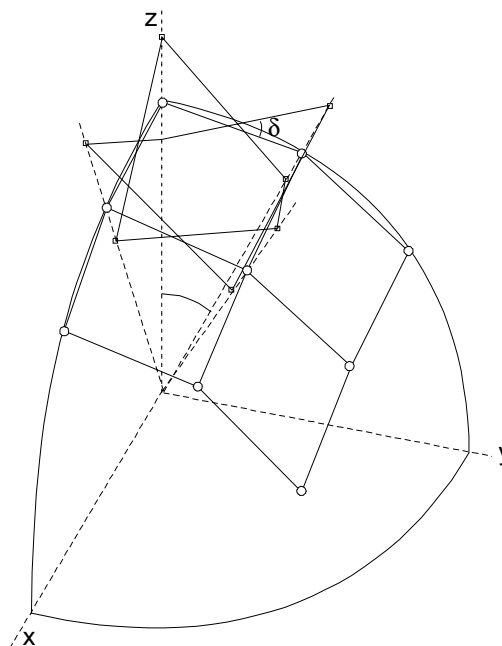


Los puntos que entrañan mayor dificultad será los intermedios. Para hallarlos, geoméricamente bastaría con trazar a partir de dos puntos conocidos e inmediatos al buscado, dos circunferencias sobre la esfera, con centro en estos puntos y con radio  $r$ . La intersección de estas circunferencias nos darían dos puntos, uno ya conocido, y el otro es el punto buscado.



Matemáticamente este proceso se traduce en hallar la intersección de tres esferas en el espacio: La esfera principal, de radio  $Ra$  y dos pequeñas de radio  $r$ . Repetimos este proceso sucesivamente hasta completar el tamaño de malla deseado.

Este proceso nos garantiza efectivamente que todos los lados tienen la misma dimensión pero presenta el inconveniente de que a medida que nos alejamos del polo y de los planos coordenados, los cuadrados se van deformando cada vez más, dando lugar a rombos pronunciados, lo cual no resulta demasiado estético.

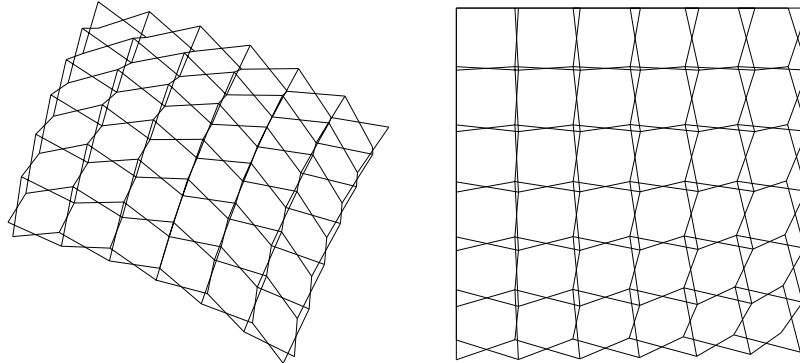


A partir de aquí el proceso ya es conocido, con la salvedad de que no es preciso calcular el punto de cruce de las espas, ya que siempre estará en el punto medio. Se define entonces el ángulo de



apertura  $\delta$ , y a partir del mismo ya se pueden hallar los nudos superiores e inferiores e ir trazando las aspas como en los métodos habituales.

En las imágenes siguientes vemos el resultado de la generación para un cuarto de cúpula.



A continuación vemos la rutina que realiza el trazado de la malla base, que como vemos resulta bastante sencilla.

```

○
○ ' Variables usadas.....
○ ' Ra - radio de la esfera en m.
○ ' d - diámetro en m.
○ ' r - lado de cada una de las barras de la cuadrícula base en m.
○ ' G - ángulo, en radianes, que se corresponde con la longitud r.
○ ' n1 - n° de divisiones de la semimalla en dirección X
○ ' n2 - n° de divisiones de la semimalla en dirección Y
○ ' .....
○
○ d = 2 * Ra
○ p = Sqr((Ra * Ra) - (R * R / 4))
○ G = 2 * Atn(R / (p * 2))
○
○ 'CREACION DE LA MALLA BASE DE 1 CAPA (1/4).....
○ np = (n1 + 1) * (n2 + 1)
○ ReDim C(np, 3)
○ ne = n1 * (n2 + 1) + n2 * (n1 + 1)
○ ReDim L(ne, 3)
○
○ 'Generación de nudos.....
○ num = 1
○ For j = 0 To n2: For i = 0 To n1
○   If i = 0 Or j = 0 Then
○     C(num, 1) = Ra * Sin(G * i)
○     C(num, 2) = Ra * Sin(G * j)
○     C(num, 3) = Ra * Cos(G * (i + j))
○   Else
○     P1 = num - n1 - 1 '.....
○     P2 = num - 1
○     Intersec P1, P2, Bx, By, Bz
○     C(num, 1) = Bx
○     C(num, 2) = By
○     C(num, 3) = Bz
○   End If
○   num = num + 1
○ Next i: Next j
○
○ 'Generación de barras.....
○ num = 1
○ For j = 1 To n2 + 1: For i = 1 To n1
○   L(num, 1) = i + ((j - 1) * (n1 + 1))
○   L(num, 2) = i + ((j - 1) * (n1 + 1)) + 1
○   num = num + 1
○ Next i: Next j
○ For j = 1 To n2: For i = 1 To n1 + 1
○   L(num, 1) = i + ((j - 1) * (n1 + 1))
○   L(num, 2) = i + ((j - 1) * (n1 + 1)) + n1 + 1
○   num = num + 1
○ Next i: Next j
○

```

La parte quizás un poco más complicada es la subrutina que calcula la intersección de las tres esferas en el espacio, que es la siguiente:

```

○
○ ' Variables usadas.....
○ ' P1,P2 - puntos a partir de los cuales se hallan los equidistantes.
○ ' M - punto medio de P1-P2
○ ' w - semi ngulo de los vectores OP1-OP2
○ ' MI - distancia M-I
○ ' V - vector asociado a MI
○ ' A - punto de intersección buscado
○ ' Be - semiángulo entre las aspas
○ ' .....
○
○ Sub Intersec(P1, P2, Bx, By, Bz)
○ Dim x1, y1, z1, x2, y2, z2, x3, y3, z3, Dx, Dy, Dz, Mx, My, Mz, M, rp, cow, MI, dis
○ Dim v, Vx, Vy, Vz, Ux, Uy, Uz, Mlx, Mly, Mlz, Iax, Iay, Iaz, Ibx, Iby, Ibz, r0, kk
○ x1 = C(P1, 1)
○ y1 = C(P1, 2)
○ z1 = C(P1, 3)
○ x2 = C(P2, 1)
○ y2 = C(P2, 2)
○ z2 = C(P2, 3)
○ Dx = (x2 - x1)
○ Dy = (y2 - y1)
○ Dz = (z2 - z1)
○ dis = Sqr(Dx * Dx + Dy * Dy + Dz * Dz)
○ Mx = (x2 + x1) / 2 'punto medio.
○ My = (y2 + y1) / 2
○ Mz = (z2 + z1) / 2
○ M = Sqr(Mx * Mx + My * My + Mz * Mz)
○ kk = R * R - (dis * dis / 4)
○ If kk < 0 Then MsgBox "Los dos puntos se hallan demasiado alejados", vbOKOnly: Stop
○ rp = Sqr(kk) ' obtenemos r'
○ If (Ra - M) > rp Then MsgBox "No hay solución sobre la esfera", vbOKOnly: Stop
○ cow = (Ra * Ra + M * M - rp * rp) / (2 * Ra * M)
○ MI = M * Sqr(1 / (cow * cow) - 1)
○ Vx = My * Dz - Mz * Dy ' V ortogonal al plano OM,P1,P2
○ Vy = Mz * Dx - Mx * Dz
○ Vz = Mx * Dy - My * Dx
○ v = Sqr(Vx * Vx + Vy * Vy + Vz * Vz)
○ Ux = Vx / v
○ Uy = Vy / v
○ Uz = Vz / v
○ Mlx = MI * Ux
○ Mly = MI * Uy
○ Mlz = MI * Uz
○ Iax = Mx + Mlx ' solución 1...
○ Iay = My + Mly
○ Iaz = Mz + Mlz
○ Ibx = Mx - Mlx ' solución 2...
○ Iby = My - Mly
○ Ibz = Mz - Mlz
○ r0 = Sqr(Ibx * Ibx + Iby * Iby + Ibz * Ibz)
○ Bx = Ra * Ibx / r0
○ By = Ra * Iby / r0
○ Bz = Ra * Ibz / r0
○
○ End Sub

```

Una vez tenemos la malla base, la rutina que crea los nudos y aspas definitivos es la siguiente

```

○
○ ' Y vamos a pasar de la malla de una capa ya calculada a la generación
○ ' de las aspas y a almacenar los datos de barras en una matriz compatible
○ ' para su cálculo LD(ned,4) así como los nudos en CD(npd,3)
○
○
○ nx = ne ' nudos en el cruce de las aspas (se calculan en barras)''''''''''''''''''''
○ npd = np * 2 + nx
○ ReDim Cd(npd, 3)
○ ned = ne * 2
○ ReDim Ld(ned, 4)
○ ReDim Enlace(ne, 5) 'barral,barra2''''''''''''''''''''
○
○
○ ' Cálculo de la1 y la2
○ P1 = 1: P2 = 2
○ x1 = C(P1, 1): y1 = C(P1, 2): z1 = C(P1, 3)
○ x2 = C(P2, 1): y2 = C(P2, 2): z2 = C(P2, 3)
○ Dx = Abs(x2 - x1)

```



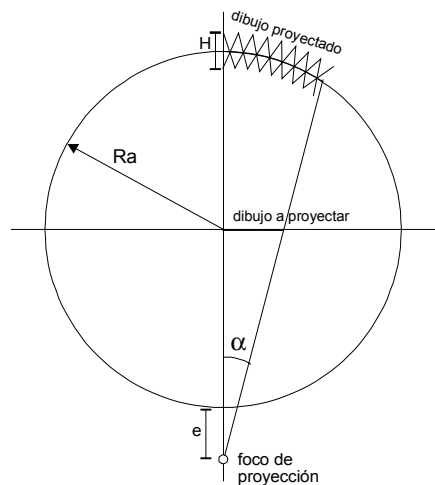
## ■ CÚPULA DE MÓDULOS TRIANGULARES.

Como ya hemos comentado, el otro tipo de malla que necesitábamos para los cálculos era una malla con bastantes incompatibilidades, y por ello optamos por la de módulo triangular.

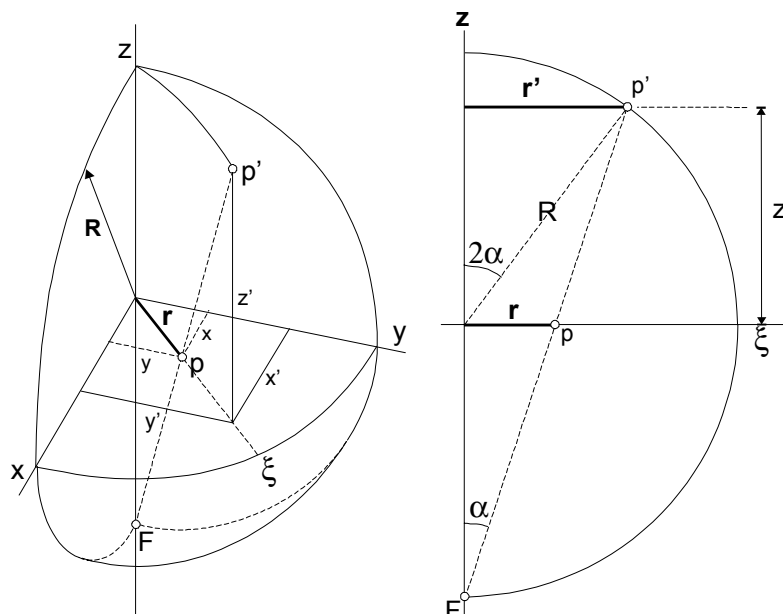
El proceso inicialmente es similar a los procesos estándares:

En primer lugar definimos la malla plana a usar y la proyectamos sobre la superficie esférica. El plano donde esté dibujada la malla no es significativo, y solo afectará a la escala de la proyección, de la misma forma que lo haría el variar el tamaño de este módulo base.

El foco de proyección sí es significativo, pues influirá en que los distintos módulos sean más o menos similares. Si los módulos son muy dispares será más difícil que cumplan las condiciones de plegado y desplegado. Normalmente dan buenos resultados focos situados cerca del polo opuesto a centro de la malla proyectada.



Si planteamos la proyección desde el polo sur de la esfera, el esquema de trabajo será el siguiente:



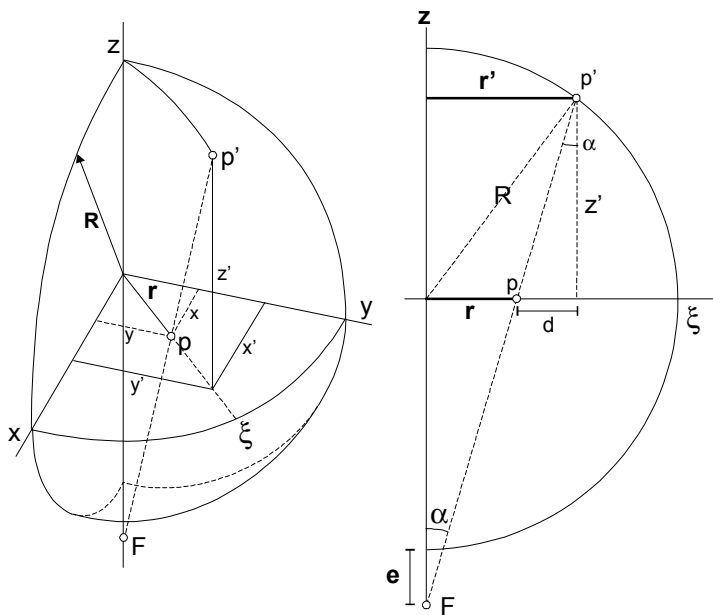
y las ecuaciones resultantes son relativamente sencillas, si ponemos en relación  $r$  y  $r'$  mediante una sencilla semejanza de triángulos.:

$$\frac{r'}{r} = \frac{R + R \cdot \cos(2\alpha)}{R} = 1 + \cos(2\alpha)$$

entonces :

$$(x, y, 0) \Rightarrow \begin{cases} x' = x \cdot \frac{r'}{r} = x \cdot [1 + \cos(2\alpha)] \\ y' = y \cdot \frac{r'}{r} = y \cdot [1 + \cos(2\alpha)] \\ z' = R \cdot \cos(2\alpha) \end{cases}$$

Pero ya hemos dicho que no siempre el polo es el Foco de proyección más adecuado, sino muchas veces un punto por debajo de él. En dicho caso el esquema de proyección con el que nos encontramos ahora es:



Las ecuaciones ahora se complican un poco más. Tenemos como datos el radio de la esfera  $R$ , la posición del punto a proyectar  $(x, y, 0)$  y por tanto  $r$ , y la excentricidad  $e$  del foco de proyección. Nos quedan como incógnitas básicas  $r'$  (a partir de la cual ya obtenemos  $x'$ ,  $y'$ ) y la cota  $z'$ .

$$\left. \begin{aligned} r' = r + d = r + z' \cdot \operatorname{tg}(\alpha) = r + z' \cdot \frac{r}{R + e} \\ p' \in R \Rightarrow R^2 = z'^2 + r'^2 \Rightarrow z' = \sqrt{R^2 - r'^2} \end{aligned} \right\} \Rightarrow r' = r + \sqrt{R^2 - r'^2} \cdot \frac{r}{R + e}$$

$$\frac{r' - r}{\sqrt{R^2 - r'^2}} = \frac{r}{R + e} \Rightarrow \dots \underbrace{r'^2 [(R + e)^2 + r^2]}_A - \underbrace{r' [2r(R + e)^2]}_B + \underbrace{[r^2 (R + e)^2 + r^2 R^2]}_C = 0$$

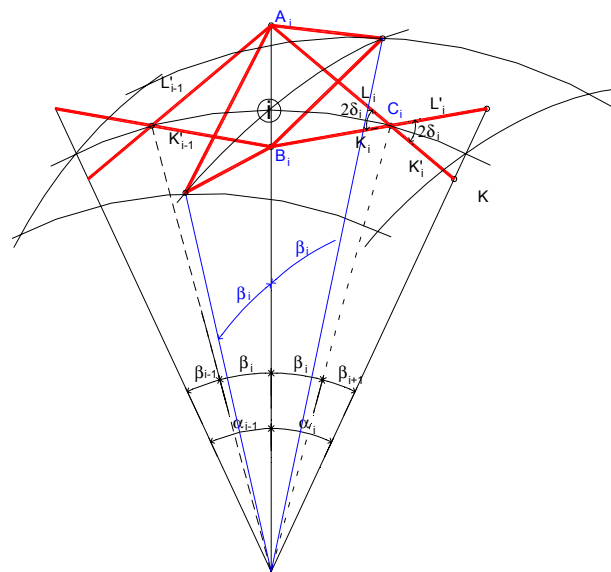
Resolviendo la ecuación de 2º grado resultante obtenemos las dos posibles soluciones para  $r'$ . Obviamente, para los casos habituales, tomaremos el valor mayor, que es la intersección del rayo de proyección con el casquete superior (el valor más pequeño lo es con el casquete inferior). Una vez determinado este valor, las coordenadas de  $p'$  ( $x'$ ,  $y'$ ,  $z'$ ) ya son inmediatas:

$$\begin{cases} x' = x \cdot \frac{r'}{r} \\ y' = y \cdot \frac{r'}{r} \\ z' = \sqrt{R^2 - r'^2} \end{cases}$$

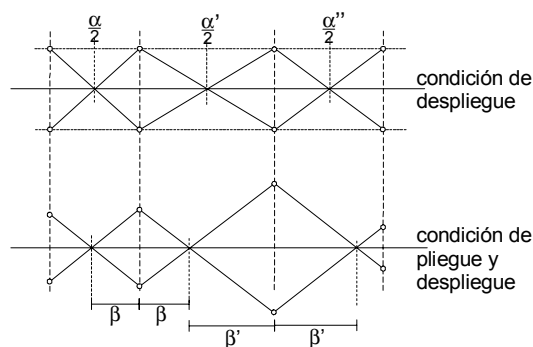
Una vez que tenemos la malla plana proyectada sobre la esfera debemos hallar los puntos **C**, que estarán en un lugar intermedio de estos segmentos circulares cubiertos por un ángulo que llamamos  $\alpha_i$  que serán datos.

Y aquí es donde planteamos una alternativa al sistema habitual, planteando la posibilidad que el canto de la malla sea variable.

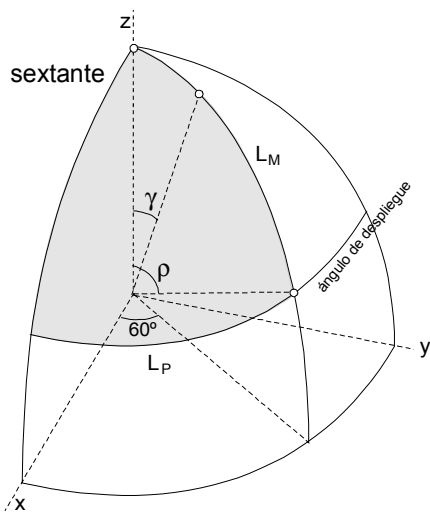
Lo normal es que planteemos la malla en su posición final desplegada y por la condición de plegabilidad sabemos que todos los  $\beta_i$  que concurren en un nudo  $i$  deben ser iguales, de modo que podemos plantear el sistema de ecuaciones:  $\alpha_i = \beta_i + \beta_{i+1}$ , con tantas ecuaciones como segmentos  $\alpha_i$  tengamos y tantas incógnitas  $\beta_i$  como vértices  $i$  tengamos. Resolvemos es sistema de ecuaciones, que en este caso será incompatible, intentando optimizar la solución en la medida de lo posible, y obtenemos los ángulos  $\beta_i$  y por tanto los puntos de cruce **C**.



Pero en vez de efectuar el planteamiento anterior podemos realizar justo el inverso: Generamos la malla en la posición plegada o muy próxima al plegado y por contra planteamos la condición de desplegado.



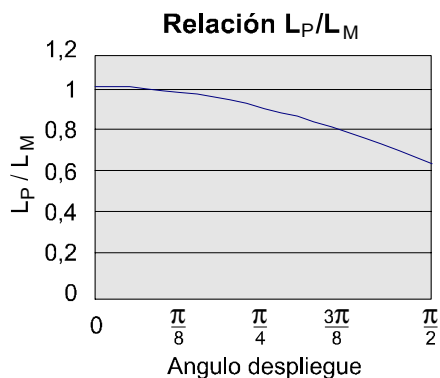
Vemos que para que se cumpla el despliegue basta con tomar los puntos **C**, entre los dos puntos de la malla plana proyectada. Esta condición bastaría para un arco, pero no para una cúpula por su carácter tridimensional. Esto lo podemos ver con el siguiente gráfico, en el cual, como primera aproximación, aunque no sea correcta, suponemos que las aspas despliegan según paralelos y meridianos perfectos:



Si estudiamos el comportamiento de un sexto de cúpula, y consideramos la misma desplegada un ángulo  $\rho$ , la longitud según un meridiano  $L_M$  y según un paralelo  $L_P$ , son sustancialmente distintas:

$$\left. \begin{aligned} L_P &= \frac{2\pi}{6} R \cdot \sin \rho \\ L_M &= \rho R \end{aligned} \right\} \begin{aligned} \frac{L_P}{L_M} &= \frac{\pi \cdot \sin \rho}{3 \cdot \rho} \end{aligned}$$

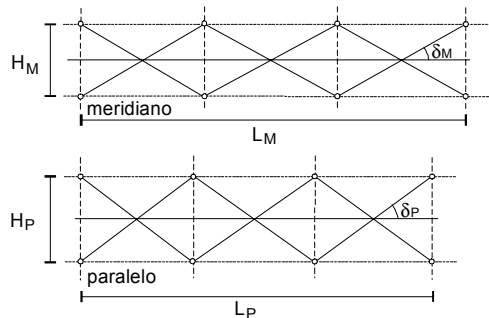
La relación entre estos valores la podemos ver en el gráfico siguiente:



siendo el límite para  $0^\circ$  el valor: 1,0471975

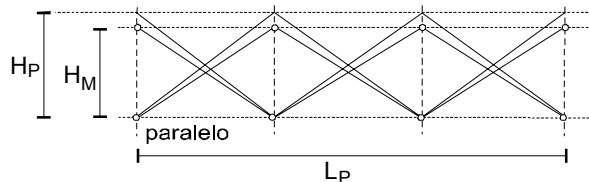
Pero en esta tipología de malla, según ambas líneas tendremos el mismo número de aspas, y si hemos generado la malla en una posición próxima al plegado, con un canto  $H$  constante, obtendríamos barras de aproximadamente la misma longitud, y nos ocurriría al desplegar lo que veremos a continuación.

Usando un esquema simplificado como si fuesen recorridos rectos para que resulte más sencilla la visualización, tendríamos:



Como al desplegar según el meridiano tienen que recorrer mayor distancia que sobre el paralelo, implica que el canto según el meridiano ( $H_M$ ) será menor que según un paralelo ( $H_P$ ). Por tanto en el encuentro de ambas aspas los nudos superior e inferior no nos coincidirían.

Para solucionar esto tenemos que plantear esta nueva condición, que  $H_P=H_M$ , pero no solo en los extremos del sextante, sino también en los haces de aspas que forman los paralelos intermedios (que llamaremos filas). En principio bastará con variar la altura del paralelo para hacerla igual a la del meridiano y listo.



Pero este cambio tiene más implicaciones, ya que esta malla que desplegada estaría perfecta, no se podría plegar porque las barras que confluyen en un nudo intersección de un meridiano y un paralelo tendrían diferentes longitudes. Ello implica, como generaremos la malla próxima a la posición de plegado, que tendremos que ajustar las longitudes de las barras de cada meridiano a las de las distintas filas por las que va pasando, lo que se traduce en dos condiciones:

$$\left. \begin{array}{l} H_P = H_M \\ b_P = b_M \end{array} \right\} \text{ siendo } b \text{ las longitudes de las barras}$$

entonces:

$$\left. \begin{array}{l} \text{sen}(\delta_M) = \frac{H_M}{b_M} \\ \text{sen}(\delta_P) = \frac{H_P}{b_P} \end{array} \right\} \Rightarrow \text{sen}(\delta_M) = \text{sen}(\delta_P) \Rightarrow \delta_M = \delta_P$$

Pero esto no puede cumplirse, ya que implicaría que  $L_P=L_M$ , y no es así, por tanto vemos que es por aquí por donde surgen los problemas de compatibilidad. Como solución aproximada, aunque los ángulos de apertura no sean iguales, vamos a suponer que son bastante similares, o sea:

$$\delta_M \approx \delta_P$$

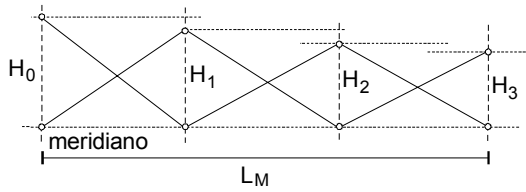
$$\left. \begin{array}{l} \text{tg}(\delta_M) = \frac{H_M}{L_M/n} \\ \text{tg}(\delta_P) = \frac{H_P}{L_P/n} \end{array} \right\} \Rightarrow \text{tg}(\delta_M) = \text{tg}(\delta_P) \Rightarrow \frac{H_M}{L_M} = \frac{H_P}{L_P} \Rightarrow H_P = \frac{L_P}{L_M} \cdot H_M$$

$$H_P = \frac{\pi \cdot \text{sen} \rho}{3 \cdot \rho} \cdot H_M$$

Significa que la relación entre longitudes de barras puede seguir una variación similar a la que vimos antes para la relación  $L_P / L_M$

Por tanto el canto de los paralelos irá reduciéndose según esa relación, y en los meridianos ocurrirá:





Vemos que implica que el canto debe ser variable. Como generamos en la posición de plegado el canto debe ser igual a las longitudes de las barras, pero podemos ver que para estas, la relación sigue siendo la misma.

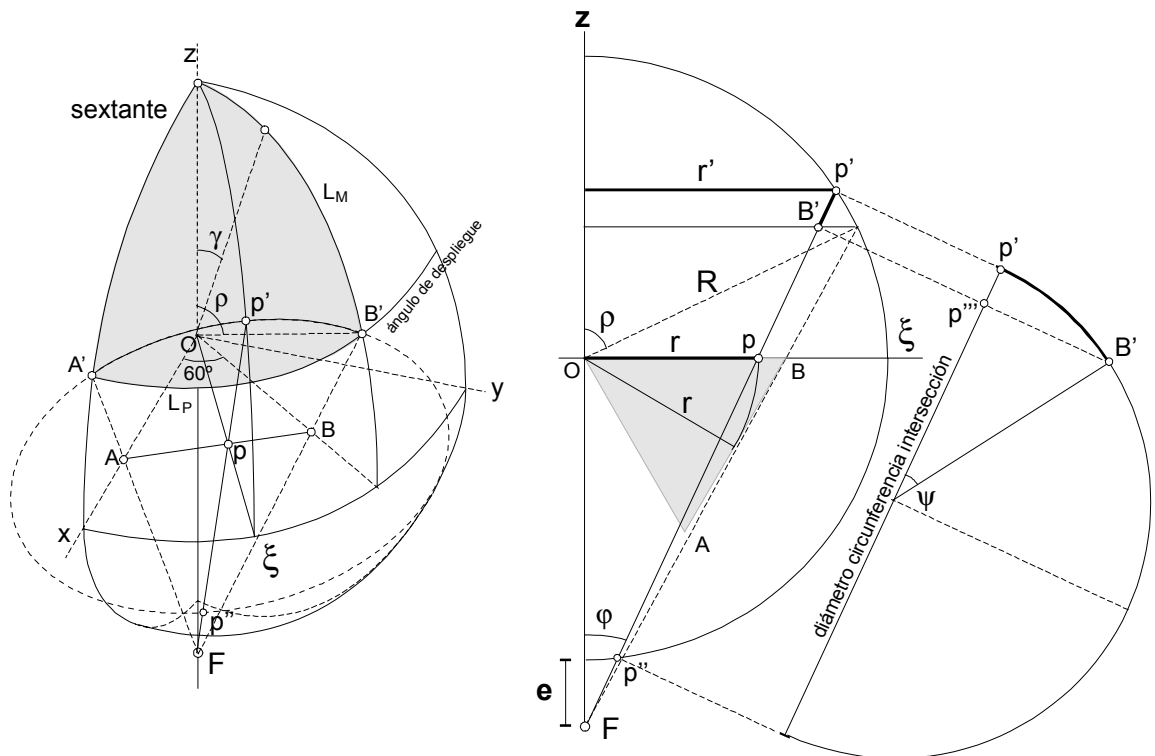
$$\left. \begin{aligned} \delta_M &\approx \delta_P \\ \text{sen}(\delta_M) &= \frac{H_M}{b_M} \\ \text{sen}(\delta_P) &= \frac{H_P}{b_P} \end{aligned} \right\} \Rightarrow \text{sen}(\delta_M) \approx \text{sen}(\delta_P) \Rightarrow \frac{H_M}{b_M} = \frac{H_P}{b_P} \Rightarrow \frac{b_P}{b_M} = \frac{L_P}{L_M}$$

La rutina para determinar un canto variable resulta sencilla:

```

○ '-----
○   Hm = H / 1.047197551
○   Roi = Ro / Orden * Fila
○   Hi = (pi / (3 * Roi) * Sin(Roi)) * Hm
○ '-----
  
```

En realidad, si somos un poco más estrictos, no resulta tan sencilla, ya que las filas de aspas no van exactamente por lo paralelos, sino que irán según arcos de circunferencia tipo el **A'p'B'**, que resultan de la proyección del segmento **AB** en la posición desplegada:



El cálculo de esta longitud parte de hallar el diámetro de la circunferencia intersección:  $p'p''$ , cuestión que ya resolvimos cuando hallábamos la proyección de un punto. Dadas las coordenadas de un punto a proyectar  $p$  obteníamos dos soluciones de  $r'$  y por tanto los puntos  $p'$  y  $p''$ .

$$r'^2 \underbrace{[(R+e)^2 + r^2]}_A - r' \cdot \underbrace{[2r(R+e)^2]}_B + \underbrace{[r^2 \cdot (R+e)^2 + r^2 \cdot R^2]}_C = 0$$

$$p' \begin{cases} x' = x \cdot \frac{r'}{r} \\ y' = y \cdot \frac{r'}{r} \\ z' = \sqrt{R^2 - r'^2} \end{cases} \quad p'' \begin{cases} x'' = x \cdot \frac{r''}{r} \\ y'' = y \cdot \frac{r''}{r} \\ z'' = \sqrt{R^2 - r''^2} \end{cases}$$

La distancia  $p'p''$  es el diámetro de la circunferencia. La relación entre  $r$  y el segmento  $AB$  es fácil de hallar, pues  $OAB$  es un triángulo equilátero, siendo  $r$  su altura. El arco que buscamos será el doble del  $\psi$ , que va de  $B'$  a  $p'$ , supuesto  $p'$  en el centro del arco.

Por tanto tenemos un arco cuyo ángulo podemos obtener de la relación:

$$\text{sen}(\psi) = \frac{B'p''}{\frac{p'p''}{2}} = \frac{\frac{A'B'}{2}}{\frac{p'p''}{2}} = \frac{A'B'}{p'p''} \Rightarrow \overset{A'}{\square} p'B' = \frac{p'p''}{2} \cdot \text{arcsen}(\psi)$$

Obteniendo  $A'$  y  $B'$  de la proyección de  $A$  y  $B$ , por el sistema ya conocido.

A continuación vemos como resulta la rutina de generación de la malla desplegable de módulos triangulares y canto variable:

```

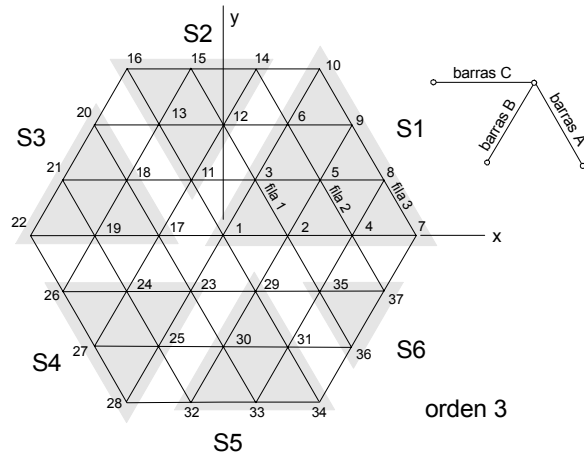
○ '-----
○ Dim C() As Single 'NP coordenadas nudos
○ Dim L() As Long 'NE barras: nudo inicio y fin
○ Dim LA() As Long 'barras de enlace: barra inicio, barra fin
○ Dim CA() As Single 'coordenadas nudos
○ '-----
○ Dim Ra As Single 'radio esfera
○ Dim Gamma As Single 'angulo de generación
○ Dim Ro As Single 'angulo total de despliegue deseado
○ Dim e As Single 'excetricidad foco
○ Dim H As Single 'canto malla
○ Dim Hvar As Boolean 'si es canto variable
○ Dim Ord As Single 'orden malla (n° módulos)
○ Dim Sext As Single 'n° de sextantes a crear: 1 a 6
○ '-----
○
○ 'lado triángulo en plano XY
○ lado = (Ra * Sin(Gamma)) / (Ra * Cos(Gamma) + Ra + e) * (Ra + e) / Ord
○
○ np = NPsext(Sext)
○ ne = NEsext(Sext)
○ nea = ne / 2
○
○ ReDim C(np, 3) 'NP coordenadas nudos
○ ReDim L(ne, 2) 'NE barras: nudo inicio y fin
○ ReDim LA(ne, 2) 'barras de enlace: barra inicio, barra fin
○ ReDim CA(ne * 2, 3) 'coordenadas nudos
○
○ '-----
○ 'arrancamos
○

```

```

O num = 0: Nbarr = 0
O numA = 0: nbarrasA = 0
O
O 'nudos centrales
O x1 = 0: y1 = 0: z1 = Ra + H / 2
O x2 = 0: y2 = 0: z2 = Ra - H / 2
O otronudo x1, y1, z1: otronudo x1, y2, z2
O
O For S = 1 To Sext: For Fila = 1 To Ord: For Nudo = 1 To Fila + 1
O
O 'Generació de nudos.....
O
O If Not (S > 1 And Nudo = 1) Then
O If Not (S = 6 And Nudo = Fila + 1) Then
O x = lado * Fila - lado * (Nudo - 1) * Cos(pi / 3)
O y = lado * (Nudo - 1) * Sin(pi / 3)
O R = Sqr(x * x + y * y)
O Fi = Atn(y / x)
O Fi = Fi + pi / 3 * (S - 1)
O x0 = R * Cos(Fi)
O y0 = R * Sin(Fi)
O Proyecta Ra, e, H, Hvar, Ro, Fila, Ord, x0, y0, x1, y1, z1, x2, y2, z2
O otronudo x1, y1, z1: otronudo x2, y2, z2
O End If
O End If
O
O

```



```

O
O
O 'Generación de barras.....
O
O 'Barras A.....
O If Nudo > 1 Then
O If Nudo = 2 And S > 1 Then 'las primeras de los sucesivos sextantes
O If S = 6 Then
O If Fila = 1 Then 'la 1ª del último sextante
O '-----
O n = NPsext(4)
O a = 4: b = n + 1: OtraBarra a, b
O a = 3: b = n + 2: OtraBarra a, b
O 'calcula el punto de cruce de las dos últimas barras y crea el enlace.
O CruceBarr
O Else 'el resto de 1º nudo del último sextante
O n = NPsext(4) + (FactS(Fila)) * 2
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n: OtraBarra a, b
O 'calcula el punto de cruce de las dos últimas barras y crea el enlace.
O CruceBarr
O End If
O Else
O n = NPsext(S - 2)
O If S = 2 Then n = n + FactS(Fila + 1) * 2 Else n = n + FactS(Fila) * 2
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n - 0: OtraBarra a, b
O CruceBarr
O End If
O Else
O If Nudo = Fila + 1 And S = 6 Then 'las últimas A varia su nodo
O '-----
O n = NPsext(5) + FactS(Fila - 1) * 2
O

```

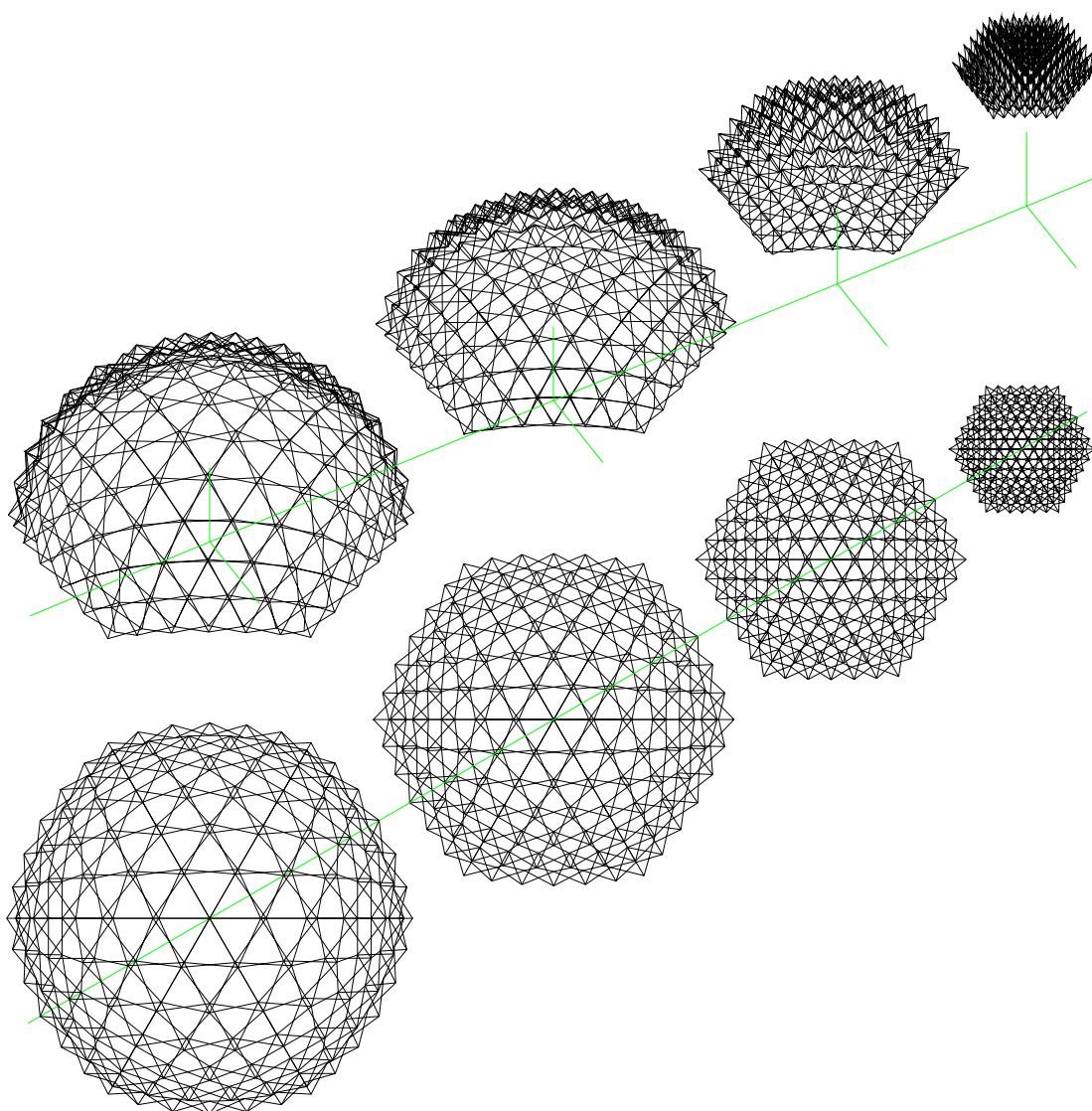
```

O      a = FactS(Fila) * 2 + 2: b = n - 1: OtraBarra a, b
O      a = FactS(Fila) * 2 + 1: b = n - 0: OtraBarra a, b
O      CruceBarr
O      Else 'caso general
O      a = num: b = num - 3: OtraBarra a, b
O      a = num - 1: b = num - 2: OtraBarra a, b
O      CruceBarr
O      End If
O      End If
O      End If
O
O      'Barras B.....
O      If Nudo > 1 Then
O      If Nudo = 2 And S > 1 Then 'las primeras de los sucesivos sextantes
O      If Fila = 1 Then
O      If S = 6 Then
O      'no hace nada
O      Else
O      a = num:      b = 1: OtraBarra a, b
O      a = num - 1: b = 2: OtraBarra a, b
O      CruceBarr
O      End If
O      Else
O      n = NPsext(S - 2)
O      If S = 2 Then n = n + FactS(Fila) * 2 Else n = n + FactS(Fila - 1) * 2
O      a = num:      b = n - 1: OtraBarra a, b
O      a = num - 1: b = n - 0: OtraBarra a, b
O      CruceBarr
O      End If
O      Else
O      If Nudo = Fila + 1 And S = 6 Then 'las últimas B ya existen
O      'no hace nada
O      Else 'caso general
O      Select Case S
O      Case 1:      n = num - Fila * 2 - 2
O      Case 2 To 5: n = num - (Fila - 1) * 2 - 2
O      Case 6:      n = num - (Fila - 1) * 2
O      End Select
O      a = num:      b = n - 1: OtraBarra a, b
O      a = num - 1: b = n - 0: OtraBarra a, b
O      CruceBarr
O      End If
O      End If
O      End If
O
O      'Barras C.....
O      If Nudo < (Fila + 1) Then
O      If Nudo = 1 And S > 1 Then 'las 1*s de los sucesivos sextantes ya están creadas
O      'no hace nada
O      Else 'caso general
O      Select Case S
O      Case 1:      n = num - Fila * 2
O      Case 2 To 5: n = num - (Fila - 1) * 2
O      Case 6: If Nudo < Fila Then n = num - (Fila - 2) * 2 Else n = (FactS(Fila - 1) + 1) * 2
O      End Select
O      a = num:      b = n - 1: OtraBarra a, b
O      a = num - 1: b = n - 0: OtraBarra a, b
O      CruceBarr
O      End If
O      End If
O
O      Next Nudo: Next Fila: Next S
O

```

A continuación vemos la malla que se genera con esta rutina, usando diversos ángulos de generación, lo que nos da una buena idea del aspecto de las cúpulas que puede generar esta rutina. Como se puede apreciar, el efecto del canto variable apenas es perceptible, e incluso puede ser necesario desde un punto de vista estético, en el sentido de que en la zona próxima al polo, el proceso de proyección origina triángulos de mayores dimensiones que en los bordes, con lo cual también parece necesario, para conservar la proporción, que estos tengan mayor canto.

En cualquier caso debemos recordar lo que señalamos en un principio: que se comportarán mejor aquellas mallas generadas próximas a la posición plegada, y que obtendremos la geometría de la malla en su posición desplegada únicamente una vez hallamos efectuado el correspondiente cálculo de despliegue dinámico, hasta que la malla llegue a su correspondiente posición de apertura estable.



## REFERENCIAS:

- [1] Laura, P. A. ; Pombo, J. L.  
"Introducción a la dinámica estructural"  
Fundación para la educación, la ciencia y la cultura. Cuenos Aires. 1980
- [2] Craig. R.R.  
Structural Dynamics  
ohn Wiley & Sons, Inc. U.S.A. 1981
- [3] Konstantin Meskouris  
"Structural Dynamics"  
Ernst & Sohn. 2000
- [4] Chapra, S.C. & Canale, R.P.  
"Métodos Numéricos para Ingenieros"  
Mc Graw - Hill. México. 1999
- [5] Ghali, A. ; Neville, A. M.  
"Análisis Estructural. Un enfoque unificado, clásico y por matrices"  
Diana. México, 1983
- [6] Landau & Lifshitz.  
"Física teórica. Mecánica"  
Reverté S.A. 1965
- [7] Beer, F. & Johnston E.  
"Mecánica vectorial para ingenieros"  
Mc Graw-Hill, Mexico 1.990
- [8] Lopez, E. & Muñoz, M.  
"Dynamic Behavior of Deployable Stuctures. Influence of the Damping Effects in the Compatibility Limitations of Grids"  
Spatial Structures In New And Renovation Projects of Buildings and Construction. Proceedings ICSS-98. 1998. Moscow, Russia.
- [9] Clark, S.K.  
"Dinámica de elementos continuos"  
Reverté, S.A. Barcelona 1975
- [10] Paz, Mario  
"Dinámica estructural. Teoría y Cálculo."  
Reverté S.A. 1992. Tercera edición.
- [11] Konstantin Meskouris  
"Structural Dynamics"  
Ernst & Sohn. 2000

---

## 3. Validación

---



## 3.1 BARRA SIMPLE

En este capítulo intentaremos ver la validez del modelo, efectuando cálculos con el mismo y comparando sus resultados con valores ya contrastados, o de solvencia indudable. Aparte de esto, los resultados de estos análisis pueden servirnos para efectuar un ajuste más fino de nuestro modelo.

En primer lugar validaremos los resultados del cálculo con el programa desarrollado, con la estructura más simple que puede existir, como es una única barra. Como en este caso tenemos los resultados para sistemas continuos, será fácil comprobar la fiabilidad del método de cálculo.



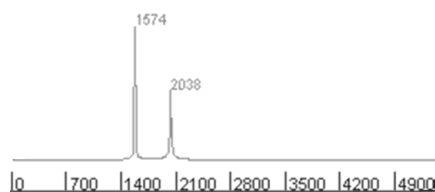
### GENERALIDADES.

Vemos que podemos comparar el funcionamiento de nuestra barra idealizada con los modos 1 y 2 de una viga libre en vibración axil, pero no podemos llegar a un modo superior, pues nuestro modelo ya no lo podría reproducir. En cualquier caso los modos más importantes son siempre los más bajos, especialmente el modo 1, que es al que tendremos que intentar aproximarnos todo lo posible.

Para unos cálculos cómodos usamos una barra con las siguientes características:

- $A = 1 \text{ cm}^2$  de sección =  $10^{-4} \text{ m}^2$
- $I = 1 \text{ cm}^4 = 1 \cdot 10^{-8} \text{ m}^4$
- $L = 1 \text{ m}$ .
- $\rho = 10.000 \text{ kg/m}^3 = [m/L^3=F/(a \cdot L^3)] = [N/(m^3 \cdot m/s^2)] = N \cdot s^2/m^4$
- $m = 1 \text{ kg/m} = [m/L=F/(a \cdot L)] = [N/(m \cdot m/s^2)] = N \cdot s^2/m^2$
- $E = 1.000.000 \text{ kg/cm}^2 = 1.000.000 \cdot 9,80665 \cdot 10^4 = 9,80665 \cdot 10^{10} \text{ N/m}^2$
- $g = 9.80665 \text{ m/s}^2$
- $\pi = 3.14159265358979$

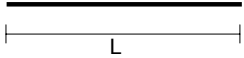
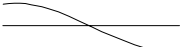

Calculando con nuestro modelo, podemos obtener los valores de una determinada frecuencia una vez efectuado el cálculo, si a la onda resultante le aplicamos la transformada de Fourier (y tenemos en cuenta que  $\omega = v \cdot 2\pi$ ), que en general nos dará unos picos muy pronunciados y claros, indicándonos cual es la frecuencia de cada caso:





## ■ VIBRACIÓN LONGITUDINAL. MODOS FUNDAMENTALES

Recordemos que para una barra con vibración longitudinal libre teníamos los modos principales según:

CASO	$k_1$	$k_2$
libre 	1 	2 

Siendo las frecuencias:

$$\omega_i = k_i \cdot \frac{\pi}{L} \cdot \sqrt{\frac{E}{\rho}} \quad [\text{rad/seg.}]$$

Sustituyendo los valores de la barra ejemplo en esta expresión obtenemos:

$$\omega_1 = 1 \cdot \frac{\pi}{L} \cdot \sqrt{\frac{E}{\rho}} = \pi \cdot \sqrt{\frac{9,80665 \cdot 10^{10}}{10000}} = 9838$$

$$\omega_2 = 2 \cdot \frac{\pi}{L} \cdot \sqrt{\frac{E}{\rho}} = 2 \cdot \pi \cdot \sqrt{\frac{9,80665 \cdot 10^{10}}{10000}} = 19676$$

Procedemos entonces al cálculo con el programa desarrollado usando los valores que hemos dado antes para la barra e intentando que la misma vibre según los modos 1 y 2.

Para conseguir que la barra empiece a vibrar con el modo 1 basta con alejar las masas de los extremos la misma distancia del centro, según el eje de la barra. Una vez que empieza el cálculo las fuerzas elásticas de recuperación inician por si solas la vibración.

El modo 2 se consigue de forma más sencilla, desplazando únicamente el nudo central en la dirección de la barra.



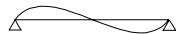
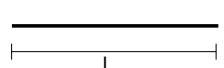

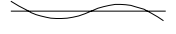
Del cálculo con el modelo desarrollado obtenemos los siguientes resultados, en función del coeficiente de distribución de masas  $f$ :

coefic. $f$	0,2	0,3	0,4	0,5	0,6	0,7
$\omega_1$	<b>7012</b>	<b>7476</b>	<b>8086</b>	<b>8853</b>	<b>9889</b>	<b>11423</b>
$\omega_2$	<b>15645</b>	<b>13647</b>	<b>12805</b>	<b>12497</b>	<b>12805</b>	<b>13647</b>

Vemos que efectivamente el valor de la primera frecuencia fundamental se aproxima a 0,6 como obtuvimos de cálculo analítico. Para ser exactos saldría  $f=0,947$ . Por contra, vemos que al segundo modo no nos podemos aproximar sin alejarnos demasiado del primero, por lo cual consideramos válido el valor que hemos tomado de  $f$ .

## ■ VIBRACIÓN TRANSVERSAL. MODOS FUNDAMENTALES

Como en el caso axial, podemos comparar el funcionamiento de nuestra barra idealizada con los datos que tenemos de una barra como sistema continuo.

CASO	$k_1$	$k_2$
biapoyada 	$\pi^2$ 	$4\pi^2$ 
libre 	22.3733 	61.6728 

Siendo las frecuencias:

$$\omega_i = k_i \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} \quad [\text{rad/seg.}]$$

### BARRA BIAPOYADA

Sustituyendo los valores de la barra ejemplo en esta expresión obtenemos:

$$\omega_1 = \pi^2 \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} = \pi^2 \cdot \sqrt{\frac{9,80665 \cdot 10^{10} \cdot 10^{-8}}{1}} = 309$$

$$\omega_2 = 4\pi^2 \cdot \sqrt{\frac{E \cdot I}{m \cdot L^4}} = 4\pi^2 \cdot \sqrt{\frac{9,80665 \cdot 10^{10} \cdot 10^{-8}}{1}} = 1236$$

Para los cálculos debemos coaccionar los dos extremos de la barra. Obtenemos el modo 1 separando el nudo 1 de su posición de equilibrio ortogonalmente a la barra. Hacemos notar aquí que se debe separar muy poco de la posición de equilibrio (en el ejemplo se usó 1 mm.) pues con valores grandes aparecerían axiles significativos por variación de la longitud de la barra

Para el cálculo del modo 2 dejemos el nudo intermedio en la posición de equilibrio y forzamos una variación del ángulo del nudo en su posición inicial.

Del cálculo con el modelo desarrollado obtenemos los siguientes resultados, en función del coeficiente de distribución de masas  $f$ . suponemos en este primer cálculo  $f=f'$ .

coefic. $f$	0,2	0,3	0,4	0,5	0,6	0,7
$\omega_1$	<b>477</b>	<b>402</b>	<b>339</b>	<b>301</b>	<b>282</b>	<b>263</b>
$\omega_2$	<b>4216</b>	<b>2299</b>	<b>1495</b>	<b>1068</b>	<b>804</b>	<b>647</b>

Para ser exactos deberíamos obtener  $\omega_1=309$  con  $f=0,49276$ . En cambio la coincidencia de  $\omega_2$  exigiría valores menores, del orden de 0,45

Para el segundo modo vamos a realizar ahora un nuevo cálculo en el cual dejamos fija  $f=0,55$  como valor por defecto y jugamos solo con el valor de  $f'$ :

coefic. $f'$	0,2	0,3	0,4	0,5	0,6	0,7
$\omega_2$	<b>2526</b>	<b>1684</b>	<b>1263</b>	<b>1012</b>	<b>842</b>	<b>723</b>

En el cálculo analítico obtuvimos  $f'=0,41$  en el modo 2. Vemos que los valores obtenidos se corresponden con este parámetro de referencia.

Vemos efectivamente que estas pruebas verifican la adopción de los valores de cálculo que hemos tomado:

<b>f = 0,55</b> <b>f' = 0,41</b>
-------------------------------------

### BARRA LIBRE

Sustituyendo los valores de la barra ejemplo en esta expresión obtenemos:

$$\omega_1 = 22,3733 \cdot \sqrt{\frac{EI}{mL^4}} = 22,3733 \cdot \sqrt{\frac{9,80665 \cdot 10^{10} \cdot 10^{-8}}{1}} = 700$$

$$\omega_2 = 61,6728 \cdot \sqrt{\frac{EI}{mL^4}} = 61,6728 \cdot \sqrt{\frac{9,80665 \cdot 10^{10} \cdot 10^{-8}}{1}} = 1931$$

Del cálculo con el modelo desarrollado obtenemos los siguientes resultados, en función del coeficiente de distribución de masas **f**. suponemos en este primer cálculo **f=f'**.

coefic. <b>f</b>	0,2	0,3	0,4	0,5	0,6	0,7
$\omega_1$	<b>534</b>	<b>477</b>	<b>440</b>	<b>438</b>	<b>440</b>	<b>477</b>
$\omega_2$	<b>4216</b>	<b>2300</b>	<b>1495</b>	<b>1112</b>	<b>880</b>	<b>766</b>

Vemos que  $\omega_1$  no se aproxima demasiado con ningún valor de **f**. por lo que no tiene sentido intentar ajustarlo; para  $\omega_2$  parece que van mejor valores inferiores a 0,4. Como en el caso anterior, efectuaremos un nuevo cálculo para el segundo modo en el cual dejamos fija **f=0,55** como valor por defecto y jugamos solo con el valor de **f'**:

coefic. <b>f'</b>	0,2	0,3	0,4	0,5	0,6	0,7
$\omega_2$	<b>2563</b>	<b>1721</b>	<b>1301</b>	<b>1068</b>	<b>917</b>	<b>804</b>

Vemos que en este caso es prácticamente imposible realizar más ajustes, con los valores adoptados para **f=** y **f'** obtenemos:  $w_1=439$  ;  $w_2=1263$ .

Andamos algo lejos de estos valores, pero optamos por no variar nuestros coeficientes, siempre y cuando, en nuestros modelos de trabajo lo habitual es que la barras trabajen más próximas a la idealización de biapoyadas que a la de libres, pues tienen siempre sus extremos conectados con otras barras.

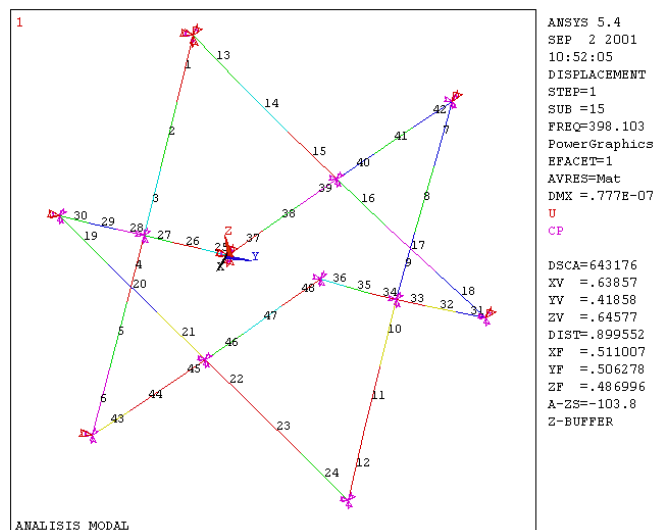
Concluimos por tanto que los coeficientes adoptados verifican y reproducen bien el comportamiento de una barra simple biapoyada, hasta el segundo modo fundamental de vibración.

## 3.2 CONJUNTO DE BARRAS.

Pasaremos a verificar ahora el comportamiento de algo más complejo, como es un conjunto de barras. Como en este caso no tendremos a nuestra disposición las ecuaciones que rigen el comportamiento de estos sistemas, la comparativa tendremos que hacerla en base a los programas de cálculo por elementos finitos que existen en el mercado, escogiendo alguno de renombre, cuyos resultados se hallen completamente garantizados.

### MODOS PROPIOS DE VIBRACIÓN.

Antes de intentar ningún cálculo en el dominio del tiempo, nos meteremos primero con algo más sencillo. Para ello usaremos un modelo sencillo de cuatro aspas estudiaremos sus modos fundamentales de vibración con el programa ANSYS 5.4



El número de elementos por barra que hemos elegido es de seis, y el tipo de elemento el BEAM4.

El listado que genera la estructura es el siguiente:

```
○ /filnam,DESPL
○ /title,ANALISIS MODAL
○ /PMETH,OFF
○ KEYW,PR_SET,1
○ KEYW,PR_STRUC,1
○ KEYW,PR_THERM,0
○ !*
○ /PREP7
○ !*
○ ET,1,BEAM4
○ R,1,.0001,1e-8,1e-8,.01,.01,0,
○ !*
○ UIMP,1,EX,,9.80665e10,
○ UIMP,1,DENS,,10000,
○ !*
○ K,1,0,0,1,
○ K,2,0,0,0,
○ K,3,1,0,1,
```

```

O K,4,1,0,0,
O K,5,0,1,1,
O K,6,0,1,0,
O K,7,1,1,1,
O K,8,1,1,0,
O K,9,0,0,1,
O K,10,0,0,0,
O K,11,1,0,1,
O K,12,1,0,0,
O K,13,0,1,1,
O K,14,0,1,0,
O K,15,1,1,1,
O K,16,1,1,0,
O !*
O /PNUM,KP,1
O /REPLOT
O !*
O /VIEW, 1, .6386 , .4186 , .6458
O /ANG, 1, -103.8
O /REPLO
O LSTR, 1, 4
O LSTR, 5, 8
O LSTR, 9, 6
O LSTR, 3, 16
O LSTR, 2, 11
O LSTR, 14, 7
O LSTR, 10, 13
O LSTR, 12, 15
O !*
O LATT, 1, 1, 1, 0
O ESIZE,0,6,
O LMESH,ALL
O /PNUM,ELEM,1
O /REPLOT
O !*
O !los nodos próximos los unimos con rótulas
O CPINTF,UX,0.0001,
O CPINTF,UY,0.0001,
O CPINTF,UZ,0.0001,
O !*
O nsel,s,,1
O nsel,a,,,8
O nsel,a,,,29
O nsel,a,,,16
O D,all,ux,0
O nsel,s,,,1
O nsel,a,,,2
O nsel,a,,,29
O nsel,a,,,22
O D,all,uy,0
O nsel,s,,,29
O nsel,a,,,1 !para isostática
O D,all,uz,0
O nsel,all
O !*
O !*
O ANTYPE, 2 !modal
O !*
O MODOPT,SUBSP,7
O EQSLV,FRONT
O MXPAND,7, , ,0
O LUMPM,0
O PSTRES,0
O FINISH
O !*
O !*
O /SOLU
O SOLVE
O !*
O !*
O !*
O /post1
O pldisp,0
O anmode,10,5e-1

```

La lista de resultados obtenida de este análisis modal es la siguiente lista de frecuencias:

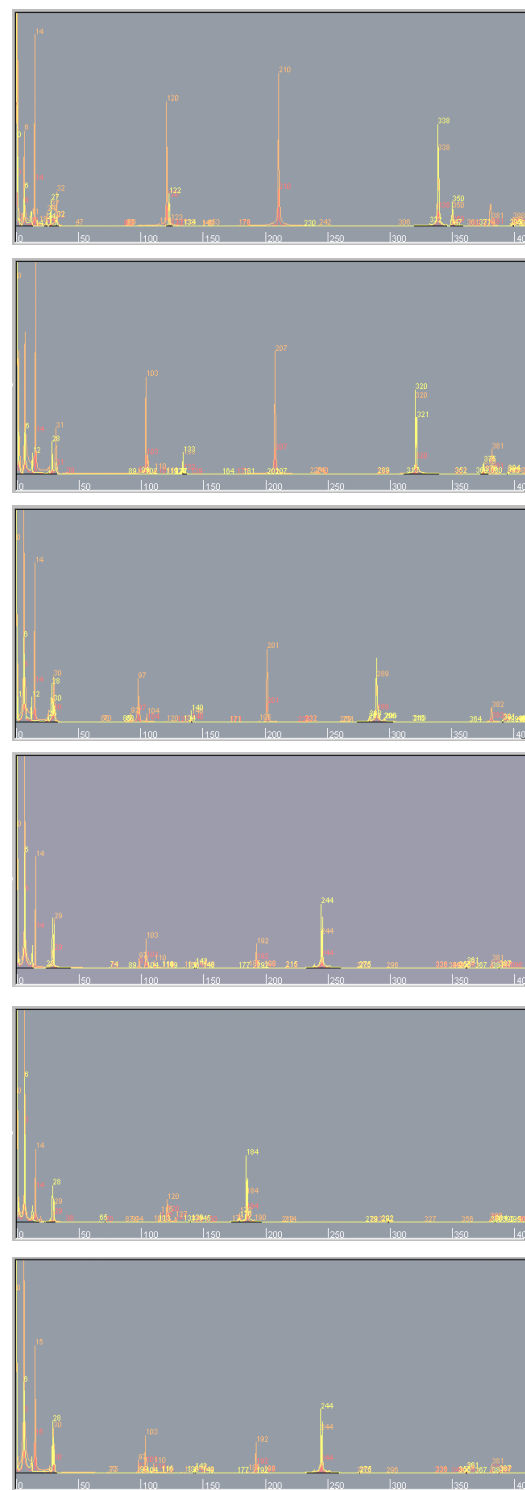
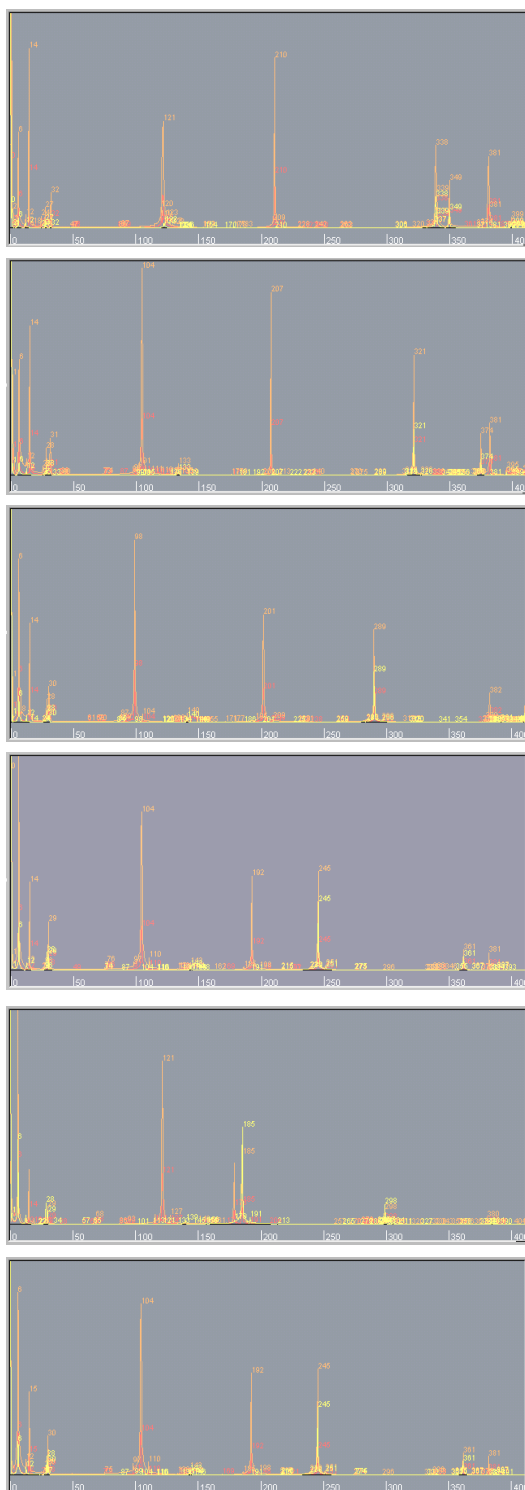
```
○
○ ANALISIS MODAL ANSYS 5.4
○
○ SET   FREQ (Hz)
○   1   5.5449
○
○   2   10.776
○
○   3   13.911
○   4   13.915
○
○   5   24.542
○   6   24.590
○   7   24.590
○   8   25.584
○
○   9   29.041
○  10   32.570
○
○  11   98.155
○  12   98.158
○  13   98.247
○  14   98.255
○  15   98.263
○  16   98.264
○  17   98.363
○  18   98.363
○  19   98.363
○  20   98.363
○
○  21  102.02
○  22  104.12
○
○  23  111.80
○  24  111.98
○
○  25  116.85
○  26  118.92
○
○  27  149.83
○  28  151.88
○  29  152.34
○  30  152.74
○  31  153.81
○  32  153.81
○  33  154.87
○
○  34  170.71
○  35  170.86
○
○  36  174.84
○  37  177.83
○
○  38  208.09
○
○  39  217.51
○
○  40  221.74
○  41  221.74
○
○  42  245.55
○
○  43  373.95
○  44  374.67
○  45  378.00
○
○  46  385.63
○
○  47  394.09
○  48  394.36
○  49  396.61
○  50  396.61
○  51  396.61
○  52  396.61
○  53  397.36
○  54  398.10
○
```

Analizando el comportamiento de nuestra estructura mediante la transformada de Fourier obtendremos los siguientes resultados:

desplazamientos nudo 7

f y f'

desplazamientos nudo 8

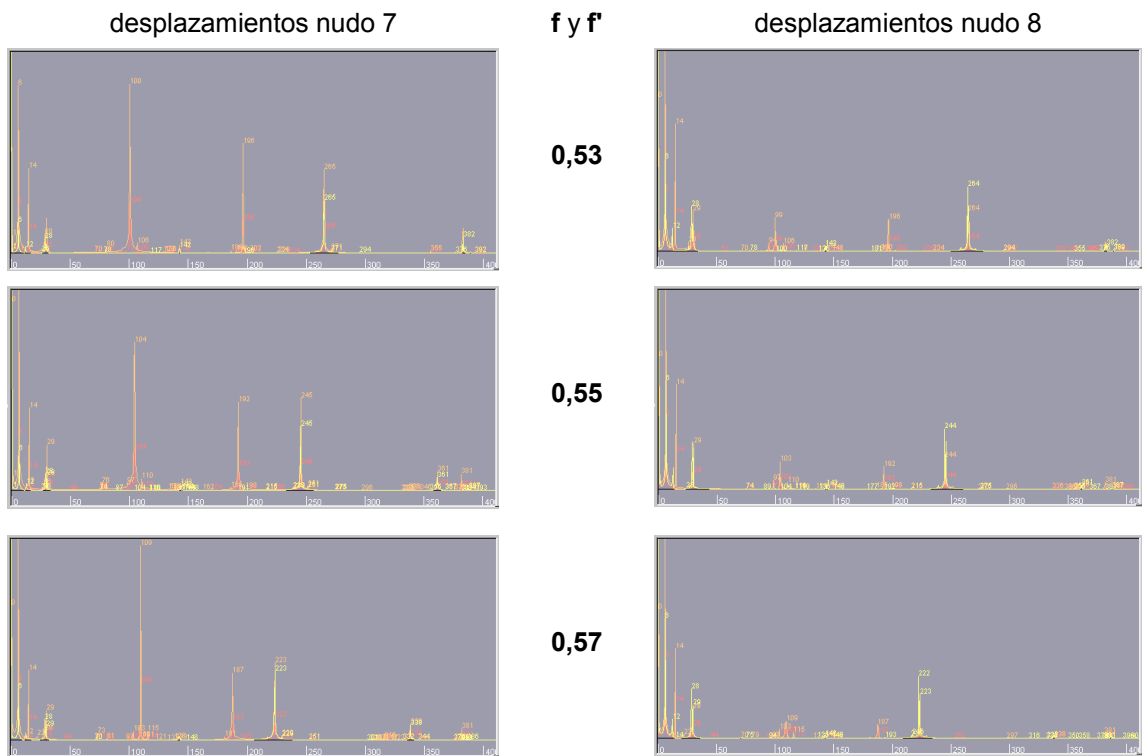


f = 0,55  
f' = 0,41

La lista de frecuencias del ANSYS la podríamos resumir como sigue, indicando con asteriscos la importancia relativa de cada frecuencia.

- 5.5 \*
- 11 \*
- 14 \*
- 25 \*\*
- 31 \*
- 98 \*\*\*\*
- 103
- 112
- 117
- 150
- 154
- 171
- 176
- 208
- 217
- 222
- 245
- 375 \*
- 385
- 395 \*\*

Vemos que los valores que más se asemejan son los correspondientes a los coeficientes 0,55 y en estos vemos que el valor de  $f'$  prácticamente no varía la gráfica. Para una comparativa más detallada, veamos lo que ocurre alrededor del coeficiente 0,55 con la finalidad de intentar ajustar un poco más.



En las frecuencias bajas, de 103 hacia abajo, los valores son casi idénticos para todas las gráficas, y la variación empieza a ser notable en las frecuencias más altas. En cualquier caso no encontramos mucha mejora con coeficientes ligeramente distintos de 0,55 por lo que consideramos éste como válido. La única discrepancia un poco notable es que aparece una frecuencia destacada de 192 que no debiera estar ahí, ya que las más próximas son 176 o 208, pero obviamente, al ser un modelo simplificado, la coincidencia no puede ser completa al 100%.



## ANÁLISIS DE DESPLIEGUE

Pasamos a intentar el análisis comparativo de un despliegue completo del módulo de cuatro aspas. Para ello usaremos el programa SAP90, de Computers and Structures, Inc. El módulo de estudio, para introducir en este programa, queda más o menos como sigue:

```
○ MALLA DESPLEGABLE, cuatro aspas solamente
○
○ SYSTEM
○ L=1 V=3
○
○ JOINTS
○ 1 X= 0 Y= 0 Z= 1
○ 2 X= 0 Y= 0 Z= 0
○ 3 X= 1 Y= 0 Z= 1
○ 4 X= 1 Y= 0 Z= 0
○ 5 X= 0 Y= 1 Z= 1
○ 6 X= 0 Y= 1 Z= 0
○ 7 X= 1 Y= 1 Z= 1
○ 8 X= 1 Y= 1 Z= 0
○ 9 X= 0 Y= 0 Z= 1
○ 10 X= 0 Y= 0 Z= 0
○ 11 X= 1 Y= 0 Z= 1
○ 12 X= 1 Y= 0 Z= 0
○ 13 X= 0 Y= 1 Z= 1
○ 14 X= 0 Y= 1 Z= 0
○ 15 X= 1 Y= 1 Z= 1
○ 16 X= 1 Y= 1 Z= 0
○
○ RESTRAINTS
○ 1 R=1,1,1,0,0,0
○ 2 R=1,1,1,0,0,0
○ 3 R=0,1,0,0,0,0
○ 4 R=0,1,0,0,0,0
○ 5 R=1,0,0,0,0,0
○ 6 R=1,0,0,0,0,0
○
○ CONSTRAINTS
○ 9 C=1,1,1,0,0,0
○ 10 C=2,2,2,0,0,0
○ 11 C=3,3,3,0,0,0
○ 12 C=4,4,4,0,0,0
○ 13 C=5,5,5,0,0,0
○ 14 C=6,6,6,0,0,0
○ 15 C=7,7,7,0,0,0
○ 16 C=8,8,8,0,0,0
○
○ FRAME
○ NM=1
○ 1 A=0.0001 E=9.80665E10 I=1E-8,1E-8 M=1 :BARRAS
○ 1 1 4 M=1
○ 2 5 8 M=1
○ 3 9 6 M=1
○ 4 3 16 M=1
○ 5 2 11 M=1
○ 6 14 7 M=1
○ 7 10 13 M=1
○ 8 12 15 M=1
○
○ LOADS
○ 7 L=1 F=10,10,0
○
○ TIMEH
○ ATYPE=0 NSTEP=4096 DT=1E-4 F=1 D=0.9999999
○ NF=1 NPL=3 DT=1
○ 1 1 1
○ LC=1 NF=1 S=1 AT=0
```

Lamentablemente SAP, pese a tener un módulo de análisis en el tiempo, no permite el estudio de mecanismos !?. Cuando intentamos efectuar el cálculo da el siguiente error:

```
○  
○ FORMING MASS MATRIX  
○ E Q U A T I O N   S O L U T I O N   P H A S E  
○ REDUCING STIFFNESS BLOCK      1 OF      1  
○ REDUCING STIFFNESS BLOCK      1 BY BLOCK      1  
○ * * * E R R O R * * *  
○ NON-POSITIVE DIAGONAL =      .00000D+00 FOUND WHILE REDUCING EQUATION #      3  
○ STRUCTURE MAY BE UNSTABLE OR MAY HAVE BUCKLED UNDER P-DELTA LOAD  
○ EXCESSIVE ERRORS OR IMMEDIATELY FATAL ERROR  
○ MODIFYING BATCH FILE "GO.BAT" TO TERMINATE EXECUTION  
○ EXECUTION TERMINATED  
○
```

En realidad, si nos fijamos en el manual más detenidamente, dice que para solucionar este tipo de problema usa el método estándar de superposición de formas modales, ya sean autovectores, o los vectores Ritz (dependientes del estado de cargas).

Lo que nos viene a decir es que en realidad no efectúa el cálculo dinámico que precisamos para efectuar la comparativa, sino que realiza uno pseudo-dinámico utilizando formas modales.

Debido a ello no podemos efectuar la comparativa, pero podemos concluir que programas de renombre, como el aquí mencionado, no llegan a efectuar cálculos que permitan observar el despliegue de mallas, cálculo que sí efectúa perfectamente el programa que hemos desarrollado en este trabajo.

# 4. Resultados.



---

## 4.1 ESTUDIO DE INCOMPATIBILIDADES.

---

Las mallas desplegadas son estructuras cuyo cálculo presentan numerosas dificultades, pero como ya comentamos, en muchas tipologías el despliegue está influido por incompatibilidades geométricas en las posiciones intermedias que añaden nuevos factores complejos a su análisis. Aunque las limitaciones de compatibilidad ya han sido estudiado por varios autores para mallas considerando un despliegue estático o cuasi estático. Esta aproximación no es suficiente para estructuras temporales, en las cuales la velocidad de erección y desmantelamiento es una parte importante de la funcionalidad de la estructura.

Para el análisis de tales fenómenos, estudiaremos comparativamente dos mallas semejantes, pero cuyas incompatibilidades geométricas son significativamente distintas. Asimismo, veremos la importancia que la velocidad de despliegue, directamente relacionada con los fenómenos de amortiguamiento, tiene en los esfuerzos máximos que soporta la estructura.

---

## INTRODUCCIÓN.

---

Las mallas desplegadas son adecuadas para superficies de cubierta que son planas sin presentar mayores problemas. Con superficies de doble curvatura ya empiezan a presentar incompatibilidades geométricas, como ocurría con la forma cilíndrica, y las incompatibilidades son en general manifiestamente mayores cuando hablamos de superficies de doble curvatura, tipo cúpulas esféricas o similares [1]. Esto ocurre, especialmente, cuando los módulos base son triángulos puesto que añaden bastante rigidez al conjunto y no poseen la flexibilidad de módulos de base cuadrada para readaptar su forma, convirtiéndose en rombos, cuando se acomodan en superficies de doble curvatura. Por esta razón, e independientemente de las incompatibilidades del proceso constructivo, cuando las superficies a cubrir son de doble curvatura, como cúpulas esféricas, con mallas de módulos triangulares, normalmente, son posibles dos configuraciones estables: la inicial y la desplegada. Sin embargo, la geometría de las posiciones intermedias no son factibles si se considera que las barras son segmentos rectos. Es necesario considerar la curvatura introducida en las barras por efecto de la flexión. Por esta misma razón, no es factible ejecutar reconstrucciones del proceso de despliegue en base a consideraciones puramente geométricas.

La reconstrucción del proceso del despliegue usando sistemas informáticos requiere por tanto tener en cuenta los esfuerzos que se generan en la estructura y por tanto sus deformaciones. Aun así esto se puede efectuar de forma estática, realizando este proceso punto a punto en las distintas posiciones de despliegue, pero lo que a nosotros nos interesa, y realiza el programa, es emular el comportamiento real de la estructura, teniendo en cuenta sus velocidades, aceleraciones, y por tanto, los efectos inerciales, que pueden ser muy importantes si la estructura es de despliegue rápido.

## MALLAS ESTUDIADAS EN ESTE TRABAJO

En este apartado estudiaremos dos estructuras desplegadas. Son relativamente semejante en geometría y dimensiones generales, características de las barras y carga aplicadas, pero substancialmente diferentes en las incompatibilidades geométricas que experimentan durante el proceso de despliegue [2].

### Características Comunes:

Con ambas, se pretende hacer una cubierta de 10 m de radio mediante una cúpula esférica con una malla desplegable de tercera orden, formado por módulos en aspa. las barras tienen una longitud aproximada de 1.80 m. En el comienzo del proceso de despliegue, estos módulos no se supusieron totalmente plegados, sino cubriendo un arco de  $10^\circ$  desde del polo de la cúpula. Las cargas verticales son de 10 kg. en cada nodo de la capa superior, y 30 kg. en los de la capa inferior. Para facilitar el despliegue, se aplicaron unas cargas constantes horizontales en los nodos superiores de las esquinas, y de 50 kg. que tiran en la dirección radial. Las barras son tubos huecos de sección circular ( de 4 cm. de diámetro y 2 mm. de espesor de pared ) hechas de acero. Durante el despliegue y para facilitararlo, la malla se considera apoyada en los nodos de la capa inferior adyacente al nodo central. En el cálculo se tiene en cuenta el peso propio de las barras y se efectuó una corrección de la rigidez a flexión en función del valor de los axiles de las barras.

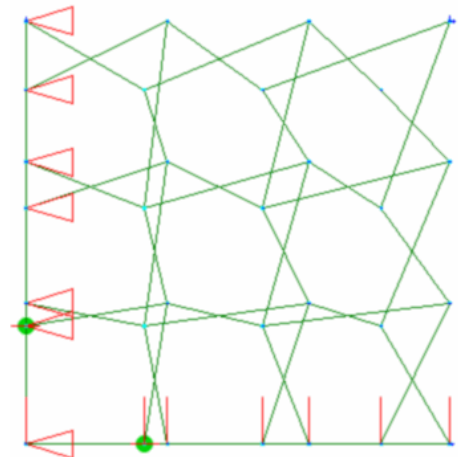
### Diferencias:

Puesto que el objetivo principal de este estudio es mostrar la relevancia de las incompatibilidades geométricas, se usarán una malla con pocas o ninguna incompatibilidad, usando el tipo de módulo cuadrangular, y otra que, a priori, tendrá grandes incompatibilidades, puesto que usaremos en ella el módulo triangular.

## LA MALLA DE MÓDULO CUADRANGULAR

Esta malla fue escogida para sus pocas incompatibilidades geométricas cuando despliega, y usamos proceso de generación de la malla que garantiza esto.

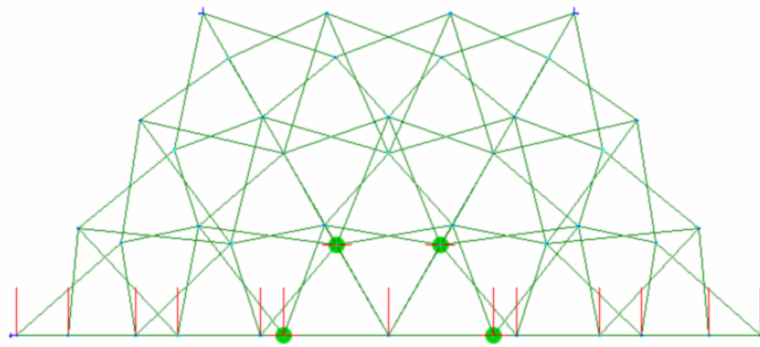
El proceso de generación ya lo hemos comentado, y sucintamente consiste de empezar con un nodo en la cima de la cúpula. El esquema de los módulos se dibuja contra la superficie de la esfera, empezando por del dos meridianos ortogonales que paso a través de el nodo central. Sobre estos meridianos se traza la longitud inicial de la retícula. El próximo punto en la superficie de la esfera se sitúa a una distancia igual de estos dos puntos previos, así sucesivamente, de modo que obtenemos una retícula cuadrangular con módulos de lados iguales dibujada sobre la superficie de la esfera. Esta forma de generación tiene la ventaja que todas las barras son de idéntica longitud y que no presenta incompatibilidades geométricas, y la desventaja que no todos los módulos son semejantes, siendo casi cuadrados en la cima de la cúpula, y resultando rombos mas o menos alargados en las esquinas.



La existencia de una doble simetría tiene la ventaja que solamente precisamos estudiar una cuarta parte de la malla. Respecto a la malla de módulos triangulares presenta la diferencia de que no tiene definida una posición máxima de despliegue. Por esta razón, para mantenerla en una posición desplegada, será necesario fijar los puntos de borde, y estos tendrán que garantizar la estabilidad de la estructura.

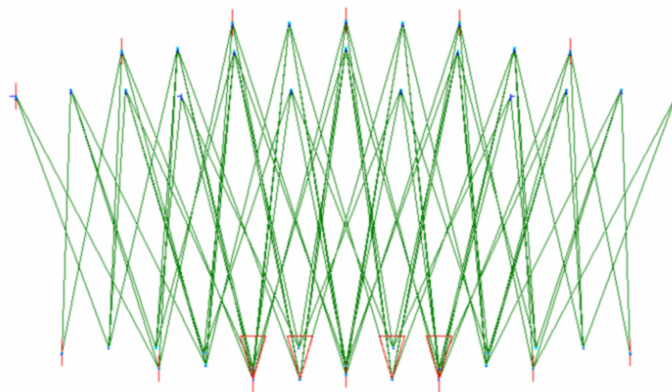
## LA MALLA DE MÓDULO TRIANGULAR

Como hemos dicho, esta malla se escogió por sus altas incompatibilidades en posiciones intermedias. El proceso de generación que nosotros hemos usado está basado en proyectar una malla llana de triángulos equiláteros situada en el plano ecuatorial sobre la superficie esférica, usando como foco un punto próximo al polo del sur, situado 0.5 m abajo, en este caso.



Para lograr que en el punto de despliegue completo los ángulos que forman las barras con la superficie esférica sean similares, el canto de la malla se modifica progresivamente. Su máximo está en la zona próxima a el polo superior, donde las barras tienen longitudes del orden de 2 m, disminuyen hacia los bordes, donde las barras tienen una longitud alrededor de 1.60 m.

Como en la otra malla usamos los puntos de la capa inferior próximos al central, como apoyos en los que suponemos sujeta o colgada la malla durante el despliegue. La simetría en este caso nos permite analizar solamente media malla. Igualmente como posición inicial se elige una posición próxima al plegado total con una extensión de  $10^\circ$  desde el polo. En esta malla también podemos fijar una posición de despliegue total, que fijamos en un ángulo de apertura de  $80^\circ$  desde polo superior. En esta posición es donde casi se puede verificar de nuevo la compatibilidad geométrica de la estructura. Veremos que esta posición de despliegue final es altamente estable no siendo tan crucial la fijación de los puntos de borde como en el caso anterior.

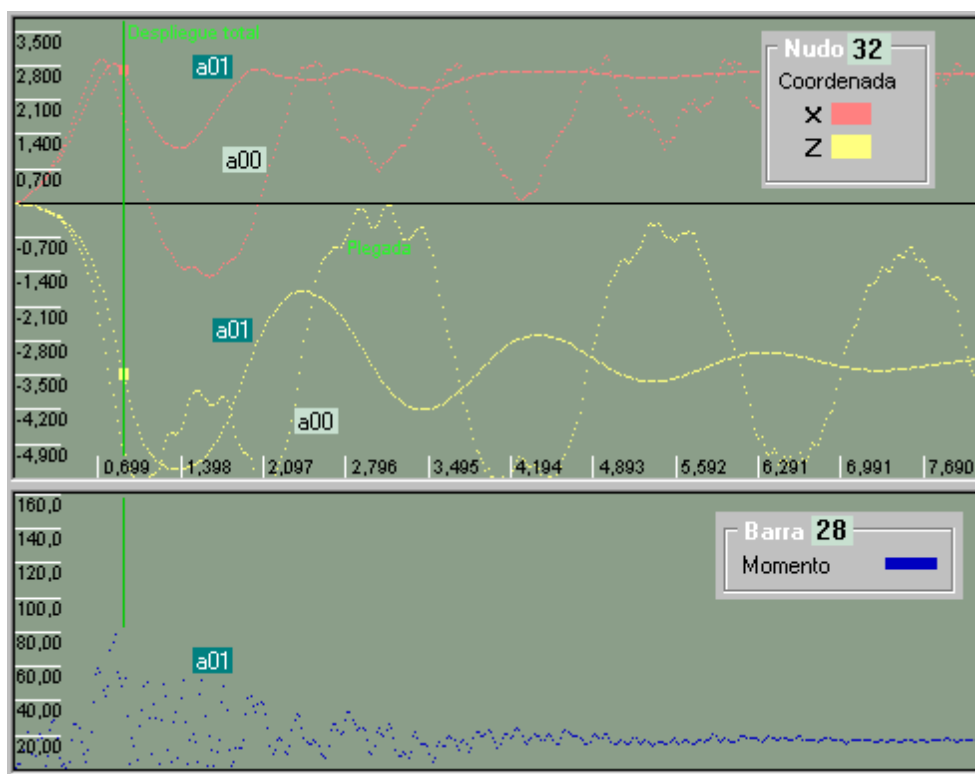


## VERIFICACIÓN DE INCOMPATIBILIDADES

Antes de del proceso de cálculo definitivo, en primer lugar, quisiéramos poner de manifiesto el diferente comportamiento de ambas estructuras durante el proceso de despliegue. Para esto, procedemos a eliminar todas las cargas verticales, excepto su propio peso (para tener una carga mínima durante el proceso de despliegue), y efectuamos un cálculo con amortiguamiento nulo y otro con un amortiguamiento pequeño. La idea de este proceso es que las cargas aplicadas son tan bajas, que cuando la malla presente cualquier incompatibilidad geométrica, las cargas no serán capaces de superarla y se producirá un efecto de 'rebote'.

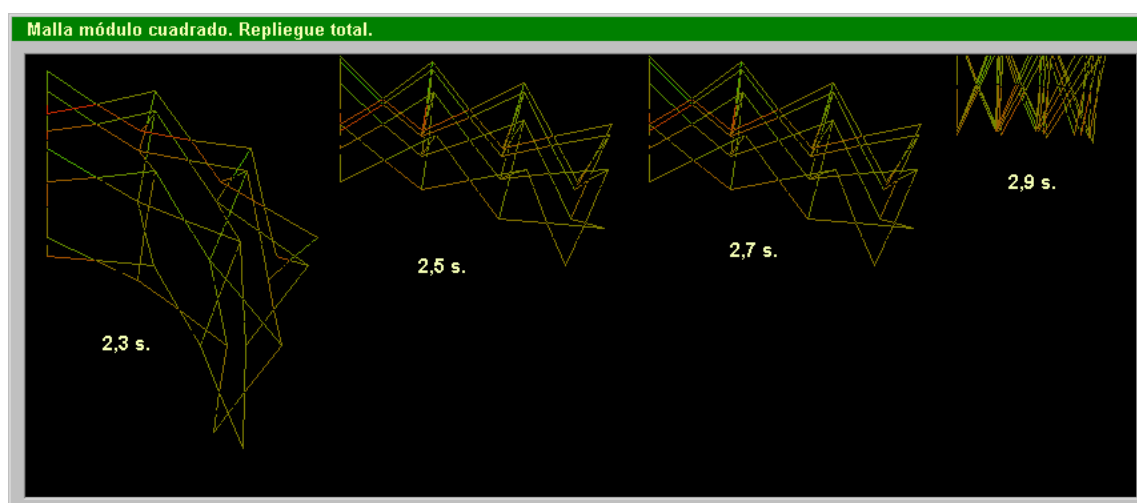
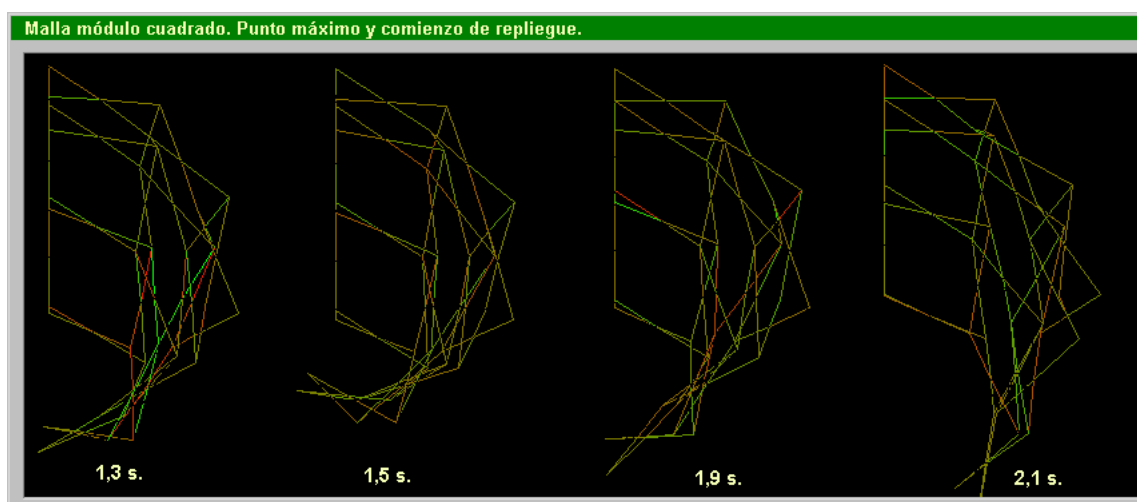
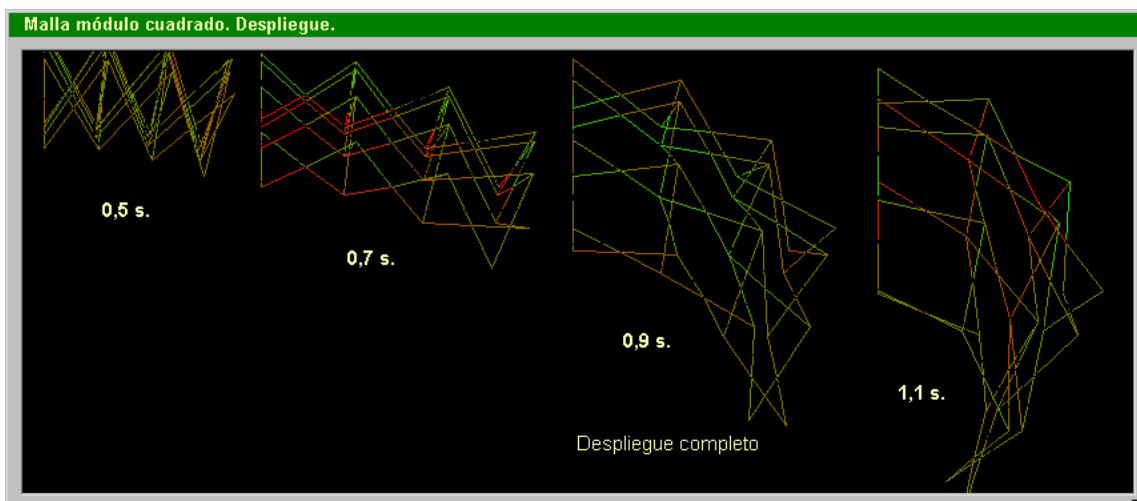
### Malla de módulo cuadrangular.

El gráfico siguiente representa el tiempo en segundos en el eje horizontal, y la posición de un punto en metros en el eje vertical, y el momento flector que se produce en una de las barras. El punto de referencia tomado para los desplazamientos es una de las esquinas de la malla.



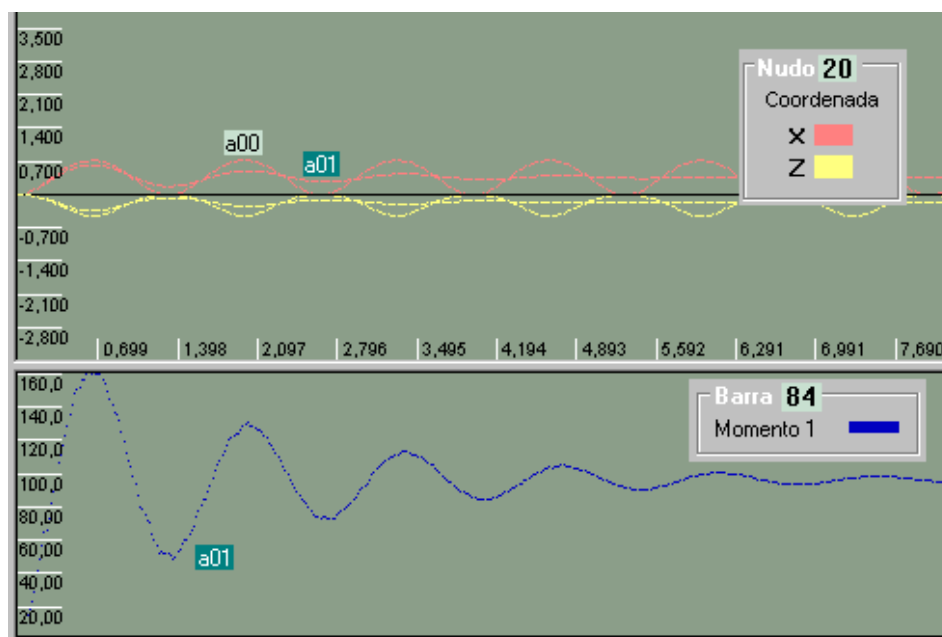
Vemos que el punto sobrepasa el punto de despliegue completo de forma substancial sin que aparezca ninguna discontinuidad en la gráfica. Las barras presentan una flexión de pequeña importancia y prácticamente independiente del proceso de despliegue, que se hace más estable si consideramos un amortiguamiento mínimos de 1 unidad ( representado en el gráfico como **a01** ). En el cálculo, si el amortiguamiento es nulo ( **a00** ), entra un ciclo repetitivo, yendo más allá de el máximo punto de despliegue y regresa, volviendo hasta el punto inicial, sin fuerzas de flexión significativas

A continuación vemos el alzado de la malla en el despliegue, sin amortiguamiento, a intervalos de 0,2 segundos, hasta completar un primer ciclo de despliegue - repliegue.



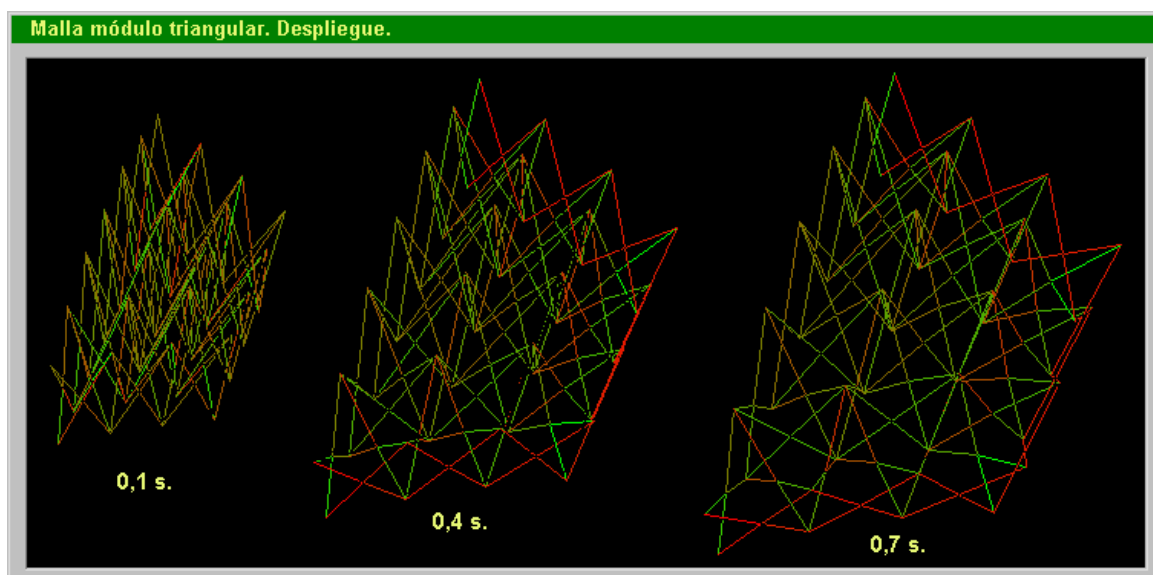


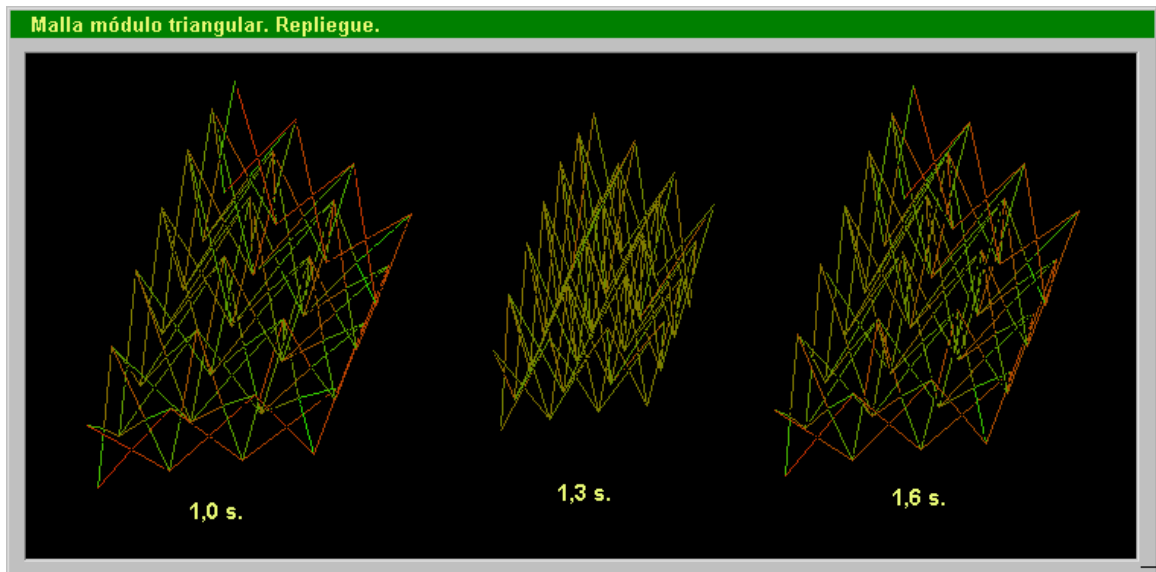
## Malla de módulo Triangular



En este caso, el proceso de despliegue no se completa. En la parte superior del gráfico, dibujado a la misma escala que el anterior, se observa que la malla despliega un poco, pero aparecen grandes incompatibilidades, como pone de manifiesto la parte inferior del gráfico, donde vemos que el punto de máximos desplazamientos coincide con el máximo momento flector, y en este caso, las fuerzas externas no son suficientes para superar la zona de incompatibilidad, y se produce el efecto de rebote.

Esto se puede apreciar perfectamente en los siguientes gráficos, donde vemos en axonometría las distintas fases del despliegue, sin amortiguamiento, a intervalos de 0,3 segundos.



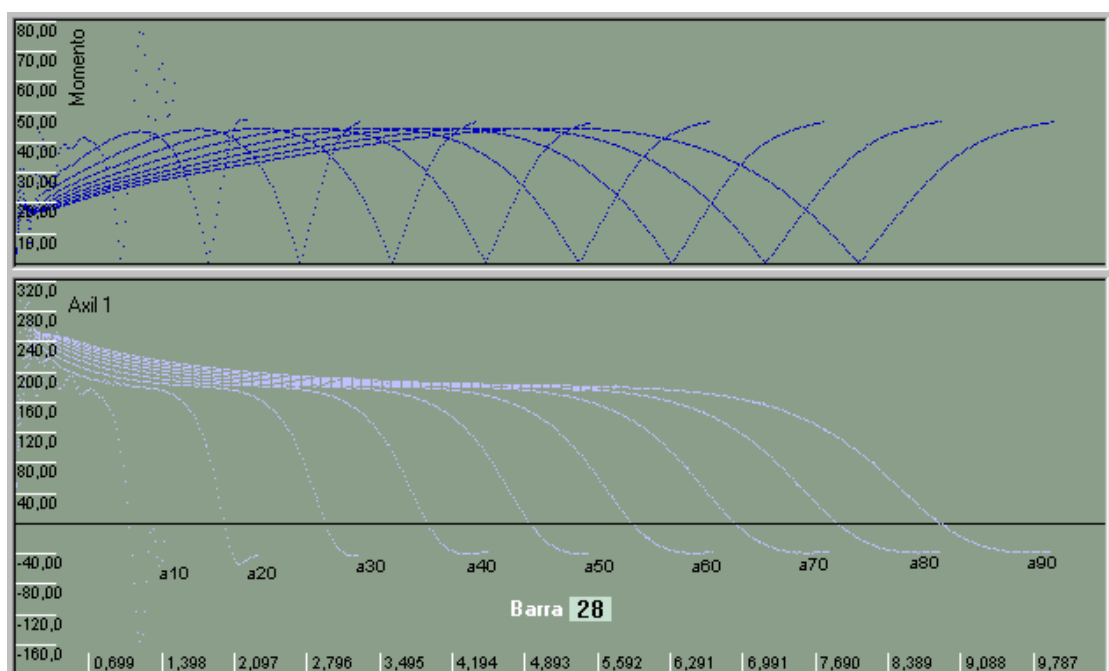


Como se observa en los gráficos, las diferencias, de este caso con el previo son notables: por una parte, el proceso de despliegue total no se alcanza y por otra, aparece una relación directa entre la posición de despliegue y el momento flector de las barras, lo cual nos indica la presencia de la incompatibilidad geométrica, que la estructura no puede superar al tener aplicadas unas cargas muy pequeñas.

## ESTUDIO DEL DESPLIEGUE COMPLETO

Esta es la sección principal de este trabajo, donde analizaremos el comportamiento de ambas mallas en un proceso típico de despliegue. Nosotros trataremos de describir, sobre todo, las diferencias de comportamiento, en cuanto a los esfuerzos en las barras y la influencia del grado de amortiguamiento en este proceso [3].

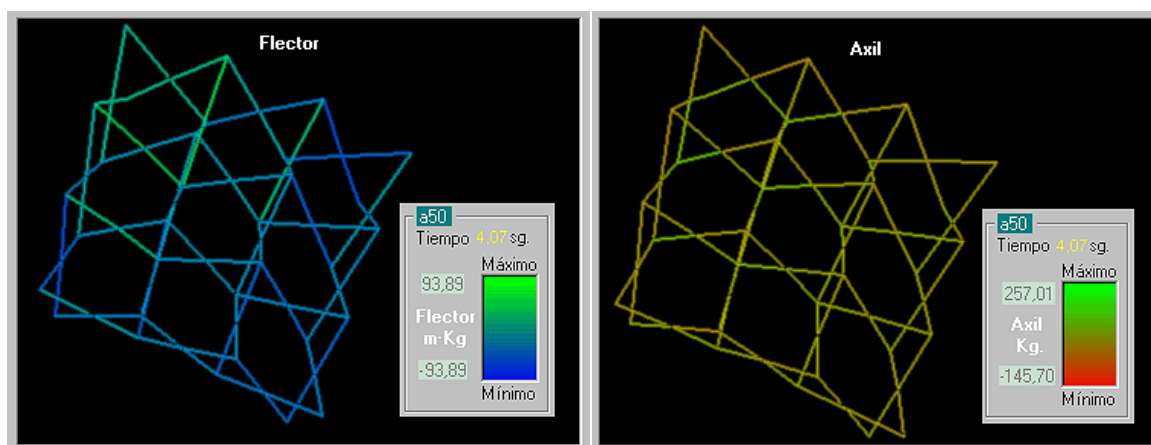
### Malla de módulo cuadrangular.



En el estudio del despliegue, se observa que la influencia del amortiguamiento en la duración del mismo. Es una relación bastante lineal para el rango de amortiguamiento estudiado, aumentando el tiempo de despliegue al aumentar el amortiguamiento.

Los esfuerzos axil y flector se han representado en función del amortiguamiento(  $a_{10}$  . .  $a_{90}$  ) en el gráfico anterior, para una barra de la zona central (nº 28), que está entre las más solicitadas. Para ambos esfuerzos, se observa que los mismos varían sin relativas brusquedades o discontinuidades y que la influencia que el grado de amortiguamiento tiene en el máximo valor alcanzado por los esfuerzos en las barras es relativamente pequeña (las unidades son kg. y m.). Precisamente, las barras en las cuales aparecieron variaciones más grandes son en las de borde, que son a su vez las que presentan los valores más bajos de los esfuerzos.

Por último, en los gráficos siguientes podemos observar la distribución de axiles y flectores, para un grado intermedio de amortiguamiento (50) y en una posición próxima al despliegue total. Se evidencia que las barras que reciben más carga son las próximas a la región central.



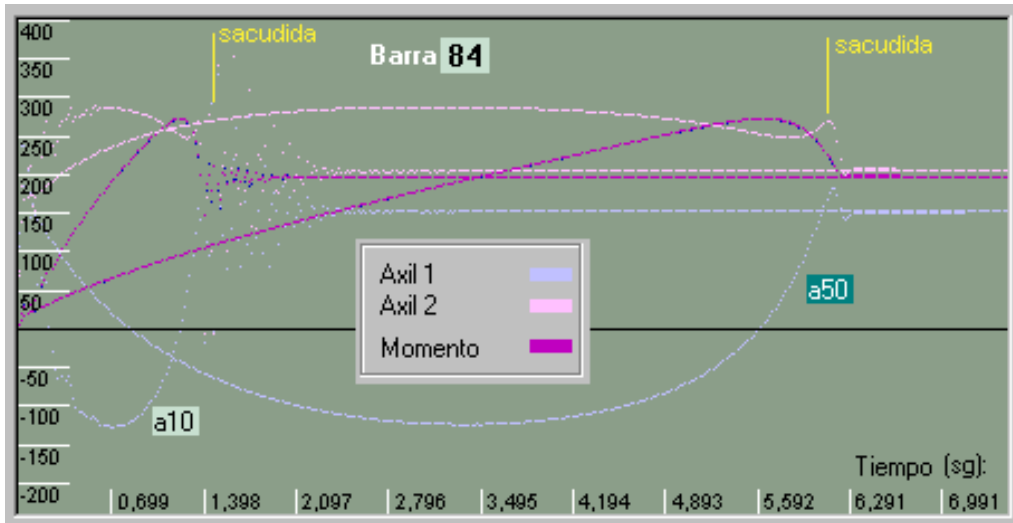
### Malla de módulo Triangular

Para conseguir el despliegue ha sido preciso incrementar la magnitud de las cargas aplicadas para superar las incompatibilidades. Aparte del peso propio y las cargas constantes de 50 kg. tirando en dirección radial y horizontal desde los nodos de esquina, se han añadido unas cargas verticales hacia abajo de 30 kg. en cada nudo de la cara superior.

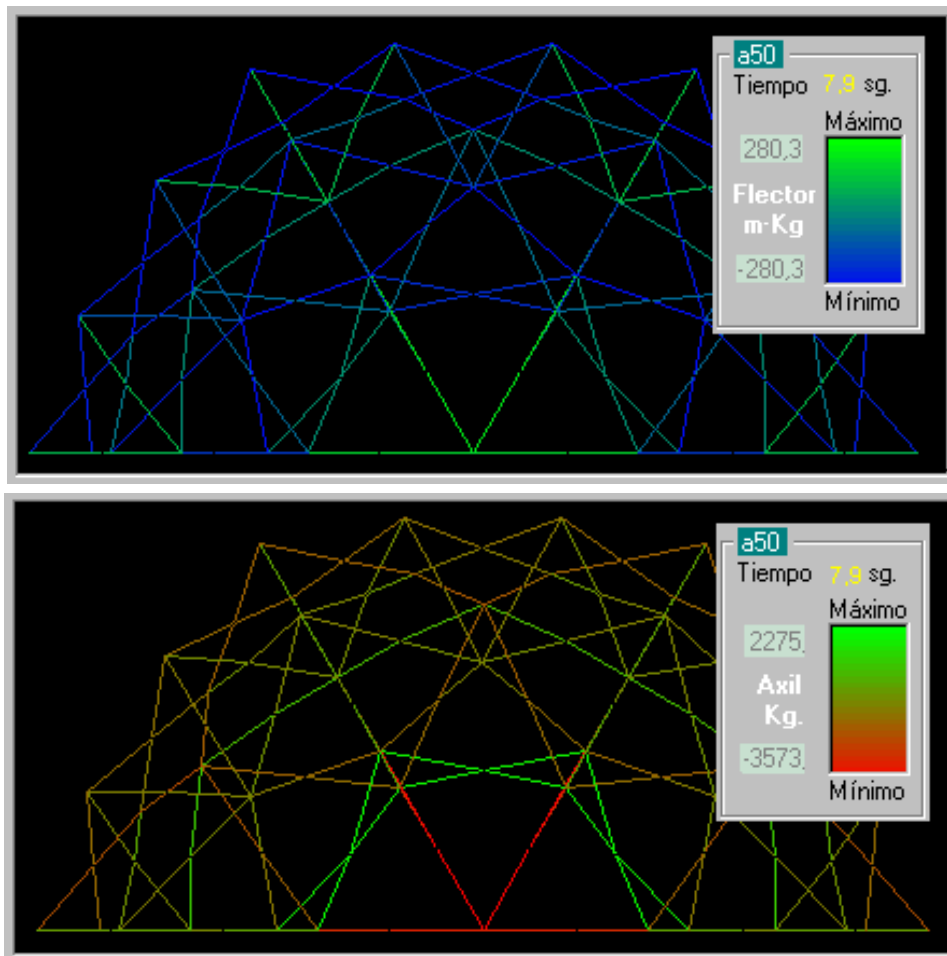
Como ya se ha visto en la sección de estudio de incompatibilidades, las fuerzas en las barras de esta estructura están bastante relacionadas con la geometría del despliegue. Ahora, una vez el proceso de despliegue se ha completado, esta característica se observa mucho mejor. Adicionalmente, se observa que la posición desplegada es altamente estable, puesto que una vez se ha alcanzado, la estructura experimenta como una sacudida, como si se parara de repente, y permanece en esta posición.

Esto puede ser perfectamente observado con un gráfico de los esfuerzos que se dan en una de las barras cercanas al borde, como es la 84. Para simplificar, en el gráfico solamente se han representado las curvas con amortiguamientos de 10 y 50. Comprobamos que el aumento de fuerzas es notable durante el proceso de despliegue hasta llegar a máximos relativos hacia el medio del despliegue, reduciendo los máximos un poco con el aumento del amortiguamiento. El efecto de tope al llegar a la posición de despliegue total también es apreciable como una breve sacudida al final de la gráfica (punto donde se produce una dispersión de la gráfica), justo antes de alcanzar una posición en la que permanece fija, por lo que se deduce que es altamente estable.

En el momento de la sacudida también se verifica que el incremento de los esfuerzos es menor a medida que se incrementa el amortiguamiento, pasando por ejemplo en la barra 84 de 350 a 270 kg. de axil en este instante al pasar el amortiguamiento de 10 a 50.



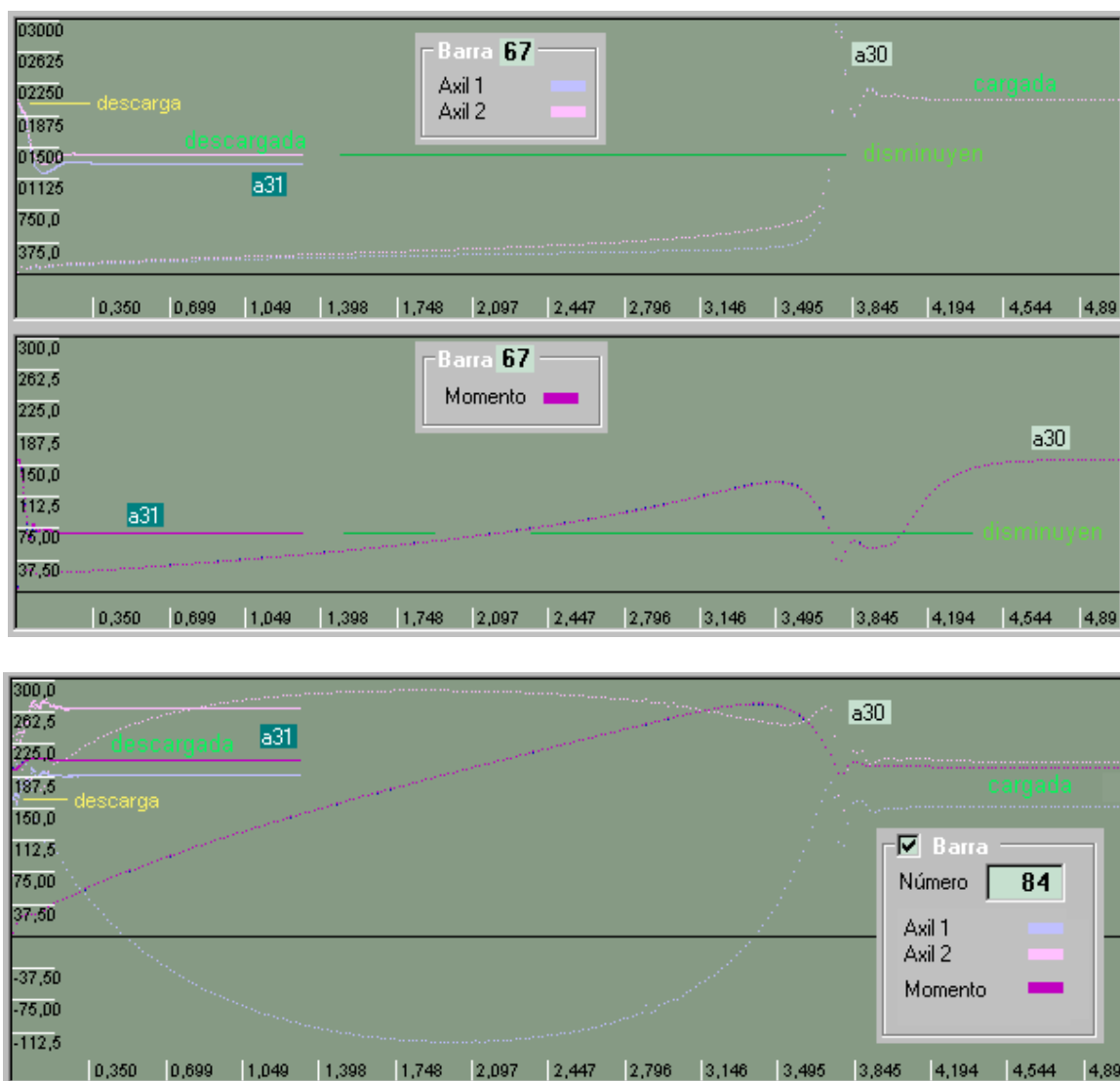
Por último, también observamos la distribución de axil y flector para un grado intermedio de amortiguamiento (50) y en una posición total despliegue. Como en el caso anterior se hace evidente que las barras centrales son las más cargadas, aunque barras con axiles importantes también se presentan en el borde.



## INFLUENCIA DE LA GEOMETRÍA EN LAS FUERZAS

En el estudio del despliegue de esta última estructura la influencia de la geometría (incompatibilidades) es apreciable en los esfuerzos. Así, también es interesante conocer la importancia relativa de los esfuerzos debidos a la geometría con respecto a los totales. Para ello hemos procedido al un cálculo en la posición ya expandida, eliminado todas las cargas externas, de modo que los esfuerzos en las barras sean debidos exclusivamente a su geometría.

Para este estudio nos hemos decantado por usar un grado de amortiguamiento pequeño, de valor 30. En el siguiente gráfico este cálculo está indicado con **a30** y su continuación, a partir del momento en que hemos eliminado las cargas externas, con **a31**, en el que el amortiguamiento de 30 se ha mantenido.



Los gráficos muestran axiles y flectores de una barra central (barra 67) y una de borde (barra 84), que se corresponden con las que presentan los mayores valores durante el cálculo. La barra central (67) es la que sufre los mayores axiles, que reducen su valor a 2/3 (de 2062 a 1400 kg.) una vez las cargas externas se eliminan. Los momentos se reducidos a la mitad, pasando de 155 a 72 m·kg.

En la barra cerca de del borde (barra 84), las fuerzas axiles son mucho más bajas, pero las de flexión son mayores. Es interesante observar que ocurre lo opuesto que en el caso anterior cuando

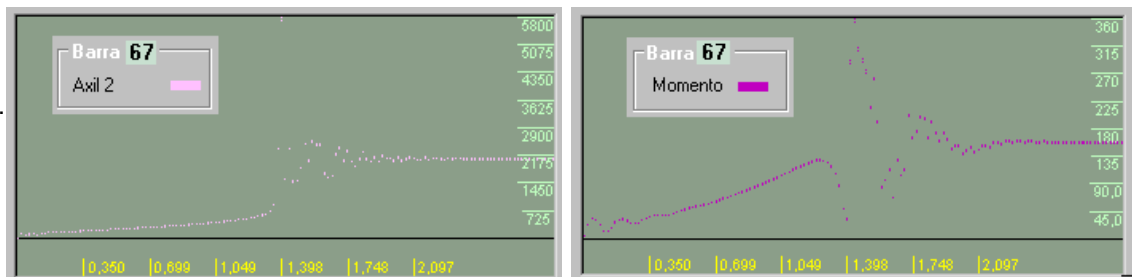
cargas externas son eliminadas. La fuerza axil pasa de 220 a 270, con un 23 % de aumento y también se produce aumento del momento pero con una incidencia menor, de 190 a 210 (10%).

## CONCLUSIONES

Después de este análisis, podemos concluir que las incompatibilidades geométricas presentadas por algunas mallas tienen una gran repercusión en las fuerzas que ellos tendrán que resistir.

En algunos casos como el estudiado, si estamos en casos extremos de bajo amortiguamiento, en el momento de la sacudida los esfuerzos máximos pueden ser multiplicados por factores algo superiores a 2.

Esto se representa en las gráficas siguientes donde se representan los esfuerzos en la barra 67 con un amortiguamiento de  $10^{-1}$ sg. En ellas se aprecia que los axiles se han multiplicado por un factor 2,6 en el momento de la sacudida (pasan de 2175 a 5800 kg.) y los flectores por 2,25 (pasan de 160 a 360 m·kg.), con la repercusión consiguiente en el dimensionado de las barras y costo.



Debemos hacer notar que aun cuando las cargas externas se eliminan, esta malla continúa cargada, indicando que en muchos casos, será la geometría, con sus incompatibilidades, la que generará una gran parte de los esfuerzos que soportarán las barras de estas estructuras.

Por otra parte, haremos notar que las estructuras con fuertes incompatibilidades presentan grandes ventajas comparadas con las otras, como es que son altamente estable en la posición desplegada, no dependiendo tanto de la correcta fijación de los nudos de borde.

## 4.2 FENÓMENOS INERCIALES.

Pasaremos ahora a estudiar un ejemplo de estructura que aun **no** presentando incompatibilidades geométricas, se originan en ellas unos picos de esfuerzos durante el proceso de despliegue.

Estudiaremos la naturaleza del fenómeno e intentaremos determinar cual es el origen de que se produzcan estos incrementos bruscos de los esfuerzos, todo ello atendiendo al comportamiento dinámico de la estructura. [4] [5]

### DEFINICIÓN DE LA ESTRUCTURA

La estructura que emplearemos en este estudio es una estructura desplegable formada por barras de aluminio. La estructura completa consta de 174 barras con 84 nodos intermedios y 101 nodos extremos. Las características de las barras son:

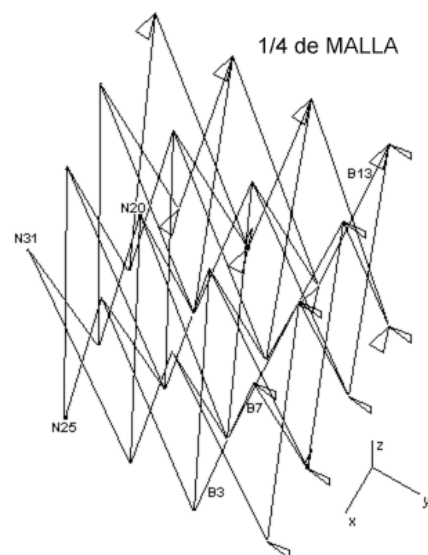
Área:  $9.1 \text{ cm}^2$   
Momento de inercia:  $108 \text{ cm}^4$   
Modulo de elasticidad:  $714286 \text{ Kg/cm}^2$   
Límite elástico:  $2363 \text{ Kg/cm}^2$   
Peso específico:  $7850 \text{ kg/m}^3$   
Longitud:  $3,80 \text{ m}$ .

Para el despliegue, la estructura se supone sujeta por cuatro cables conectados con los nudos de la capa inferior adyacentes al central. Con esta configuración la estructura puede desplegar con su propio peso únicamente, pero para abreviar el tiempo de despliegue, se aplicó a mayores una carga horizontal de 28 Kg. en cada esquina y en la dirección diagonal. En la figura siguiente se puede ver la estructura dibujada con las barras (B3, B13, ...) y nodos (N20, N25, ...) relevantes.

Para reducir el tiempo de cálculo y espacio de almacenamiento de datos, aprovechamos la doble simetría de la estructura y analizamos sólo un cuarto de la misma. Por tanto hemos restringido el movimiento horizontal de los nudos situados en los planos de simetría.

El peso de la estructura es el debido a sus barras y nudos más una cubrición textil, estimada en  $5 \text{ kg/m}^2$  de estructura totalmente desplegada, que fue aplicada en los nudos de la capa superior.

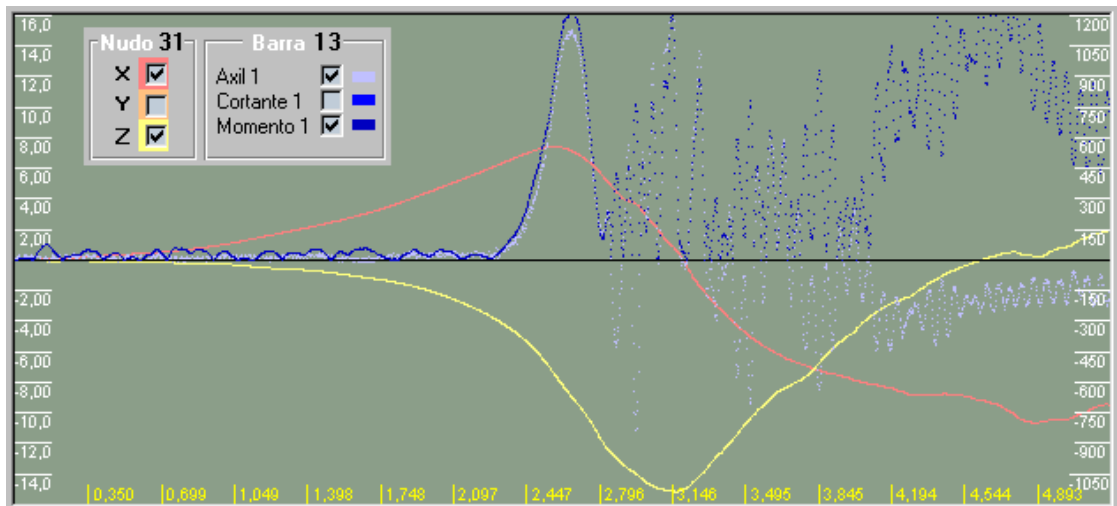
El proceso de despliegue se ha dejado libre, analizando un poco más allá de lo que sería la posición desplegada total. En esta fase final, la estructura se curva hacia adentro, con una variación significativa de la dirección de despliegue, que origina que las fuerzas inerciales causen un incremento de los esfuerzos de las barras, que no deberían existir en condiciones normales, como ya se ha visto en el estudio anterior.



## RESULTADOS

Se comprueba que en la fase inicial del despliegue, los esfuerzos en las barras son casi constantes o con oscilaciones poco importantes, y esto continúa así hasta que pasamos lo que sería la posición típica de despliegue y proseguimos más allá. Poco después se produce una subida brusca de las tensiones, sin que esto aparentemente tenga nada que ver con temas de incompatibilidades de la estructura.

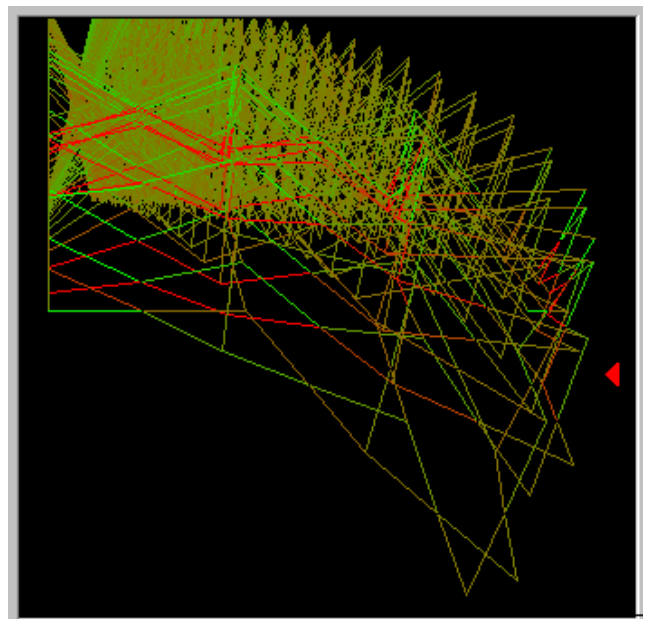
Si observamos el comportamiento del nudo extremo 31 y la barra más solicitada a flexión, la 13, la gráfica obtenida es la siguiente:



Se observa que el pico de esfuerzos se produce poco después de que la coordenada X del nudo extremo alcance su máximo, o sea, cuando la dirección de despliegue de la estructura cambia de expandirse en horizontal a empezar a contraerse.

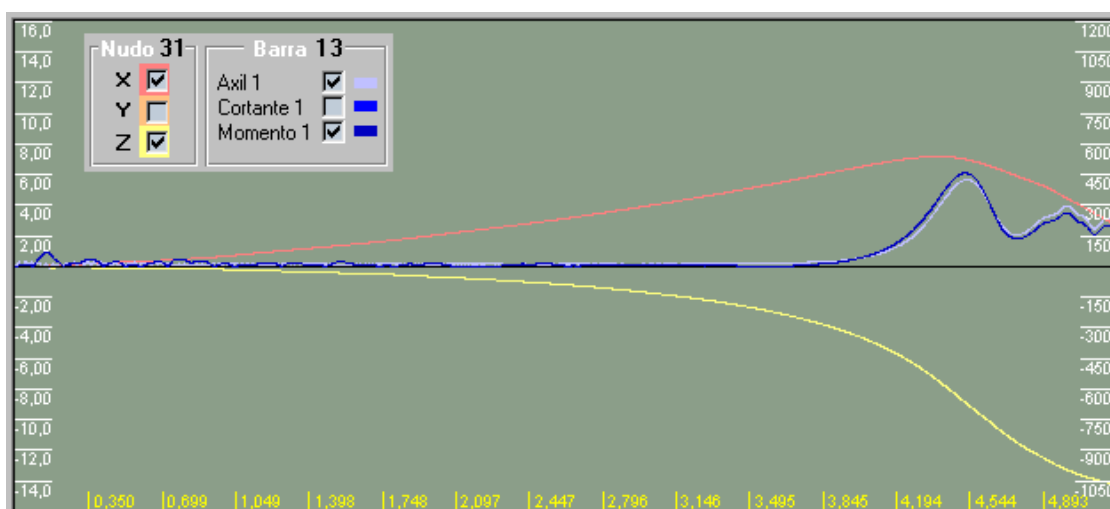
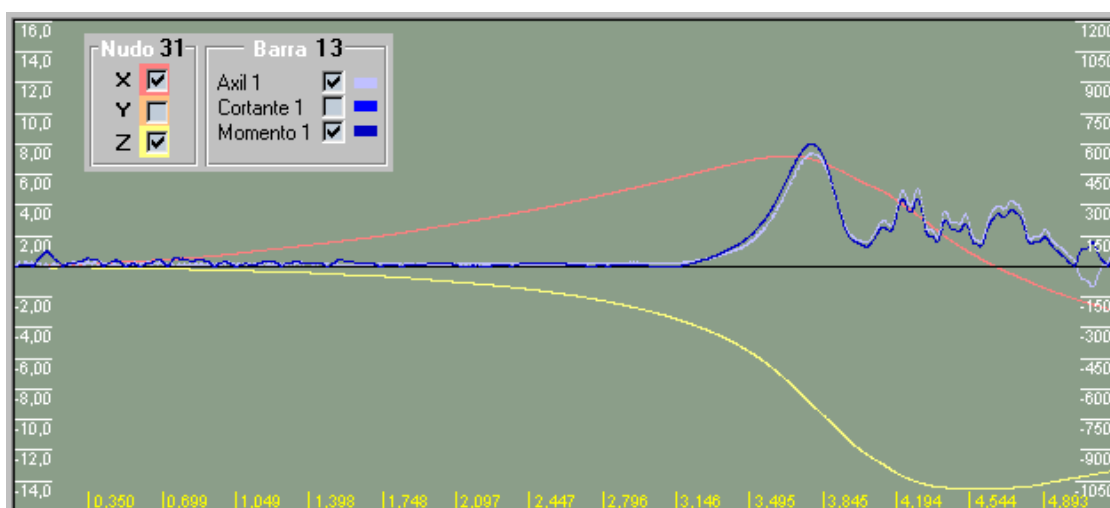
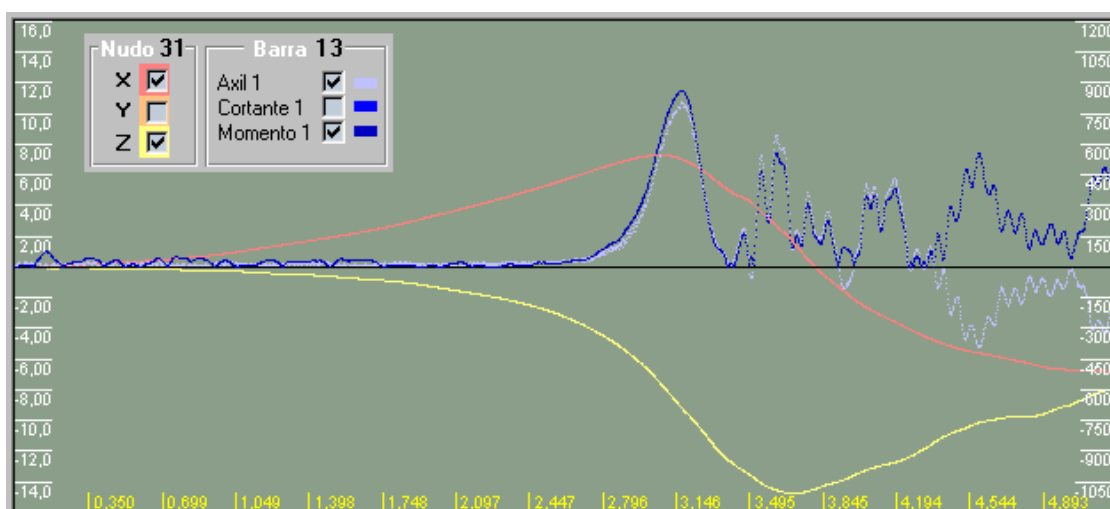
Si superponemos los gráficos del despliegue de la estructura en una vista en alzado, tendríamos lo siguiente:

La marca roja nos señala la posición del extremo de la malla en la posición en que se da el pico de esfuerzos, y la última posición dibujada es justo cuando el pico de esfuerzos ha decaído. A partir de esta posición la malla prosigue su despliegue, aunque no tiene sentido estudiar más allá de este punto. La malla incluso repliega totalmente sobre sí misma, posición en la que el gráfico de las X toma valores negativos, y sigue más allá, pero en este caso los otros cuartos simétricos chocarían entre sí.





Si repetimos el cálculo inicial, pero añadiendo algo de amortiguamiento, obtenemos tres nuevos gráficos, con amortiguamientos de **0.5**, **1** y **1.5**  $\text{sg}^{-1}$  respectivamente.

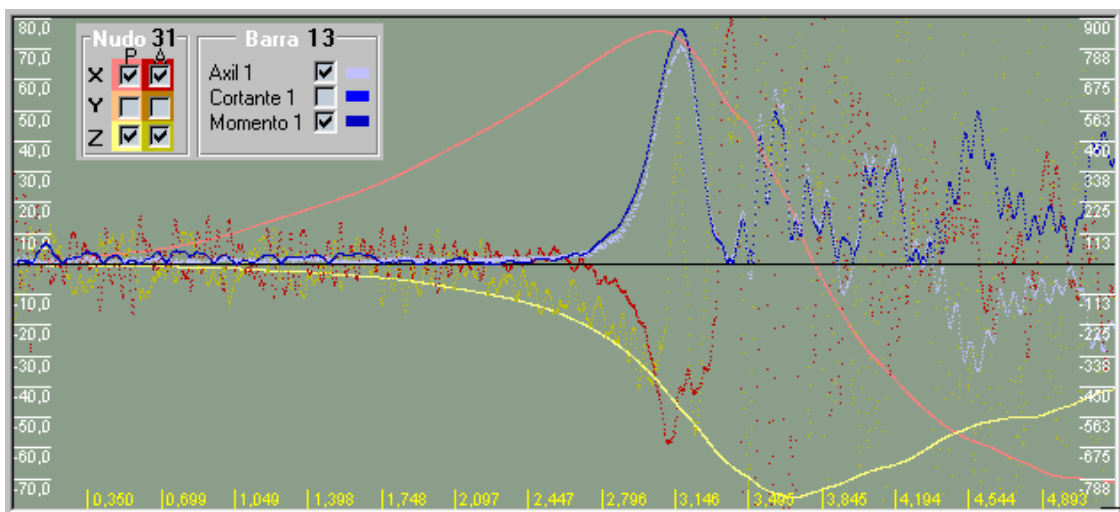


Como era de esperar, el primer efecto del amortiguamiento es un retraso de las funciones. En cuanto a las posiciones del nudo, siguen alcanzando máximos similares, llegando la X a un máximo

de cerca de 8 metros, lo que nos indica que la forma general de despliegue será prácticamente la misma con independencia del amortiguamiento.

En cambio, en cuanto los esfuerzos, aparte del retraso, sí sufren variaciones significativas en los valores de los máximos, reduciéndose drásticamente a medida que se aumenta el amortiguamiento. En cualquier caso se siguen produciendo los máximos justo después de que el avance horizontal (X) de la malla cambie de sentido, hecho que nos induce a pensar, junto con que sean tan dependientes del amortiguamiento, que son los efectos inerciales los causantes de este incremento de esfuerzos.

Para confirmarlo podemos añadir a una gráfica anterior las aceleraciones que tiene el nudo extremo de la estructura.



Como esperábamos, el pico de esfuerzos de la barra coincide con el pico de aceleraciones del nudo, pese a tratarse de un nudo extremo y una barra interior que no están en contacto directo. Esto nos viene a confirmar que efectivamente, son las fuerzas inerciales debidas al movimiento de la estructura la provocan los aumentos de tensión, sin que intervenga para nada las incompatibilidades geométricas de la estructura.

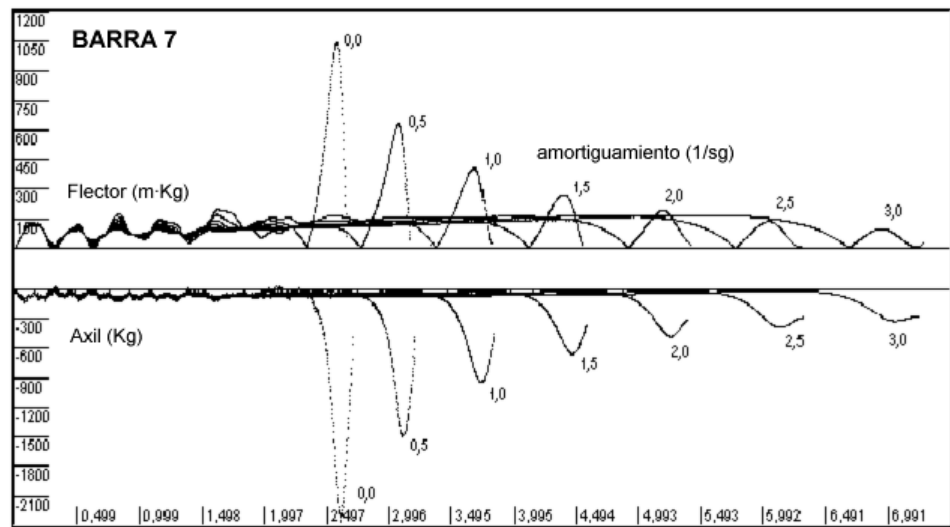
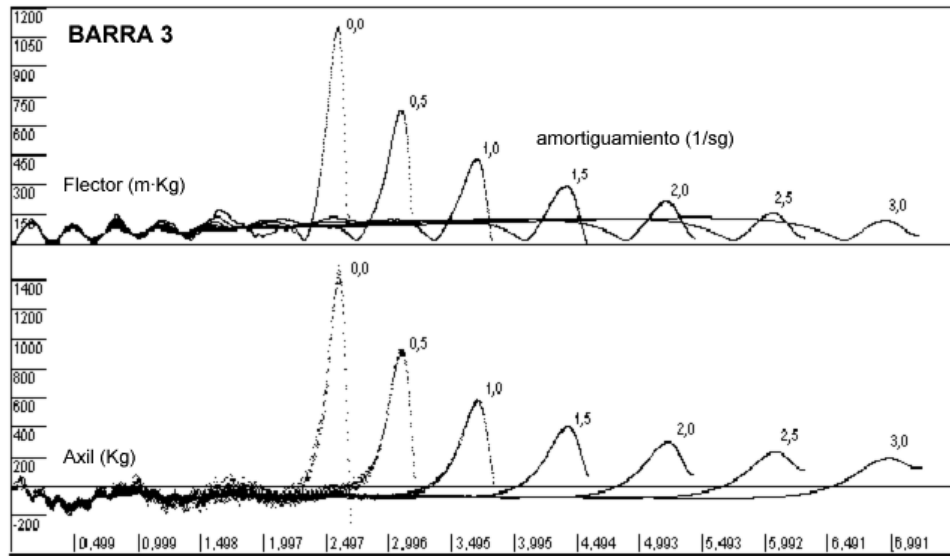
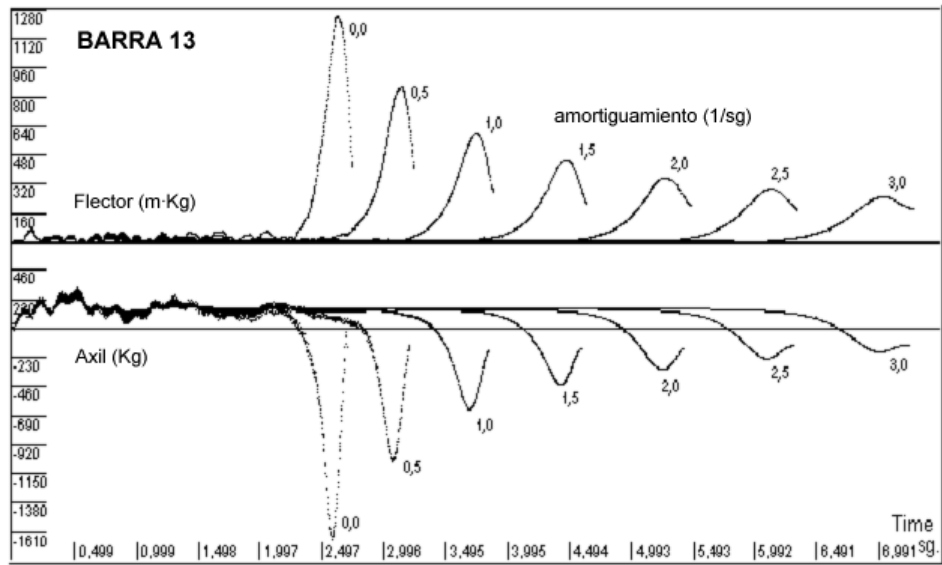
Los esfuerzos más importantes de la estructura se localizan en las barras siguientes.

Barra 13: momento flector

Barra 3: axil positivo

Barra 7: axil negativo

Para ver la influencia del amortiguamiento, para estas tres barras superpondremos los gráficos de esfuerzos para los diversos grados de amortiguamiento, para ver la evolución del pico de esfuerzos:



En todos los casos se observa que las tensiones al comienzo del despliegue son casi constantes, con mayor o menor dispersión dependiendo del amortiguamiento. Es en la fase final cuando se produce el incremento de tensiones, y el valor máximo sí es dependiente del amortiguamiento, decreciendo su valor a medida que crece el amortiguamiento, pero no de manera lineal, sino según una curva que puede aproximarse a una función de segundo grado o logarítmica.

## CONCLUSIONES

Hemos estudiado una estructura que de por sí, no presenta ningún tipo de incompatibilidades para el despliegue, pero que sufre unos picos importantes de esfuerzos en un momento dado. Analizando el comportamiento llegamos a la conclusión de que este incremento es debido a las fuerzas inerciales durante un proceso de despliegue más o menos rápido, y que el valor máximo varía mucho en función del grado de amortiguamiento.

Así por ejemplo, los axiles de la barra 3, cuando existe amortiguamiento suficiente, no pasan de 200 kg. y la malla despliega en unos 7 sg., pero el caso sin amortiguamiento nos arroja un pico de 1500 kg. lo que significa un elevado factor de multiplicación de 7,5 y un tiempo de despliegue de 2,7 sg.

Por tanto podemos resumir, que en estructuras de despliegue rápido, aun no presentando incompatibilidades geométricas, los efectos inerciales son muy importantes y dependientes del amortiguamiento. Se debe efectuar por tanto un diseño de estas mallas que vaya frenando por sí solo el despliegue de la malla cuando se está alcanzando la posición final.

Por último señalar que un análisis estático o cuasi-estático no podría poner de manifiesto la magnitud de los esfuerzos que se pueden llegar a generar en la estructura, siendo en este caso básico el análisis dinámico efectuado.

---

## 4.3 OTRAS POSIBILIDADES

---

Una vez abierta esta posibilidad de cálculo dinámico de estructuras de barras, las posibles aplicaciones de las rutinas de cálculo son muy variadas, entre ellas solo haremos una sucinta mención de dos de ellas:

- 1.- Vibraciones y frecuencias propias
- 2.- Estructuras tensadas.

---

### 4.3.1 VIBRACIONES Y FRECUENCIAS PROPIAS

---

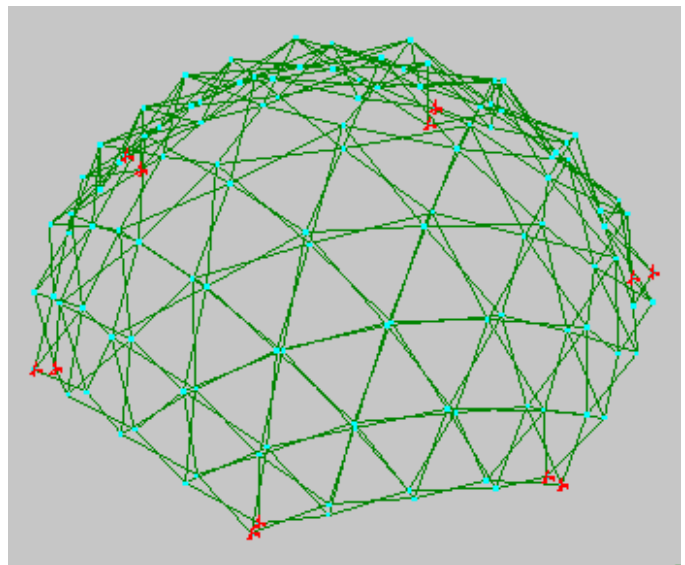
Otra posibilidad que nos abre el programa desarrollado con este trabajo es el análisis de frecuencias propias de mallas, ya sean desplegadas o no.

Esta utilidad ya fue comentada por encima cuando vimos el apartado de validación de resultados. Ahora pasaremos a verla con un par de ejemplos, estudiando dos mallas bastante diferentes. Una de ellas será una desplegable, y para la otra tomaremos una malla espacial tradicional.

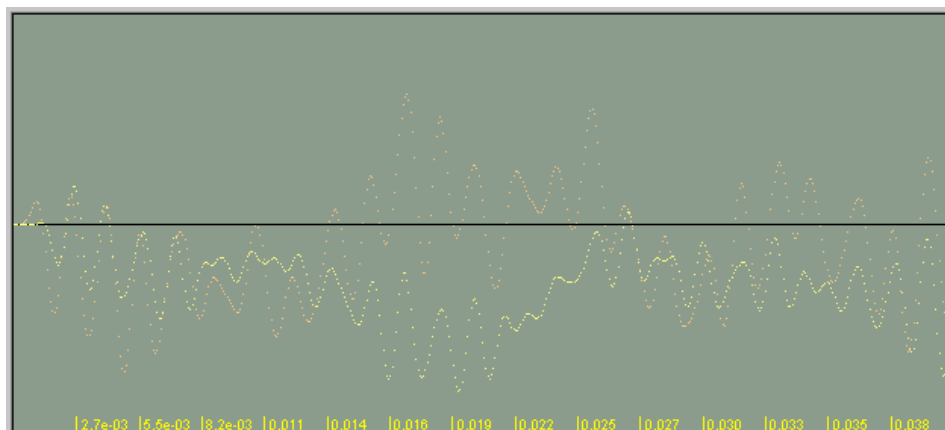
#### MALLA DESPLEGABLE

En este cálculo emplearemos una malla desplegable en forma de cúpula, de 20 m. de luz y 9 m. en su punto más alto. La malla es de módulos triangulares, de orden 4, y con barras de longitud media del orden de 3 metros,  $5.37\text{cm}^2$  de sección transversal,  $21.80\text{ cm}^4$  de inercia y  $2100000\text{ kg/cm}^2$  de Modulo de Young.

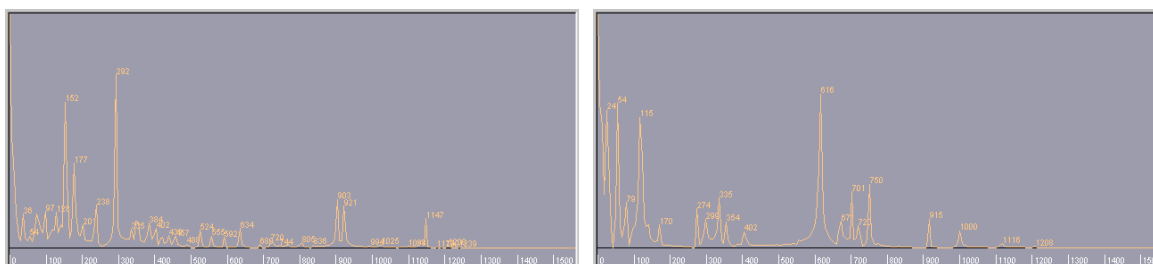
La cúpula se halla totalmente descargada, y para hacer que empiece a vibrar basta simplemente desplazar algunos nodos de su posición de equilibrio.



Bajo este esquema, las ondas de desplazamiento de uno de sus nudos adoptan la forma siguiente:



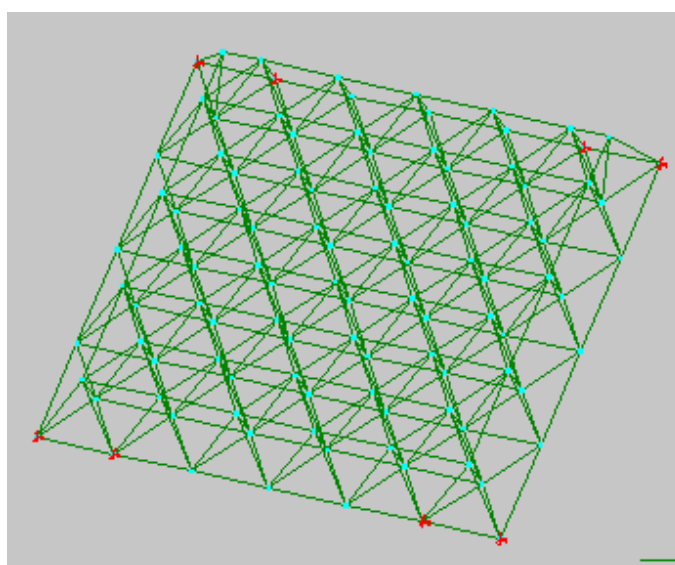
Analizando estas ondas en un par de puntos mediante la transformada rápida de Fourier, obtenemos los gráficos en el dominio de las frecuencias siguientes:



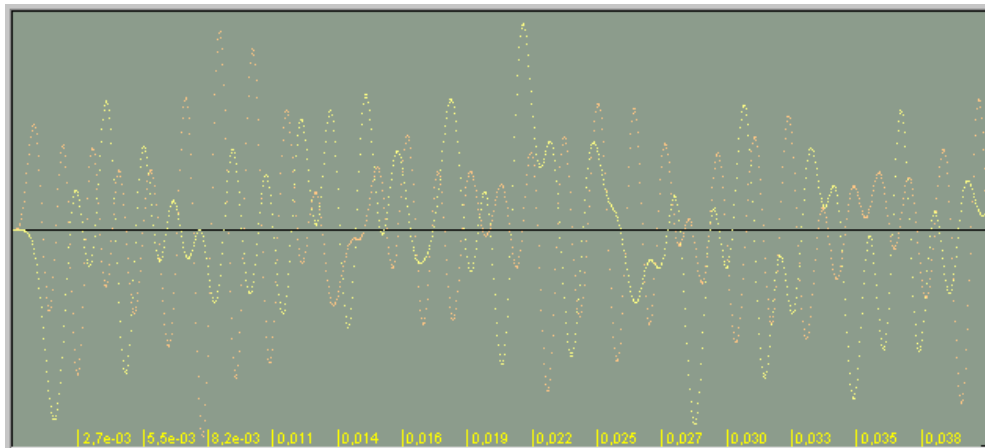
En estos gráficos se puede deducir que las frecuencias de oscilación se sitúan todas por debajo de los 1250Hz, siendo frecuencias dominantes las de 24, 54, 150, 300 y 600 Hz.

## MALLA ESPACIAL

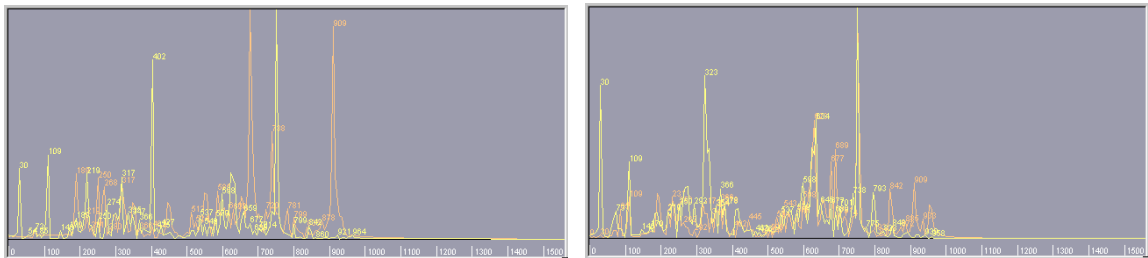
Nuestro modelo ahora es una malla espacial tradicional plana, la hemos elegido de tetraedros por estar más en consonancia con el modelo anterior de triángulos. Tiene unas dimensiones en planta de 18x20 m. y las barras son similares a las de la malla anterior. Tienen una longitud media del orden de 3 metros,  $5.37 \text{ cm}^2$  de sección transversal,  $21.80 \text{ cm}^4$  de inercia y  $2100000 \text{ kg/cm}^2$  de Modulo de Young.



Apoyamos la malla en sus cuatro esquinas, y la hacemos entrar en vibración de forma similar al caso anterior. Ahora la ecuación de onda de desplazamientos de un nudo de la zona central es:



A simple vista ya se aprecia una frecuencia ligeramente superior que en el caso anterior. Pasemos ahora a confirmarlo aplicando de nuevo la transformada de Fourier:



En este caso las frecuencias se sitúan todas por debajo de los 1000 Hz., siendo frecuencias predominantes las de 30, 109, 323, 402, 677, 750 y 909.

## CONCLUSIONES.

Hemos intentado analizar el comportamiento en vibración libre de dos mallas con el fin de determinar sus frecuencias propias. Las dos mallas son muy dispares en cuanto a forma (cúpula y plana) y tipología (desplegable y fija), pero cubren unas luces similares, con unas barras de características iguales y longitudes parecidas.

Comprobamos en primer lugar que el rango de frecuencias que aparecen en ambas es similar, llegando a frecuencias más altas la malla en forma de cúpula (1250 frente a 1000 Hz).

En cuanto a las frecuencias principales, las más bajas, también son relativamente similares, con 24 Hz de la primera frente a 30 Hz de la segunda.

Se trata de mallas por tanto bastante dispares en geometría, que sin embargo no presentan grandes diferencias en el campo de las frecuencias. Aparentemente la mayor rigidez que debería presentar la forma de cúpula queda contrarrestada por la gran rigidez que presenta el módulo tetraédrico en la malla plana, equilibrándose de esta forma ambos comportamientos.

---

## 4.3.2 ESTRUCTURAS TENSADAS

---

Aún alejándonos del tema central de este trabajo, resulta interesante plantear las posibilidades que quedan abiertas con el mismo, entre ellas la generación y cálculo de las llamadas estructuras tensadas, ya sean a base de cables, o superficies de tipo textil, que podremos emular como si fuesen cables.

En España estas estructuras se usaron con profusión con carácter institucional en la Expo 92 celebrada en Sevilla. En ésta destacan las puertas Barqueta e Itálica, el paseo de Europa y el Palenque.



Antes de meternos de lleno en el tema de la generación y cálculo, haremos una introducción a las formas habituales de generación:

### MÉTODOS DE GENERACIÓN.

Para la definición inicial de la forma existen varios métodos estándar que permiten la generación de superficies anticlásticas. A continuación resumimos cuales son [6] y [7] :

#### **a.- Uso de superficies de geometría conocida.**

Tales como paraboloides hiperbólicos, hiperboloides de revolución o bien superficies de revolución que parten de una curva cualquiera con curvatura de signo contrario a la que produzca el proceso de revolución. En este caso los bordes extremos deberán ser rígidos o bien usar cables de suficiente rigidez que van definiendo el perímetro que deseemos.

#### **b.- Utilización de superficies mínimas.**

A un entorno alabeado cerrado en el espacio le podemos ajustar infinitas superficies que cumplan la condición de curvatura de distinto signo, pero puede ocurrir que al entrar en carga no funcione bien. Un tipo de condición que funciona bastante bien es la de superficie



mínima. En este aspecto nos podríamos remitir a los estudios clásicos de Frei Otto sobre superficies equitensionales que nos proporcionan superficies mínimas. Su expresión analítica responde a:

$$\frac{d^2z}{dx^2} + \frac{d^2z}{dy^2} + \left(\frac{dz}{dx}\right)^2 \cdot \frac{d^2z}{dy^2} + \left(\frac{dz}{dy}\right)^2 \cdot \frac{d^2z}{dx^2} - 2 \cdot \frac{dz}{dx} \cdot \frac{dz}{dy} \cdot \frac{d^2z}{dxy} = 0$$

Integrando esta expresión en el contorno cerrado continuo que deseemos nos dará la expresión de la superficie mínima  $z=f(x,y)$ . Como lo único que precisamos es una serie de puntos sobre los que definir una malla, podemos proceder por diferencias finitas pasando a resolver un sistema de ecuaciones lineales cuya resolución nos dará la coordenada  $z$  de cada punto. El problema se complica cuando el contorno delimitador no es continuo o cerrado.

### c.- Método de Relajación Dinámica.

Se parte de una forma conocida, sencilla, para llegar a otra que cumpla nuestras condiciones de contorno. El proceso consiste en realizar cálculos sucesivos desplazando poco a poco los puntos de contorno o imponiendo las cargas que nos interese para ir deformando la superficie. Las superficie de partida puede ser plana, o bien alguna de las superficies sencillas mencionadas en el caso A. Como el proceso iterativo responde a sucesivos cálculos se obtienen superficies que funcionan bien, pero a cambio se llega con una malla muy deformada que es preciso sustituir por otra para efectuar el cálculo.

### d.- Método Variacional.

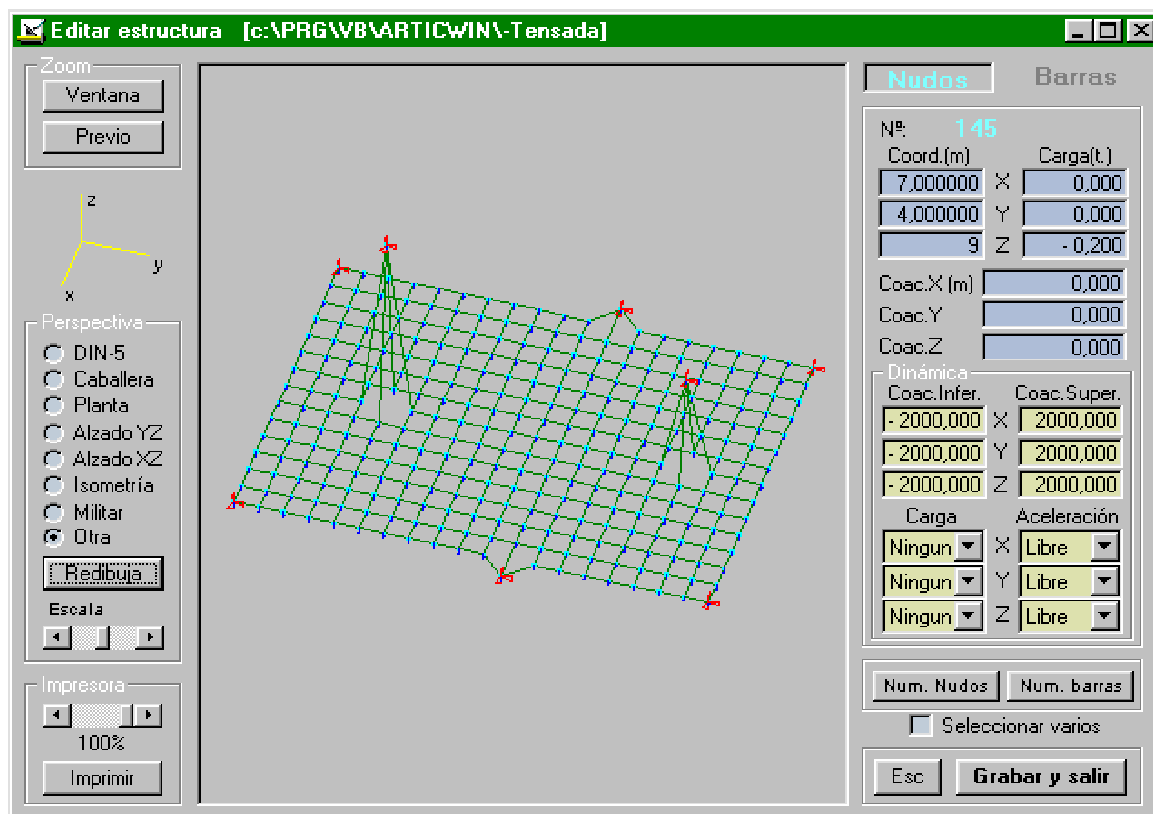
Consiste en la búsqueda de una superficie mínima a partir de unas condiciones previas. Se parte de una forma inicial y se hace un proceso de optimización de esa superficie. Es el método más usado. Para la aproximación de la superficie se somete la misma a un estado de tensión generalizado, se procederá al cálculo y se obtendrá una nueva superficie. Una vez que lleguemos a la forma final se deben eliminar las tensiones internas y casi siempre suele ser preciso un nuevo mallado debido a las grandes deformaciones que puede adquirir la malla inicial.

Evidentemente también existen métodos experimentales, como las maquetas, superficies generadas por pompas de jabón, o el empleado por Gaudí en la Sagrada Familia, que vemos en la fotografía adjunta, y que podríamos emular informáticamente con el programa desarrollado, con lo que conseguiríamos también desarrollar no solo estructuras tensadas sino arcos catenarios cuando le damos la vuelta a la estructura.



## GENERACIÓN DE ESTRUCTURAS TENSADAS.

Con el algoritmo de cálculo que hemos expuesto en apartados anteriores vemos que tenemos abiertas todas las posibilidades para conseguir un método de generación similar a la relajación dinámica. Podemos partir por ejemplo de una malla sencilla plana y a partir de ella ir fijando los puntos que nos interese, y situándolos en las coordenadas que tendrán en la posición final :



En todo este proceso mantendremos las barras con las longitudes que tenían en la malla plana inicial, o mejor todavía, con una longitud ligeramente menor, lo que emulará el efecto de tensado de la estructura. Por tanto ahora tenemos una malla con unas cuantas barras altamente estiradas.

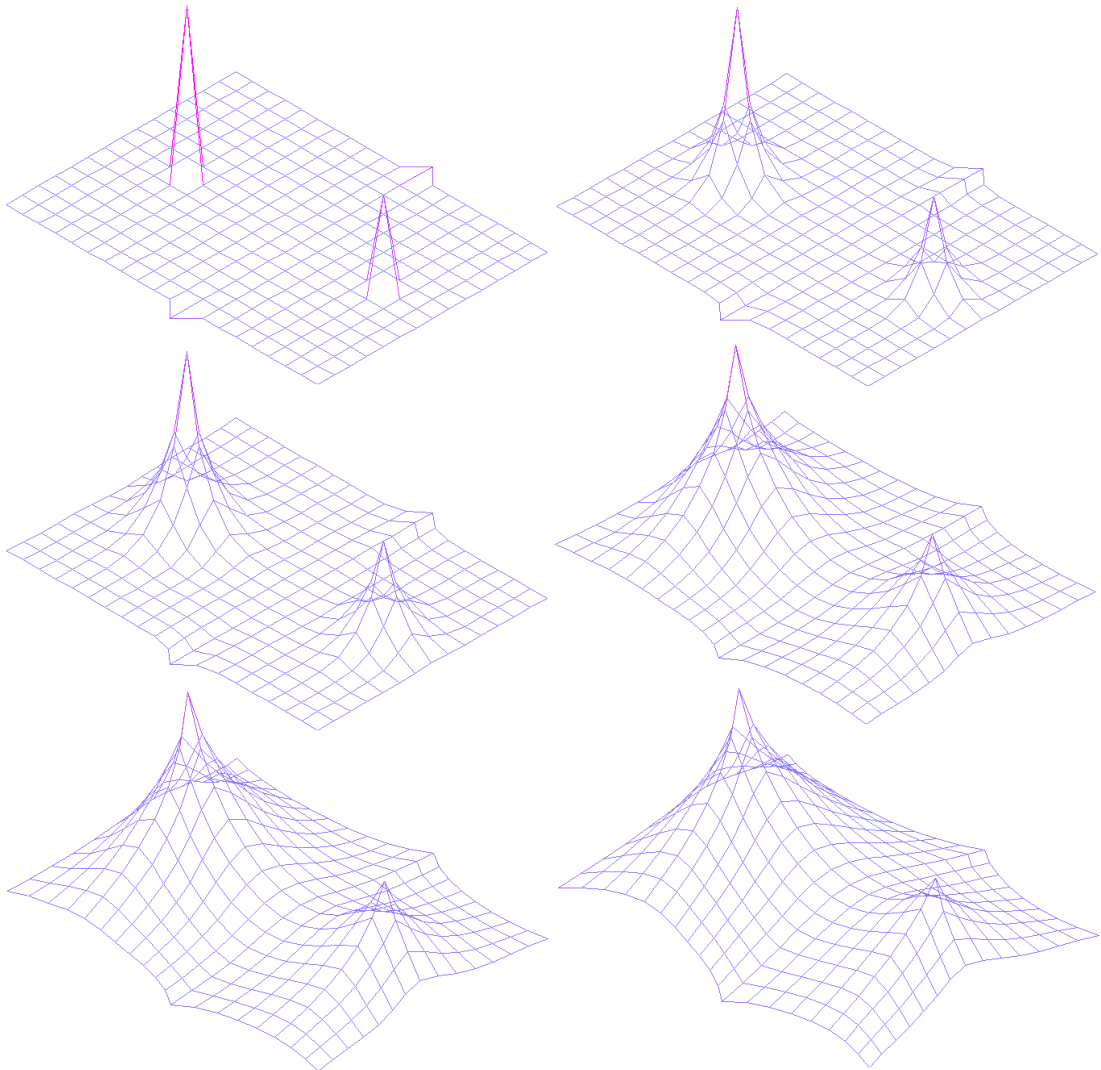
Deberemos usar tipos de barras acordes con lo que será la solución constructiva final. Usualmente tendremos al menos dos tipos: el general de toda la estructura y otro más rígido que formará el perímetro.

Si procedemos al cálculo, el propio efecto de recuperación elástica hará que todas las barras tiendan a recuperar sus longitudes iniciales, y al final del proceso tendremos una superficie próxima a lo que puede ser una estructura tensada típica.

Para que el cálculo evolucione correctamente, deberemos usar un módulo de Young bajo, y unas densidades del material altas, para que de esta forma los nudos no alcancen velocidades demasiado elevadas. Por otra parte, para que el proceso converja en una solución, tenemos que aplicar amortiguamiento durante el cálculo.

El valor adecuado del amortiguamiento es difícil de estimar, debe ser lo suficientemente alto como para evitar que las se disparen y que el proceso alcance un punto estable lo antes posible, pero no demasiado alto que ralentice demasiado este proceso. Una posibilidad interesante es hacer el cálculo por pasos, aplicando al principio amortiguamientos mayores, pues las barras están

demasiado estiradas, y originarán elevadas aceleraciones; e iremos bajando el amortiguamiento en los sucesivos pasos, a medida que la estructura alcance posiciones más estables.



Vemos que el resultado final tiene una apariencia correcta, aunque presenta el inconveniente, ya mencionado al ver el método de relajación dinámica, de que la malla final resulta bastante desfigurada, sobre todo cerca de los picos.

Ante esto la solución será que a partir de la superficie obtenida realicemos un nuevo mallado más equilibrado. Otra opción hubiese sido haber previsto esto inicialmente, usando una malla plana que no fuese totalmente regular, sino que se adaptase a las singularidades que presentará la superficie final. Así, por ejemplo, sería interesante hacer un mallado radial donde vayamos a tener picos pronunciados.

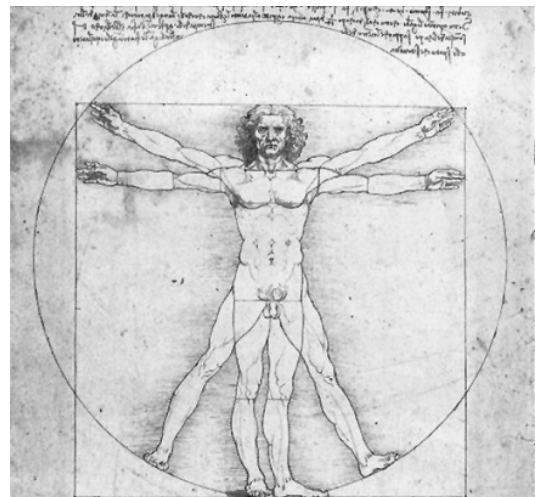
## CONCLUSIONES.

Vemos que el método de cálculo desarrollado deja abiertas las puertas a otro tipo de estructuras, tales como las tensadas, tanto para el proceso de generación como para el cálculo propiamente dicho. También observamos que nuestro método resuelve perfectamente estas estructuras a pesar del alto grado de no linealidad de las mismas.

## REFERENCIAS

- [1] F. Escrig, J. Sánchez and J. Pérez ,“Two Way Deployable Spherical Grids”. Space Structures, Vol.11, No. 1&2, 1996, pp.257-274
- [2] F. Escrig, “Expandable Space Structures”. Space Structures, Vol.1, No. 2, 1985, pp. 79-91
- [3] M. Muñoz and E. López, “Dynamic behavior of deployable stuctures. Influence of the damping effects in the compatibility limitations of grids”. International Congress ICSS-98. Moscow, RUSSIA.
- [4] E. López & M. Muñoz "Influencia de la no linealidad del material en la dinámica de estructuras articuladas de madera". Informes de la Construcción. nº 444 julio/agosto 1996 (pags. 55-61)
- [5] M Muñoz & E. López "Influencia del modelo elástico en la dinámica de estructuras articuladas." Informes de la Construcción. Vol. 49 nº 449 mayo/junio 1997 (pags. 15-21)
- [6] Hans A. Buchholdt and Barry R. McMillan. "Iterative Methods For The Solution Of Pretensioned Cable Structures And Pinjointed Assemblies Having Significan Geometrical Displacements." Proc. 1971 IASS Pacific Symposium Part II on TENSION STRUCTURES ans SPACE FRAMES, Tokyo and Kyoto, pp. 305-316, Paper N° 3-7, 1072, Architectural Institute of Japan.
- [7] J.Monjo Carrió.. "Arquitectura Textil" COAM 91

## 5. Conclusiones.



---

## 5.1 CONCLUSIONES.

---

En este trabajo hemos desarrollado una serie de métodos de cálculo y aplicaciones complementarias entre sí, que nos permitan el estudio dinámico de estructuras de barras y su posterior análisis. A modo de resumen y conclusión final, a continuación veremos qué hemos desarrollado en cada uno de los diversos apartados, y la utilidad de los mismos.

---

### MÉTODO DE CÁLCULO.

---

Las mallas desplegadas son estructuras cuyo cálculo presenta numerosas dificultades. En este trabajo se ha desarrollado un método de cálculo para mallas desplegadas, que permite tener en cuenta efectos inerciales, y por tanto permitirá estudiar mallas de despliegue rápido.

Además podemos trabajar con estructuras altamente no lineales, ya que podemos tener en cuenta el efecto del amortiguamiento o rozamiento en este proceso, y además hemos añadido correcciones a la rigidez debidas al efecto del axil.

Todo ello nos da idea de la potencia de este método, que nos permite calcular mecanismos (mallas desplegando), estructuras isostáticas e hiperestáticas (mallas en la posición final o mallas espaciales tradicionales). Además podemos trabajar con cables, lo que nos permite otra forma de limitar el despliegue de las estructuras, o también el cálculo de otro tipo de estructuras altamente no lineales, como son las estructuras tensadas de cables.

Este método se basa en ecuaciones y métodos de cálculo ampliamente usados y avalados por la experiencia, como son las ecuaciones de movimiento de un sólido, leyes de esfuerzos de barras, y método de las diferencias finitas básicamente.

El método funciona permitiendo visualizar con total claridad el proceso de despliegue de las mallas, y se ha ajustado en la medida de lo posible para un funcionamiento óptimo. Se han verificado los resultados del mismo comparándolos con los de programas de solvencia reconocida, y se puede decir por tanto que nuestro método es totalmente fiable para el tipo de estructuras estudiadas.

---

## ANÁLISIS DE RESULTADOS.

---

Complementariamente al apartado anterior, es preciso procesar la ingente cantidad de datos que un cálculo dinámico nos suministra. Típicamente tendremos datos de la posiciones, velocidades o aceleraciones durante el despliegue, así como los axiles, flectores y cortantes de todas las barras y cargas aplicadas, en función del tiempo.

Como primer método de análisis realizamos un estudio general de la estructura, visualizando el proceso de despliegue de la malla, lo que ya nos da idea del comportamiento de la malla, si ha desplegado totalmente o si hubo problemas durante el despliegue. Todas estas apreciaciones nos servirán para ajustar los parámetros en un posible nuevo cálculo.

A mayores, en este análisis global tenemos información de qué zona de la estructura es la más cargada en cada momento, así como identificación exacta de qué barra es la más solicitada de todas y el valor de este esfuerzo máximo y en qué tiempo se produce. Con todos estos datos ya podemos centrarnos para verificación o dimensionamiento de la estructura en un número reducido de barras que serán las más problemáticas.

Un siguiente análisis desarrollado (análisis local) nos permite trabajar con una barra o nudo a nivel individual y reproducir con todo tipo de detalle su comportamiento durante todo el proceso de desplegado. De esta manera podremos analizar posición, velocidad o aceleración de un nudo, y para las barras los axiles, flectores y cortantes. Básicamente se trata de realizar una gráfica acotada que representa la ecuación de onda de estas variables con respecto al tiempo, lo que nos permite ver en que momentos se dan los picos de valores. Además podemos realizar un análisis de esta onda mediante la transformada de Fourier, lo que nos permite visualizar ese mismo comportamiento, pero ahora en el dominio de las frecuencias y detectar las frecuencias fundamentales de vibración de la estructura.

---

## ESTUDIO DE INCOMPATIBILIDADES.

---

De entre las posibilidades de estudio del método desarrollado cabe destacar la del análisis directo de mallas con incompatibilidades geométricas que dificultan su despliegue.

El método de cálculo permite por sí solo determinar si un tipo de malla presenta incompatibilidades y de que calibre, en función de que fuerzas sea preciso aplicar para que se llegue al final del despliegue, y de los esfuerzos a los que se ve sometida la estructura.

Se abre aquí una forma fácil de evaluar la incidencia del amortiguamiento en la magnitud los esfuerzos, y en la velocidad de despliegue, buscando soluciones en un punto de compromiso entre ambas, aun a falta de desarrollar un paralelismo del rozamiento con el sistema constructivo.

Otra gran ventaja que hemos comprobado en el estudio de estas mallas es que su configuración final más estable viene determinada por el propio cálculo, dependiendo de la geometría de la malla así como del sistema de cargas aplicado. Podemos determinar así de forma precisa las

dimensiones exactas de la malla en este punto, sin depender en absoluto de lo precisos que hallamos sido en el diseño y generación inicial de la malla.

---

## GENERACIÓN DE MALLAS.

---

Ya hemos comentado, que aunque no fundamental, esta es una parte del trabajo necesaria para poder efectuar cálculos. La hemos solucionado además con la propuesta de dos métodos de generación de mallas desarrollados independientemente de los sistemas habituales de generación, abriendo nuevas vías en este campo, que pueden permitir, en estudios posteriores, hacer una comparativa con los métodos tradicionales señalando las ventajas e inconvenientes de cada uno de los métodos.

De entrada, en el sistema de mallas de módulo cuadrangular, la propuesta efectuada presenta la innegable ventaja de una gran simplicidad constructiva, al generar todas las barras exactamente iguales. Puede presentar el inconveniente de grandes distorsiones del módulo hacia las esquinas, pero eso se soluciona en el proceso de diseño no tomando ángulos de apertura demasiado grandes, con lo que el efecto no será apreciable. Por otra parte, la inestabilidad que presenta una vez desplegada, es propia de toda la tipología de mallas que usa este módulo, y se deberán dar soluciones constructivas a la misma, ya sea fijando los puntos de borde, usando cables, o variando ligeramente el diseño de los bordes mediante la disposición de barras que le den esa estabilidad buscada.

El sistema de generación de mallas de módulo triangular aquí presentado tiene la ventaja de su innegable rapidez, ya que ahorramos el proceso de optimización con un sistema de ecuaciones lineales, sustituyéndolo por una definición a priori de como debe ser la forma de crecimiento. Puede que con esta forma de generación la malla no presente menores incompatibilidades que con el sistema usual, cuestión pendiente de estudio, pero sí se ha visto que las que presenta no son excesivas, y pueden superarse aplicando unas fuerzas no muy grandes.

La forma de generación de éste último si nos garantiza que el paquete de barras sea totalmente plegable, y a su vez tenemos una forma desplegada totalmente estable.

---

## IMPLEMENTACIÓN INFORMÁTICA.

---

Los métodos de cálculo, análisis y generación comentados anteriormente se han implementado con éxito en un programa informático que permitirá trabajar con estas estructuras en ordenadores personales, sin excesivos requerimientos de hardware.



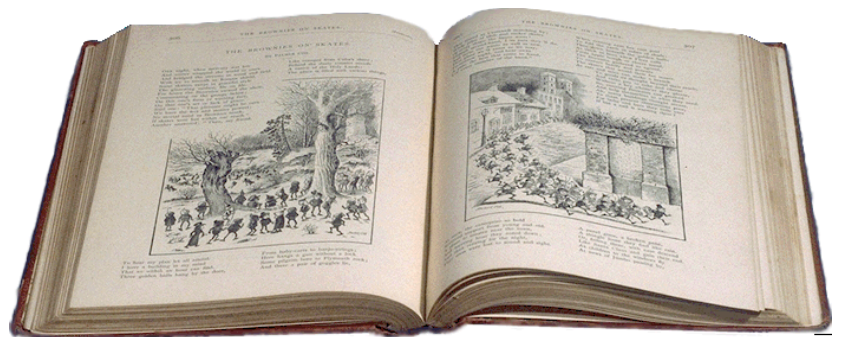
La generación de mallas y el posterior análisis de resultados son tareas relativamente rápidas, siendo la más lenta, obviamente, el propio proceso de cálculo de la estructura, cuestión que se ha intentado paliar optimizando en la medida de lo posible las rutinas de cálculo empleadas. También es esta parte del proceso la que precisa gran cantidad de espacio en disco para el almacenamiento de los datos que se van generando, que son proporcionales al tamaño de la estructura y al tiempo de cálculo elegido.

El programa se ha desarrollado en dos lenguajes: el conocido lenguaje de programación Visual Basic, en su versión 6, debido a sus facilidades gráficas y de depuración, y el módulo principal de cálculo dinámico que se ha escrito en el lenguaje Fortran en su versión 6 de la casa Digital Research. El La razón de haber desarrollado el núcleo de cálculo en Fortran es debido a que el ejecutable muestra ganancia notable de velocidad cálculo, siendo en esta versión del orden del doble de rápido que su correspondiente en Visual Basic. En cuanto a la exactitud de los resultados, estos son prácticamente idénticos, suelen variar únicamente en la última cifra significativa, que es la nº 15, pues estamos trabajando en ambos casos con doble precisión. Otra ventaja que se tuvo en cuenta a la hora de elegir el Fortran es que es más fácil de portar a otras plataformas y sistemas operativos para llegar a hacer cálculos intensivos en los superordenadores de los centros de cálculo.

Como datos anecdóticos comentaremos que el programa en Visual Basic está dividido en 6 módulos y 26 formularios, alcanzando las 21.759 líneas de código. Antes de compilar estos módulos ocupan 823 Kb, y una vez compilado, el ejecutable tiene un tamaño de 932 Kb. Por su parte la versión Fortran del núcleo tiene 3192 líneas ocupando 92 Kb antes de compilar y 564 Kb después.

Por otra parte, lo que realmente precisa gran tamaño de disco, como ya comentamos, son los resultados del cálculo, por poner solo un ejemplo, los diversos cálculos que efectuamos con un único módulo cuadrado en el apartado de validación de resultados, llegó a alcanzar la cantidad de 17 Mb. de disco duro.

## 6. Bibliografía



---

## 6.1 BIBLIOGRAFÍA PRINCIPAL

---

---

### 6.1.1 MALLAS DESPLEGABLES.

---

1.

Autor: ALLEN, M.  
Título: **Toronto Skydome Roof Structure**  
Editorial: Innovative Large Span Structures. Srivastava. IASS International Congress. Toronto. 1992 pp.62-71

2.

Autor: BRUNER, A.  
Título: **Expansible Surface Structure**  
Editorial: 9 Enero, 1968. U.S. Patent 3.362.118

3.

Autor: CALATRAVA, S.  
Título: **Disertaciones**  
Editorial: El Croquis nº 38. Madrid. 1989 pp.4-12

4.

Autor: CALATRAVA, S.  
Título: **Sobre la Plegabilidad de Entramados**  
Editorial: Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993. pp.33-93

5.

Autor: CALATRAVA, S.; ESCRIG, F. & VALCARCEL, J.P.  
Título: **Arquitectura Transformable**  
Editorial: E.T.S.A. de Sevilla. 1993 ISBN 84-600-8583-X

6.

Autor: CANDELA, F.  
Título: **En defensa del Formalismo y otros escritos**  
Editorial: Xarait Editores. Bilbao. 1985

7.

Autor: CANDELA, F.  
Título: **Una estructura metálica Reticulada. El Palacio de los Deportes de Mejico**  
Editorial: I Encuentro Internacional Estructuras Ligeras para grandes Luces. Fundación Emilio Pérez Piñero. Murcia. 1992 pp.9-28

8.

Autor: CANDELA, F.  
Título: **Prólogo, Estructuras Transformables**  
Editorial: Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993 ISBN 84-600-8583-X pp.7-8

9.

Autor: CASINELLO, F.  
Título: **Estructuras Desplegables de Emilio Pérez Piñero**  
Editorial: Pabellón de Murcia 1992.

10.

Autor: CLARKE, RC.  
Título: **The kinematics of a novel deployable space structural system**  
Editorial: Proc. Third Int. Conference on Space Structures. Elsevier. Surrey. 1984 pp.820-822

11.

Autor: CHILTON, J.  
Título: **Geometric transformation concept for expanding rigid Structures**  
Editorial: NASA Contractor Report. NASA CR-1735. 1971

12.

Autor: CHILTON, J.  
Título: **Reciprocal Frame Long Span Structures**  
Editorial: Innovative Large Span Structures. Srivastava. The Canadian Society for Civil Engineering. Montreal, 1992, pp. 100-103.

13.

Autor: CHILTON, J.; CHOO, B.S. & YU, J.  
Título: **Morphology of Reciprocal Frame 3-Dimensional Grillage Structures**  
Editorial: International Association for Shell and Spatial Structures Atlanta. April 1994, pp.1065-1074.

14.

Autor: CHILTON, J.; CHOO, B.S. & COULLIETTE, P.  
Título: **Retractable Roofs using the Reciprocal Frame**  
Editorial: Int. Ass. for Bndge and Structural Eng. Sym. Birmingham. V.K. Sept. 1994, pp. 6.

15.

Autor: CHILTON, J.C.; CHOO, B.S. & POPOVICK, O.  
Título: **Reciprocal Frame. Retractable Roofs**  
Editorial: Spatial Structures: Heritage, Present and Future. Giuliani, SGE Editoriai. Padova, 1995. ISBN 88-86281-10-2, pp. 467-474.

## 16.

Autor: DERUS, D.L.  
Título: **Collapsible Articulated Wall Structure**  
Editorial: U.S. Patent nº 4.580.375. 1386

## 17.

Autor: ESCRIG, F.  
Título: **Sistema modular para la construcción de estructuras espaciales desplegables de barras.**  
Editorial: Mayo, 1984. Patente española nº 532117

## 18.

Autor: ESCRIG, F.  
Título: **Expandable Space Frame Structures**  
Editorial: Space Structures. Elsevier. Surrey. 1984  
ISBN 0-85334-309-8 pp.845-850

## 19.

Autor: ESCRIG, F.  
Título: **Expandable Space Frame Structures**  
Editorial: International Journal of Space Structures. Vol. 1 nº 2. Surrey. 1985 ISSN 0266-3511

## 20.

Autor: ESCRIG, F.  
Título: **Estructuras Espaciales de Barras Desplegables**  
Editorial: Informes de la Construcción, nº 365. 1985. ISSN 0020-0883 pp.3546

## 21.

Autor: ESCRIG, F.  
Título: **Cubrir la Plaza de S. Francisco de Sevilla con Sombrillas Desplegables de Grandes Dimensiones**  
Editorial: Revista de la Edificación, nº 2. Pamplona. 1987 ISSN 0213-8948 pp.60-61

## 22.

Autor: ESCRIG, F.  
Título: **Geometrías de las Estructuras Desplegables de Aspas**  
Editorial: Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993 pp.35-124

## 23.

Autor: ESCRIG, F.  
Título: **Instalaciones deportivas de nueva creación. Carpas desplegables**  
Editorial: Sevilla 2004. Propuesta de Candidatura. Cámara Oficial de Comercio de Sevilla. Sevilla. 1994 Dep. Leg. SE-675-94 pp.113

24.

Autor: ESCRIG, F.  
Título: **Pabellón de Venezuela. Capítulo III**  
Editorial: Instituto de Desarrollo Experimental de la Construcción. Facultad de Arquitectura y Urbanismo. Universidad Central de Venezuela. Caracas. 1993 ISBN 980-00-0621-4

25.

Autor: ESCRIG, F.  
Título: **General Survey of deployability in architecture**  
Editorial: Mobile and Rapid y Assembled Structures II. ESCRIG & BREBBIA. Computational Mechanics Pub. Southampton, 1996. ISBN 1853123986, pp. 3-22.

26.

Autor: ESCRIG, F. & VACCARCEL, J. P.  
Título: **Analysis of Expandable Space Bar Structures**  
Editorial: Membrane Structures. Elsevier. Osaka. 1986 ISBN 0-444-42686-8 pp.269-276

27.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Great size Umbrellas solved with Expandable Bar Structures**  
Editorial: Lightweight Structures. University of New South Gales. Sydney. 1986 ISBN 85823-558-7 pp.676-689

28.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Introducción a la geometría de Estructuras Desplegables de Barras**  
Editorial: Boletín Académico, nº 3. E.T.S.A. Coruña. 1986. ISSN 0213-3474 pp.48-57

29.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Curved Expandable Space Grids**  
Editorial: Non-Conventional Structures. Edimburgo. 1987 ISBN 0-948749-09-1 pp.157-166

30.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Estructuras Espaciales Desplegables Curvas**  
Editorial: Informe de la Construcción, nº 383. Madrid. 1987 ISSN 0020-0883

31.

Autor: ESCRIG, F.; VALCARCEL, J. P. & GIL DELGADO, O.  
Título: **Design of Expandable Spherical Grids**  
Editorial: 10 years of Progress in Shell and Spatial Structures. CEDEX-IASS. Madrid. 1989 ISBN 84-7790-041-8 p.16

32.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **To cover a Swimming Pool with an Expandable Structure**  
Editorial: Rapidly Assembled Structures. P.S. Bulson. Computational Mechanics Publications. 1991. ISBN 1-85312-136-3 pp.273-282

33.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Geometry of Expandable Space Structures**  
Editorial: Space Structures International Journal. Multiscience Publishing Co. Ltd. Vol. 8 nº 1 y 2. 1993 ISSN 0266-3511 pp.71-84

34.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Foldable structures with textile covering**  
Editorial: Techtextil Symposium'94. Frankfurt. 15-17 Junio, 1994

35.

Autor: ESCRIG, F.; VALCARCEL, J. P. & SANCHEZ, J.  
Título: **Arquitectura móvil y de rápido montaje**  
Editorial: 1º Congreso Nacional de Tecnología. E.T.S.A. Madrid. 1994. Tomo II pp.53-62

36.

Autor: ESCRIG, F. & VALCARCEL, J. P.  
Título: **Cubiertas rígidas transformables.**  
Editorial: Hormigón y Acero nº 193. 1994, Madrid. pp. 85-104

37.

Autor: ESCRIG, F.; VALCARCEL, J. P. & SANCHEZ, J.  
Título: **Deployable Structures Squared in Plan. Design and Construction**  
Editorial: Spatial Structures: Heritage, Present and Future. Giuliani SGE Padova 1995. ISBN 88-86281-10-2, pp. 483-492.

38.

Autor: ESCRIG, F.; VALCARCEL, J. P. & SANCHEZ, J.  
Título: **Deployable cover on a swimming Pool in Seville**  
Editorial: IASS Journal of The International Association for shell and Spatial Structures. Vol. 37. 1996, nº1 pp. 39-70.

39.

Autor: ESCRIG, F.; VALCARCEL, J. P. & SANCHEZ, J.  
Título: **Las cubiertas desplegadas de malla cuadrangular**  
Editorial: Boletín Académico, nº20. ETSA Coruña. 1996. ISSN 0213-3474 pp. 37-46.

40.

Autor: FULLER, R. B.  
Título: **Inventions**  
Editorial: St. Martin's Press. N. York. 1983

41.

Autor: GANTES, C. y otros  
Título: **Structural Analysis and Design of Deployable Structures**  
Editorial: Computer & Structures. vol. 32 n° 3/4. 1989, pp.661-669

42.

Autor: GANTES, C.; CONNOR, J.J. & LOGCHER, R.D.  
Título: **Combining Numerical Analysis and Engineering Judgement to Design Deployable Structures**  
Editorial: Computer & Structures. 1991 Vol. 40, n° 2, pp. 431-440.

43.

Autor: GANTES, C.; CONNOR, J.J. & LOGCHER, R.D.  
Título: **Geometric and Structural Design Considerations for Deployable Space Frames**  
Editorial: Rapidly Assembled Structures. P.S. Bulson. Computational Mechanics Pub. Southampton, U.K. 1991 pp.249-260

44.

Autor: GANTES, C.; CONNOR, J.J. & LOGCHER, R.D.  
Título: **A simple Friction Model for Scissor-Type Mobile Structures**  
Editorial: ASCE. Journal of Engineering Mechanics. Vol. 119, n° 3, March 1993, pp. 456-475.

45.

Autor: GANTES, C.  
Título: **Geometric constraints in assembling polygonal deployable units to form multi-unit structural systems**  
Editorial: Space Structures. Parke and Howard. Thomas Telford. London. 1993 pp.793-803

46.

Autor: GANTES, C. y otros  
Título: **Deployability Condition for Curved and Flat, Polygonal and Trapezoidal Deployable Structure**  
Editorial: International Journal of Space Structures. Vol. 8 n° 1 y 2. Multi-science Publishing Co. Ltd. 1993. ISSN 0266-3511 pp.97-106

47.

Autor: GANTES, C. y otros  
Título: **Geometric Design of Deployable Structures with Discrete Joint Size**  
Editorial: International Journal of Space Structures. Vol. 8 n° 1 y 2. Multi-science Publishing Co. Ltd. 1993. ISSN 0266-3511 pp.107-118

48.

Autor: GANTES, C.; CONNOR, J.J. & LOGCHER, R.D.  
Título: **An equivalent continuum Model for Deployable Flat Lattice Structures**  
Editorial: ASCE. Journal of Aerospace Structures. Vol. 7, n° 1, January 1994, pp. 72-91



49.

Autor: GANTES, C.; CONNOR, J.J. & LOGCHER, R.D.  
Título: **A Systematic Design Methodology for Deployable Structures**  
Editorial: Int. Journal of Space Structures. Vol. 9, nº 2, 1994. Multi-Science Publishing Co. Ltd. ISSN 0266-3511, pp. 67-86.

50.

Autor: GANTES, C.  
Título: **Analytical predictions of the snap-through characteristics of deployable structures**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Pub. Southampton 1996. ISBN 185312398 6, pp. 83-92

51.

Autor: C. J. Gantes, A. N. Kounadis, J. Mallis  
Título: **Approximate dynamic buckling loads of discrete systems via geometric considerations of their energy surface**  
Editorial: Computational Mechanics 21. 1998. pp.398-402

52.

Autor: GANTES, C.  
Título: **Deployable Structures. Analysis and Design.**  
Editorial: Wit Press. 2001

53.

Autor: HATATO, T. y otros  
Título: **Design Loads Estimation for a Retractable Roof System**  
Editorial: Innovative Large Span Structures. Srivastava. IASS International Congress. Toronto. 1992 pp.533-544

54.

Autor: HERNANDEZ, C. H.  
Título: **Deployable Structures**  
Editorial: Master's Thesis M.I.T. Dep. of Architecture. 1987

55.

Autor: HERNANDEZ, C. H.  
Título: **Estructuras transformables. Estran I**  
Editorial: Tecnología y Construcción nº 4. Caracas. 1988 pp.103-118

56.

Autor: HERNANDEZ, C. H.  
Título: **Realización de estructuras desplegables**  
Editorial: I Encuentro Internacional Estructuras ligeras para grandes luces. Fundación Emilio Pérez Piñero. Murcia. 1992. pp.235-242

57.

Autor: HERNANDEZ, C. H.  
Título: **New ideas on deployable structures**  
Editorial: Mobile and Rapidly Assembled Structures II. Escng & Brebbia. Computational Mechanics Publ. Southampton 1996. ISBN 1853123986 pp. 63-72.

58.

Autor: HERNANDEZ, C. H. & ZALEWSKI, W.  
Título: **Deployable Structures**  
Editorial: Rapidly Assembled Structures. P.S. Bulson. Computational Mechanics Pub. Southampton, U.K. 1991 pp.237-248

59.

Autor: HERNANDEZ, C. & ESCRIG, F.  
Título: **El Pabellón de Venezuela en la EXP0'92, una estructura desplegable en Aluminio**  
Editorial: Informes de la Construcción, nº 423. Madrid. 1994 ISSN 0020-0883 pp.61-69

60.

Autor: HOBERTMAN, Ch.  
Título: **Art and Science of folding structures**  
Editorial: Sites Vol. 24. N.Y. 1992 pp.34-53

61.

Autor: HOBERTMAN, Ch.  
Título: **Folding in Architecture**  
Editorial: Architectural Design. March-April 1993, pp.56-59

62.

Autor: JACQUEMIN, G.G. et al.  
Título: **Development of assembly and joint concepts for erectable space structures**  
Editorial: NASA CR 3131. Diciembre, 1980

63.

Autor: KARNI, E.  
Título: **Retractable Spatial Structures for Swinning Pool Enclosures**  
Editorial: Int. Journal of Space Structures. Vol. 10, nº 4, 1995. Multi-Science Publishing Co. Ltd. ISSN 0266-3511, pp. 231-236.

64.

Autor: KAWAGUCHI, M.  
Título: **Mecanismo de estructuras espaciales ligeras**  
Editorial: I Encuentro Internacional Estructuras Ligeras para grandes Luces. Fundación Emilio Pérez Piñero. Murcia. 1992 pp.53-84

65.

Autor: KAWAGUCHI, K. & MANGAI, Y.  
Título: **Numerical anlysis for folding of space structures**  
Editorial: Space Seructures. Parke, Thomas Telford. London. 1993 pp.813-823

66.

Autor: KWAN, A.S.K. & PELLEGRINO, S.  
Título: **Matrix Formulation of Macro-Elements for Deployable Structures**  
Editorial: Computer & Structures Vol. 50 nº 2. 1994 pp.237-254

67.

Autor: KWAN, A.S.K. & PELLEGRINO, S.  
Título: **A New Concept for Large Deployable Space Frames**  
Editorial: Int. Journal of Space Structures. Vol. 9, nº 3, 1994. Multi Science Publishing Co. Ltd. ISSN 0266-3511, pp. 153-162.

68.

Autor: LOPEZ, E. & MUÑOZ, M.  
Título: **Dynamic Behavior of Deployable Structures. Influence of the Damping Effects in the Compatibility Limitations of Grids**  
Editorial: Spatial Structures In New And Renovation Projects of Buildings and Construction. Proceedings ICSS-98. 1998. Moscow, Russia. pp.434-440.

69.

Autor: LOPEZ, E. & MUÑOZ, M.  
Título: **Comparative Study of the Geometric Incompatibilities in the Dinamica of Expandable Structures**  
Editorial: IASS-IACM 2000. Fourth International Colloquium on Computation of Shell & Spatial Structures June 2000. Chania-Crete, Greece.

70.

Autor: MC NULTY, O.  
Título: **Foldable Space Structures'**  
Editorial: Proc. Int. Symposium on Membrane Structures and Space Frames. IASS. Osaka. 1986 pp.277-284

71.

Autor: MORALES, J.; SANCHEZ, J. & ESCRIG F.  
Título: **Real time animation of architectural mobile elements**  
Editorial: Visualization and Intelligent Design in Engineering and architecture II. HERNANDEZ & BREBBIA. Computational Mechanics Publications. Suthampton 1995. pp. 175-182.

72.

Autor: MORALES, J.; SANCHEZ, J. & ESCRIG F.  
Título: **Geometry of deployable structures generated by computer-aided design**  
Editorial: Mobile and Rapidly Assembling Structures II. Escrig & Brebbia. Computational Mechanics Pub. Southampton 1996. ISBN 185312398 6, pp. 253-258

73.

Autor: PELLEGRINO, S. & YOU, Z.  
Título: **Foldable ring structures**  
Editorial: Space Swctures. Parke, Thomas Telford. London. 1993 pp.783-792

74.

Autor: PEREZ BELDA, E.  
Título: **Constructive problems in the deployable structures of Emilio Pérez Piñero**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton 1996. ISBN 1853123386 pp. 23-34.

75.

Autor: PIÑERO, E.  
Título: **Patentes españolas 266.801, 283.201, 311.901**

76.

Autor: PIÑERO, E.  
Título: **U.S. Patent. 3.185.164**

77.

Autor: PIÑERO, E.  
Título: **Teatro Ambulante.**  
Editorial: Arquitectura nº 30. Junio 1961, Madrid. pp. 27-33.

78.

Autor: PIÑERO, E.  
Título: **Estructuras reticulares**  
Editorial: Arquitectura nº 112. Madrid. Abril, 1968 pp.1-9

79.

Autor: PIÑERO, E.  
Título: **Estructures reticulées**  
Editorial: L'Architecture d'aujourd'hui. Vol. 141. Diciembre, 1968 pp.76-81

80.

Autor: PIÑERO, E.  
Título: **Teatros Desmontables**  
Editorial: Informes de la Construcción nº 231. Madrid 1371, pp. 34-43

81.

Autor: PIÑERO, E. P. y otros.  
Título: **La Obra de E.P.P.**  
Editorial: Arquitectura nº 163-164. Julio-Agosto 1972, Madrid. pp. 1-28.

82.

Autor: PIÑERO, E. P. & ESCRIG, E.  
Título: **Las Estructuras de Emilio Pérez Piñero**  
Editorial: Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993 pp.9-32

83.

Autor: PUERTAS DEL RIO, L.  
Título: **Estructuras Espaciales Desmontables y desplegadas**  
Editorial: Informes de la Construcción. Vol. 42 nº 403. Madrid. 1990 pp.43-53

84.

Autor: PUERTAS DEL RIO, L.  
Título: **Space Frames for Deployable Domes**  
Editorial: IASS Journal of the International Association for shell and spatial structures. Vol. 32, nº 2. 1991,pp. 107-113.

85.

Autor: PUERTAS DEL RIO, L.  
Título: **Las estructuras de Emilio Pérez Piñero**  
Editorial: I Encuentro Internacional Estructuras Ligeras para grandes luces. Fundación Emilio Pérez Piñero. Murcia. 1992 pp.29-48

86.

Autor: ROSENFELD, Y. & LOGCHER, R.  
Título: **New Concepts for Deployable-Collapsible Structures**  
Editorial: International Journal of Space Structures. Vol. 3. 1388  
pp.20-32

87.

Autor: ROSENFELD, Y. y otros  
Título: **A Prototype 'clicking' scissor-Link Deployable Structure**  
Editorial: International journal of Space Structures. Vol. 8 nº 1 y 2. Multi-science Publishing Co. Ltd. 1993 ISSN 0266-3511 pp.85-96

88.

Autor: SANCHEZ CUENCA, L.  
Título: **Flexible Geometry for Space Structures**  
Editorial: Spatial Structures: Heritage, Present and Future. Giuiani; SGE Editorial. Padova, 1995. ISBN 88-86281-10-2, pp. 239-246.

89.

Autor: SANCHEZ CUENCA, L.  
Título: **Geometric models for expandable structures**  
Editorial: Mobile and Rapidly Assembling Structures II. Escrig & Brebbia. Computational Mechanics Pub. Southampton1996. ISBN 185312398 6, pp. 93-102

90.

Autor: SANCHEZ SANCHEZ, J.  
Título: **Estructuras Desplegables de Aspas para Mallas Polédricas Curvas.**  
Editorial: Tesis Doctoral. Universidad de Sevilla. E.T.S. Arquitectura. Septiembre 1996

91.

Autor: SANCHEZ, J.; ESCRIG, F. & VALCARCEL, J.P.  
Título: **Analysis of reduced scale models of x-frame structures**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton 1996. ISBN 1853123986 pp. 55-62.

92.

Autor: SANCHEZ, J.; ESCRIG, F. & VALCARCEL, J.P.  
Título: **The adventure of covering a swimming pool with a x-frame structure**  
Editorial: Mobile and Rapidly Assembling Structures II. Escrig & Brebbia. Computational Mechanics Pub. Southampton 1996.

93.

Autor: SASTRE, R.  
Título: **Expandable arches**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton 1996. ISBN 1853123986 pp. 123-131.

94.

Autor: SHAN, W.  
Título: **Foldable space structures**  
Editorial: Ph. D. Thesis. University of Surrey. 1990

95.

Autor: SHAN, W.  
Título: **Computer analysis of foldable Structures**  
Editorial: Computer and Structures Vol. 42 nº6. 1992 pp.903-912

96.

Autor: SHAN, W.  
Título: **Configuration Studies of foldable Structures**  
Editorial: Space Structures. Parke, Thomas Telford. London. 1993 pp.824-832

97.

Autor: TODA, I. & ISHII, K.  
Título: **Design of Retractable Roofs**  
Editorial: Innovative Large Span Structures. Srivastava. IASS International Congress. Toronto. 1992 pp.510-520

98.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Analysis of Curved Expandable Space Bar Structures**  
Editorial: 10 years of Progress in Shell and Spatial Structures. CEDM-IASS. Madrid. 1989 ISBN 84-7730-041-8 p.10

99.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Un primer planteamiento de estructuras desplegadas. El Código I de Madrid de Leonardo da Vinci**  
Editorial: Boletín Académico de la E.T.S.A. nº 10. La Coruña. 1989. ISSN 0213-3474 pp.13-20

## 100.

Autor: VALCARCEL, J.P. y otros  
Título: **Ideas para un Pabellón en Expo'92 Sevilla. Principado de Asturias**  
Editorial: Asturias'92. Oviedo. 1991 pp.87-95

## 101.

Autor: VALCARCEL, J.P.; ESCRIG, F. & MARTIN, E.  
Título: **Expandable Space Structures with Self-folding textile Cover**  
Editorial: Rapidly Assembled Structures. Computational Mechanics Publications. 1991. ISBN 1-85312-136-3 pp.283-295

## 102.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **La obra arquitectónica de Emilio Pérez Piñero**  
Editorial: Boletín Académico de la E.T.S.A. nº 16. La Coruña. 1992. ISSN 0213-3474 pp.3-12

## 103.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Sistemas de Cálculo de Estructuras desde Entornos CAD**  
Editorial: Tribuna de la Construcción, nº 3. Junio, 1992. Dep. Leg. V-973-1992 pp.19-26

## 104.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Large Span Expandable Domes**  
Editorial: Innovative Large Span Structures. The Canadian Society for Civil Engineering. 1992. ISBN 0-921303-37-8 pp.619-631

## 105.

Autor: VALCARCEL, J.P.; ESTÉVEZ, J. & ESCRIG, F.  
Título: **Expandable Triangular Cylindrical Vaults**  
Editorial: Métodos Numéricos en Ingeniería y Ciencias Aplicadas. Barcelona. 1992. ISBN 84-87867-16-2 pp.327-337

## 106.

Autor: VALCARCEL, J.P.  
Título: **Sistemas de Cálculo de Estructuras desde Entornos de CAD**  
Editorial: III Jornadas de Informática Aplicada a la Arquitectura. E.T.S.A. de Sevilla. 1992 pp.31-36

## 107.

Autor: VALCARCEL, J.P.  
Título: **Cúpulas de grandes luces con Módulos de aspas**  
Editorial: I Encuentro Internacional Estructuras ligeras para grandes luces. Fundación Emilio Pérez Piñero. Murcia. 1992. pp.109-136

## 108.

Autor: VALCARCEL, J.P.  
Título: **Cálculo de Estructuras Desplegables de Barras**  
Editorial: Arquitectura Transformable. Textos de Arquitectura. E.T.S.A. de Sevilla. 1993. pp.125-152

109.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Cálculo no lineal de Estructuras desplegadas con barras curvas**  
Editorial: Métodos Numéricos en Ingeniería. Sociedad Española de Métodos Numéricos en Ingeniería. Barcelona. 1993. ISBN 84-87867-23-5 pp.400-410

110.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Estructuras espaciales. Entre la imaginación y la crisis**  
Editorial: Boletín Académico de la E.T.S.A. de La Coruña nº 17. 1993. pp.35-45

111.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Expandable Domes with incorporated Roofing Elements'**  
Editorial: Space Structures. Parke. Thomas Telford, London. 1993. ISBN 07277-1968-8

112.

Autor: VALCARCEL, J.P. & ESCRIG, F.  
Título: **Pionering in expandable structures: The Madrid I notebook by Leonardo da Vinci**  
Editorial: Bulletin of the IASS, Vol. 35, nº 114. Madrid. April, 1994. ISSN 0304-3622 pp.33-45

113.

Autor: VALCARCEL, J.P.; ESCRIG, F. & MARTIN, E.  
Título: **Expandable Structures with incorporated Roofing Elements**  
Editorial: Spatial, Lattice and Tension Structures. ASCE-IASS. Atlanta. 1994 ISBN 0-87262-953-8

114.

Autor: VALCARCEL, J.P.; ESCRIG, F.; MARTIN, E. & VAZQUEZ, J.  
Título: **Visualization of expandable structures with self-folding roofing plates**  
Editorial: Visualization and Intelligent Design in Engineering and Architecture II. Hernández & Brebbia. Computational Mech. Pub. 1995. pp.191-198.

115.

Autor: VALCARCEL, J. P. & ESCRIG, F.  
Título: **Recent advances in the analysis of expandable structures**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton 1996. ISBN 1853123986 pp.45-54

116.

Autor: VALCARCEL, J.; ESCRIG, F.; MARTIN, E.; DOMINGUEZ, E.; JAUREGUIZAR, F.  
Título: **Recent advances in analysis of expandable structures: an improved method**  
Editorial: Spatial Structures In New And Renovation Projects of Buildings and Construction. Proceedings ICSS-98. 1998. Moscow, Russia. pp.192-199.

117.

Autor: VERHEYEN, H.F.  
Título: **Expandable Polyhedral Structures based on Dipolygonios**  
Editorial: Proc. Third Int. Conference on Space Structures. Elsevier. Surrey. 1984 pp.88-93



## 118.

Autor: WAGNER, Rosemarie  
Título: **Exhibition Pavillion**  
Editorial: Spatial Structures: Heritage, Present and Future. Giuliani, SGE Editorial. Padova, 1995. ISBN 88-86281-10-2, pp.261-268

## 119.

Autor: YOU, Z. & PELLEGRINO, S.  
Título: **Deployable Mesh Reflector**  
Editorial: Spatial Lattice and Tension Structures. IASS International Symposium. Atlanta. 1994 pp.103-112

## 120.

Autor: YOU, Z. & PELLEGRINO, S.  
Título: **Dynamic Deployment of the CRTS Reflector**  
Editorial: Structural Dynamics and Materials Conference. AIAA. South Carolina. 1994

## 121.

Autor: YOU, Z. & PELLEGRINO, S.  
Título: **New Solutions for foldable roff structures**  
Editorial: Mobile and Rapidly Assembled Structures II. Escrig & Brebbia. Computational Mechanics Publ. Southampton. 1996. ISBN 1853123986 pp.35-44

## 122.

Autor: ZANARDO, A.  
Título: **Two dimensional articulated systems developable on a single or double curvature**  
Editorial: Meccanica 21. 1986, pp. 106-111.

## 123.

Autor: ZEIGLER, T.  
Título: **Collapsible Self-Supporting Structures**  
Editorial: U.S. Patent nº 4.437.275. 1981

## 124.

Autor: Varios  
Título: **Mobile and Rapidly Assembled Structures III**  
Editorial: F. ESCRIG, University of Seville, Spain, and C.A. BREBBIA, Wessex Institute of Technology, UK. Advances in Architecture, Vol 11. ISBN: 1-85312-817-1 2000

---

## 6.1.2 CÁLCULO DINÁMICO Y NUMÉRICO.

---

1.

Autor: ALARCÓN, E. ; BREBBIA, C. ; HACAR, M.A. ; SAMARTÍN, A.  
Título: **Curso de cálculo dinámico en la ingeniería civil**  
Editorial: Edix. S.A. Madrid

2.

Autor: ALVAREZ BALERIOLA, J. I.  
Título: **Introducción a la Dinámica de Estructuras**  
Editorial: Monografías del Instituto Eduardo Torroja. nº 535 . Madrid. 1978  
Notas: Estudio profuso, pero se ciñe a un grado de libertad.

3.

Autor: BEER, F. ; JOHNSTON, E  
Título: **Mecánica vectorial para ingenieros.**  
Editorial: Mc Graw-Hill, Mexico 1.990  
Notas: Es un libro muy didáctico, para los primeros niveles de ingeniería.

4.

Autor: BIGGS, J.M.  
Título: **Introduction to Structural Dynamics.**  
Editorial: Mc Graw-Hill, New York 1.964  
Notas: Es un libro práctico en el que se estudian con claridad los problemas de un grado de libertad, así como los de varios por análisis modal.

5.

Autor: CLARK, S.K.  
Título: **Dinámica de elementos continuos**  
Editorial: Reverté, S.A. Barcelona 1975  
Notas: Libro muy completo en el que se estudian sistemas de uno, dos y varios grados de libertad, así como sistemas continuos. Realiza una introducción a los métodos numéricos de resolución, por elementos finitos e integración numérica. Breve exposición del comportamiento sísmico.

6.

Autor: CLOUGH, R.W. ; PENZIEN, J.  
Título: **Dynamics of structures**  
Editorial: Mc Graw - Hill. New York

7.

Autor: CHAPRA, S.C. & CANALE, R.P.  
Título: **Métodos Numéricos para Ingenieros**  
Editorial: Mc Graw - Hill. México. 1999. ISBN 970-10-2008-1

8.

Autor: CRAIG, R.R.  
Título: **Structural Dynamics**  
Editorial: John Wiley & Sons, Inc. U.S.A. 1981  
Notas: Desarrolla los temas típicos. Interesante el estudio del movimiento de varillas.

9.

Autor: CRAVEUR, J. C.  
Título: **Modelisation des structures. Calcul par éléments finis.**  
Editorial: Masson S.A. Paris. 1996  
Notas: Buena, concisa y clara aproximación al tema de los elementos finitos. Incluye una pequeña introducción final al estudio dinámico.

10.

Autor: CSI  
Título: **"SAP2000. ANALYSIS REFERENCE". Volume 1**  
Editorial: Computers and Structures, Inc. California. 1995  
Notas: Manual de referencia del programa de cálculo por elementos finitos SAP 2000.

11.

Autor: DEN HARTOG, J.P.  
Título: **Mechanical vibrations**  
Editorial: Mc Graw - Hill Book Company, Inc. New York

12.

Autor: FREEBERG, C.R. ; KEMLER, E.N.  
Título: **Elements o mechanical vibrations**  
Editorial: John Wiley and Sons, Inc. New york

13.

Autor: GOULD ; ABU-SITTA.  
Título: **Dinamic Response of Structures to Wind & Earthquake Loading.**  
Editorial: Halsted Press. 1.980.  
Notas: Libro interesante en lo referente al efecto del viento sobre edificios. Es de una lectura innecesariamente difícil.

14.

Autor: HANSEN, H.M. ; CHENEA, P.F.  
Título: **Mechanics of vibration**  
Editorial: John Wiley and Sons, Inc. New york

15.

Autor: HATTER, D.J.  
Título: **Matrix Computer Methods of Vibration Analysis.**  
Editorial: London Butterwoths. 1.973  
Notas: Libro que expone de forma muy breve los métodos matriciales aplicados al análisis dinámico de estructuras para su posterior implementación en el ordenador. Muy útil para prácticas.

## 16.

Autor: JACOBSEN, L.S. ; AYRE, R.S.  
Título: **Engeneering vibrations**  
Editorial: Mc Graw - Hill Book Company, Inc. New York  
Notas: Libro muy completo en el que se estudian sistemas de uno, dos y varios grados de libertad, así como sistemas continuos. Recomendable.

## 17.

Autor: KOLSKY, H.  
Título: **Stress Waves in solids**  
Editorial: Mc Graw - Hill Book Company, Inc. New York, 1970

## 18.

Autor: LANDAU & LIFSHITZ.  
Título: **Física teórica. Mecánica**  
Editorial: Reverté S.A. 1965  
Notas: Libro muy completo a la vez que breve sobre movimientos de cuerpos.

## 19.

Autor: LAFITA BABIO, F. ; MATA CORTÉS, H.  
Título: **Introducción a la teoría de vibraciones mecánicas**  
Editorial: Labor S.A. Barcelona, 1968  
Notas: Estudia sistemas de uno y varios grados de libertad.

## 20.

Autor: LAURA, P. A. ; POMBO, J. L.  
Título: **Introducción a la dinámica estructural**  
Editorial: Fundación para la educación, la ciencia y la cultura. Cuenos Aires. 1980  
Notas: Estudia vibraciones de sistemas continuos. Dinámica de barras, membranas y placas y análisis experimental.

## 21.

Autor: LOPEZ, E. & MUÑOZ, M.  
Título: **Influencia de la no linealidad del material en la dinámica de estructuras articuladas de madera**  
Editorial: Informes de la Construcción. nº 444 julio/agosto 1996 pp.55-61.

## 22.

Autor: MARION, J. B.  
Título: **Dinámica clásica de las partículas y sistemas.**  
Editorial: Reverté, S. A. 1.975.  
Notas: Expone la mecánica de Newton y principio de Hamilton de forma muy clara.

## 23.

Autor: MERIAM J. L.  
Título: **Dinámica**  
Editorial: Reverté, S. A. 1.975.  
Notas: Planteamientos iniciales de cinética y cinemática.

## 24.

Autor: MESKOURIS, KONSTANTIN  
Título: **Structural Dynamics**  
Editorial: Ernst & Sohn. 2000  
Notas: Estudia sistemas de uno y varios grados de libertad, así como sistemas continuos. También estudia la respuesta estructural frente terremotos.

## 25.

Autor: NEWLAND, D.E.  
Título: **An Introduction to Random Vibrations and Spectral Analysis.**  
Editorial: Longman. Londres. 1.975.  
Notas: Planteamiento simple de un tema que casi siempre puede encontrarse desarrollado en términos muy complejos. Recomendable.

## 26.

Autor: PAZ, MARIO  
Título: **Dinámica estructural. Teoría y Cálculo.**  
Editorial: Reverté S.A. 1992. Tercera edición. ISBN 84-291-4854-X  
Notas: Estudio del comportamiento dinámico de estructuras porticadas. Incluye las estructuras tridimensionales y programas de ejemplo. Muy recomendable.

## 27.

Autor: PAZ, MARIO  
Título: **Structural Dynamics. Theory and computation.**  
Editorial: Van Nostrand Reinhold Company. New York. 1997. Cuarta edición.  
Notas: Estudio del comportamiento dinámico de estructuras porticadas. Incluye las estructuras tridimensionales y programas de ejemplo. Muy recomendable.

## 28.

Autor: SIMIU  
Título: **Wind effects on structures.**  
Editorial: Scanlan. 1986  
Notas: Estudio del efecto dinámico del viento en las estructuras.

## 29.

Autor: THIMOSHENKO, S.  
Título: **Vibration Problems in Engineering**  
Editorial: D. Van Nostrand Company, Inc., Princeton.

## 30.

Autor: WARBURTON, G.  
Título: **Dinamical Behaviour of Structures.**  
Editorial: Pergamon. 1.977.  
Notas: Texto elemental pero muy claro.

## 31.

Autor: WILLIAM T. THOMSON.  
Título: **Teoría de vibraciones**  
Editorial: Prentice / Hall internacional. 1982

## 32.

Autor: WILLIAM T. THOMSON

Título: **Theory of vibrations with applications**

Editorial: Chapman & Hall. London, 1993

Notas: Estudia sistemas vibrantes lineales y no lineales. Métodos numéricos aproximados de resolución.

---

## 6.2 BIBLIOGRAFÍA SECUNDARIA

---

---

### 6.2.1 MALLAS ESPACIALES.

---

1.

Autor: BORREGO, J.  
Título: **Space Grid Structures.**  
Editorial: M.I.T. Press. Mass. 1.972  
Notas: Es más un texto de construcción que propiamente de estructuras.

2.

Autor: ESPINOSA MALDONADO, S.P. ; GACHET GARCÍA, J.J.  
Título: **Generación de datos de sistemas estructurales**  
Editorial: Tesis de grado. Escuela politécnica nacional de Quito, Ecuador, Mayo 1986

3.

Autor: ESTÉVEZ CIMADEVILA, F.J.  
Título: **Análisis no lineal de mallas espaciales de doble capa con dimensionado estricto**  
Editorial: Tesis doctoral. Universidad de La Coruña, 1990

4.

Autor: MAKOWSKY, Z.S.  
Título: **Estructuras espaciales de acero.**  
Editorial: Gustavo Gili. Barcelona. 1.972.  
Notas: Libro clásico sobre estructuras espaciales, más interesante por su completa exposición de fotografías y dibujo que por su formulación teórica.

5.

Autor: MAKOWSKI, Z.S.  
Título: **Space structures of today an tomorrow. A review of their future use**  
Editorial: Proceedings in the third ICSS. Science Publishers. London, 1984

6.

Autor: MARGARIT, J. ; BUXADE, C.  
Título: **Las mallas espaciales en arquitectura.**  
Editorial: Gustavo Gili. Barcelona. 1.972.  
Notas: Estudia exhaustivamente el comportamiento de los módulos. Métodos de cálculo por discretización o por asimilación del entramado a estructuras sólidas continuas (placas). Se incluyen programas de cálculo electrónico y ejemplos.

7.

Autor: MARGARIT, J. ; BUXADE, C.  
Título: **Cálculo matricial de estructuras de barras.**  
Editorial: Colegio Oficial de Arquitectos de Cataluña y Baleares 1.970.  
Notas: Libro útil en especial los temas de programación en ordenador.

8.

Autor: MARGARIT, J. ; BUXADE, C.  
Título: **Cálculo de esfuerzos de estructuras de barras mediante ordenadores y métodos manuales.**  
Editorial: Monografía nº 9.1. E.T.S.A. de Barcelona.  
Notas: Realiza un estudio de los distintos métodos simplificados de cálculo, el método de Cross y el matricial.

9.

Autor: MUÑIZ GÓMEZ, S. ; FREIRE TELLADO, M.J. ; ESTÉVEZ CIMADEVILA, F.J.  
Título: **Mallas espaciales.**  
Editorial: Departamento de Tecnología de la Construcción. Universidad La Coruña. 1.992  
Notas: Libro donde se contempla desde los aspectos teóricos hasta los prácticos y detalles constructivos de mallas espaciales, aunque de forma bastante resumida.

10.

Autor: NOOSHIN, H.  
Título: **Formex configuration processing in structural engineering**  
Editorial: Elsevier Applied Science Publishers.  
Notas: Se expone el programa Formex de cara a la ingeniería.

11.

Autor: NOOSHIN, H.  
Título: **Formex formulation of double layer grids**  
Editorial: Z.S. Makowski, Applied Science Publishers, 1981  
Notas: Formex aplicado a mallas de doble capa.

12.

Autor: PÉREZ VALCÁRCEL, J.B. ; ESTÉVEZ, J.  
Título: **Mallas espaciales. Matricial: Manual de usuario.**  
Editorial: E.T.S.A. La Coruña. 1.985.  
Notas: Monografía en la que se exponen los principios básicos del cálculo matricial de mallas espaciales, se aporta un programa y se da el manual de usuario del mismo.



13.

Autor: RUSTOM MADI, U.  
Título: **An investigation into the design parameters of double layer space frame grids**  
Editorial: Space Structures 2, 1986

14.

Autor: TSUBOI, Y.  
Título: **Analysis, Design and Construction of Space Frames.**  
Editorial: I.A.S.S. Madrid. 1.979.  
Notas: Libro interesante, pero estudia un gran número de temas sin una hilación entre ellos que los haga más asimilables.

---

## 6.2.2 CÁLCULO MATRICIAL Y ELEMENTOS FINITOS.

---

1.

Autor: ALARCÓN ÁLVAREZ, E. ; ÁLVAREZ CABAL, R. ; GÓMEZ LERA, M. S.  
Título: **Cálculo matricial de estructuras**  
Editorial: Reverté S. A.. Colección de Matemática Aplicada e Informática. Barcelona, 1983  
Notas: Texto de cálculo matricial de gran interés. Trata el cálculo estático, dinámico y los problemas de estabilidad, acompañados de ejemplos y de un programa de aplicación en Basic. Recomendable.

2.

Autor: ARGYRIS, J.  
Título: **Recent Advances in Matrix Methods of Structural Analysis.**  
Editorial: Pergamon Press. 1.964  
Notas: Es un libro excelente para el estudio avanzado de los métodos matriciales.

3.

Autor: AZIZ, A.K.  
Título: **The mathematical foundations of the finite element method with applications to partial differential equations**  
Editorial: Academic. New York, 1972

4.

Autor: BATHE, K.J.  
Título: **Finite element procedures in engineering analysis**  
Editorial: Prentice - Hall. Englewood Cliffs. New Jersey, 1981  
Notas: Incluye ejemplos de como implementar rutinas de cálculo en fortran.

5.

Autor: BECKER E.B. ; CAREY, G.F. ; ODEN, J.T.  
Título: **Finite element: an introduction, vol. 1, 2**  
Editorial: Prentice - Hall. Englewood Cliffs. New Jersey, 1981

6.

Autor: BRAY, K.H.M. ; CROXTON, P. C. L. ; MARTIN, L. H.  
Título: **Análisis matricial de estructuras**  
Editorial: Paraninfo S.A. Madrid, 1979  
Notas: Excelente libro de iniciación al cálculo matricial en el que, pese a su nivel elemental, se llega a temas avanzados.

7.

Autor: BUNCH; ROSE.  
Título: **Sparse matrix computations.**  
Editorial: American Elsevier. New York. 1.969.  
Notas: Libro excelente aunque limitado a una técnica muy concreta.

8.

Autor: COOK, R.D.  
Título: **Concepts and applications of finite element analysis, 2nd ed.**  
Editorial: Wiley. New York, 1981

9.

Autor: DESAI, C.S. ; ABEL, J.F.  
Título: **Introduction to the finite element method. A numerical method for engineering analysis**  
Editorial: Van Nostrand Reinhold. New York, 1972

10.

Autor: FORNONS GARCIA, JOSE MARÍA  
Título: **El método de los elementos finitos en la ingeniería de estructuras**  
Editorial: Marcombo de Boixareu Editores. U. P. Barcelona, 1982  
Notas: Formulación directa, variacional y residual del método de los elementos finitos. Análisis dinámico y no lineal. Buen libro.

11.

Autor: GERSTLE, K.H.  
Título: **Basic Structural Analysis.**  
Editorial: Prentice Hall. 1.974  
Notas: Introducción al análisis estructural planteado por métodos matriciales. Acompaña programas para el cálculo de estructuras articuladas y pórticos.

## 12.

Autor: GHALI, A. ; NEVILLE, A. M.  
Título: **Análisis Estructural. Un enfoque unificado, clásico y por matrices**  
Editorial: Diana. México, 1983  
Notas: Son muy interesantes los capítulos dedicados al cálculo de plástico de pórticos y placas y los de diferencias finitas y elementos finitos, claramente explicados, así como el estudio en profundidad de la variación de la rigidez por influencia del axil.

## 13.

Autor: HAYRETTIN KARDESTURNCER  
Título: **Introducción al análisis estructural con matrices**  
Editorial: Mc Graw - Hill. México, 1975

## 14.

Autor: HINTON, E. ; OWEN, D.R.J.  
Título: **An introduction to finite element computations**  
Editorial: Pineridge Press. Swansea, United Kindgom, 1979

## 15.

Autor: HUEBNER, K.H. ; THORNTON, E.A.  
Título: **The finite element method for engineers, 2nd ed.**  
Editorial: Wily - Interscience. New York, 1972

## 16.

Autor: KIKUCHI, N..  
Título: **Finite element method in mechanics**  
Editorial: Cambrigde University Press. Cambrigde, 1986  
Notas: Muy desarrollado, con ejemplos concretos y aplicaciones informáticas.

## 17.

Autor: LIVESLEY, R.K.  
Título: **Cálculo matricial de estructuras.**  
Editorial: Blume. Barcelona. 1.970.  
Notas: Libro clásico de cálculo matricial, completo y bien expuesto. Recomendable.

## 18.

Autor: LIVESLEY, R. K.  
Título: **Matrix methods of structural analysis**  
Editorial: Pergamon Press. 1964

## 19.

Autor: LÓPEZ DE CEBALLOS, G. ; LECEA, C.  
Título: **Cálculo matricial de estructuras y placas.**  
Editorial: Rugarte, S.L. Madrid. 1.978  
Notas: Libro claro y bien estructurado sobre el cálculo matricial. También incluye el estudio de placas por métodos elásticos y de líneas de rotura.

## 20.

Autor: LUTHE, R.  
Título: **Análisis estructural.**  
Editorial: Representaciones y servicios de ingeniería S.A. México. 1.971  
Notas: Después de una presentación de los métodos energéticos, se analizan con detalle las estructuras articuladas, comenzando por el empleo de los métodos clásicos para terminar con el planteamiento matricial del método de las fuerzas. Estudia además las estructuras reticuladas.

## 21.

Autor: MAJID, K.I.  
Título: **Matrix Methods of Analysis and Design by Computers. Non-linear Structures.**  
Editorial: Butterwoods. Londres. 1.972.  
Notas: Planteamiento bastante completo en términos matriciales. Interesante para problemas de estabilidad y plasticidad.

## 22.

Autor: MARTÍN GUTIERREZ, E. ; PÉREZ VALCARCEL, J. B. ; ESTÉVEZ, F. J.  
Título: **Análisis matricial de sistemas estructurales**  
Editorial: Galicia Editorial S.A. La Coruña, 1993  
Notas: Desarrolla con mucho detalle los métodos matriciales.

## 23.

Autor: MARTIN, H.C.  
Título: **Introduction to matrix methods of structural analysis.**  
Editorial: Mc Graw-Hill. 1.966  
Notas: Trata el método matricial de la Rigidez y Flexibilidad. De destacar el capítulo dedicado a elementos curvos.

## 24.

Autor: Mc GUIRE ; GALLAGHER, B.H.  
Título: **Matrix structural analysis.**  
Editorial: John Wiley and Sons. 1.979  
Notas: Expone los conceptos básicos para la aplicación del Análisis Estructural al ordenador. Destaca el capítulo dedicado a temas especiales (condensación y subestructuración) y el correspondiente a la resolución de sistemas de ecuaciones.

## 25.

Autor: MORÁN CABRÉ, F.  
Título: **Análisis matricial de estructuras en ordenadores personales compatibles.**  
Editorial: Rueda. Madrid. 1.990  
Notas: Plantea el estudio de estructuras de nudos articulados y rígidos en dos y tres dimensiones. Acompaña un diskette con los programas elaborados.

## 26.

Autor: NOBORU KIKUCHI  
Título: **Finite element methods in mechanics**  
Editorial: Cambridge University Press. New York, 1986

27.

Autor: NORRIE, D.H. ; DE VRIES, G.  
Título: **An introduction to the finite element method, 2nd ed.**  
Editorial: Academic. New York, 1978

28.

Autor: ODEN, J.T.  
Título: **Finite elements of nonlinear continua**  
Editorial: Mc Graw - Hill. New York, 1972

29.

Autor: ODEN, J.T. ; REDDY, J.N.  
Título: **An introduction to the mathematical theory of finite elements**  
Editorial: Wiley. New York, 1976

30.

Autor: OWEN, D.R.J. ; HINTON, E.  
Título: **A simple guide to finite elements**  
Editorial: Pineridge Press, Swansea. United Kingdom, 1980

31.

Autor: PANDIT, G. S. ; GUPTA, S. P.  
Título: **Structural analysis. A matrix approach**  
Editorial: Mc Graw - Hill Publising Company Limited. New Delhi, 1981

32.

Autor: PRZEMIENIECKI, J.S.  
Título: **Theory of Matrix Structural Analysis.**  
Editorial: Mc Graw-Hill. Londres, 1.968  
Notas: Es un libro interesante, presenta la generación de los métodos matriciales para estructuras de barras, generalizándolos al método de los elementos finitos.

33.

Autor: RUBINSTEIN, M.F.  
Título: **Introduction to matrix methods of structural analysis.**  
Editorial: Mc Graw-Hill. 1.966  
Notas: Métodos matriciales de análisis de estructuras por ordenador. De destacar los capítulos dedicado al análisis de sistemas por subestructuras y aplicación al cálculo de placas y láminas.

34.

Autor: SAEZ-BENITO ESPADA, J. M.  
Título: **Cálculo matricial de estructuras formadas por piezas prismáticas**  
Editorial: Fondo editorial de Ingeniería Naval. Madrid, 1975  
Notas: Exposición en general clara y didáctica con un estudio muy completo de estructuras formadas por piezas prismáticas. Recomendable.

35.

Autor: STEVEN C. CHAPRA; RAYMOND P. CANALE  
Título: **Métodos Numéricos para Ingenieros**  
Editorial: Mc Graw-Hill. 1.999  
Notas: Exposición de métodos numéricos para diferenciación e integración numérica y optimización, con aplicación a los cálculos por ordenador, y notas sobre los programas MathCad y MathLab.

36.

Autor: TONG, P. ; ROSSETTOS, J.N.  
Título: **Finite element method - basic techniques and implementation**  
Editorial: MIT Press. Cambridge, Massachusetts, 1977

37.

Autor: VARGA.  
Título: **Matrix Iterative Analysis.**  
Editorial: Prentice Hall. Nueva York. 1.962.  
Notas: Libro muy completo sobre métodos de resolución de ecuaciones. Es un libro de matemáticas, pero útil para el estudio del cálculo matricial y de elementos finitos.

38.

Autor: VICHNEVETSKY, R.  
Título: **Computer methods for partial differential equations, vol. 1**  
Editorial: Prentice - Hall. Englewood Cliffs, New Jersey, 1981

39.

Autor: ZIENKIEWICZ O. C.; TAYLOR, R. L.  
Título: **The Finite Element Method. 4th ed.**  
Editorial: Mc Graw - Hill. London, 1989

40.

Autor: ZIENKIEWICZ O. C. ; MORGAN, K.  
Título: **Finite element method and approximation**  
Editorial: Wiley - Interscience. New York, 1982

---

# 7. Anexos

---



---

## 7.1 EL PROGRAMA ARTIC.

---



---

### 7.1.1 GENERALIDADES.

---

#### GENERALIDADES

Todos los procesos de cálculo anteriormente comentados han sido posibles gracias a su implementación informática, ya que de otra forma no serían posibles los ingentes cálculos que requieren estos procesos de cálculo.

Por ello se ha desarrollado un programa denominado ARTIC, que en principio engloba a estructuras de nudos articulados, tales como mallas desplegadas, o las mallas espaciales tradicionales.

Este programa está pensado para su instalación en ordenadores personales, tipo PC o compatibles, bajo el entorno operativo Windows. Es aconsejable elegir directorio propio para su instalación; en principio se sugiere que sea: C:\Archivos de programa\Artic

La **entrada de datos** se realizará a través del ratón y teclado, ya sea para la introducción de datos o selección de opciones.

La **salida de datos** se realizará por la pantalla y también por la impresora. Nos suministrará los datos numéricos de la estructura y los resultados de los cálculos efectuados. También se obtienen resultados gráficos como dibujos en axonometría y planta con la numeración de nudos y barras y perspectivas de la malla en cuestión, que pueden ser exportados a programas de dibujo a través del formato de intercambio gráfico DXF.

La **instalación** se efectúa como es habitual en los programas Windows haciendo doble click sobre el programa de instalación, el cual creará los directorios y accesos directos precisos para su correcto funcionamiento.

La **estructura general** de funcionamiento del programa es a través de un MENÚ principal que nos permitirá acceder a cualquiera de los otros módulos (siempre que hallamos generado los datos precisos), recuperando el control seguidamente él mismo de nuevo. Aparte nos permitirá realizar ciertas operaciones con los ficheros de los trabajos que realicemos, como copiarlos, borrarlos, etc. y también la forma de salir del programa es a través de él.

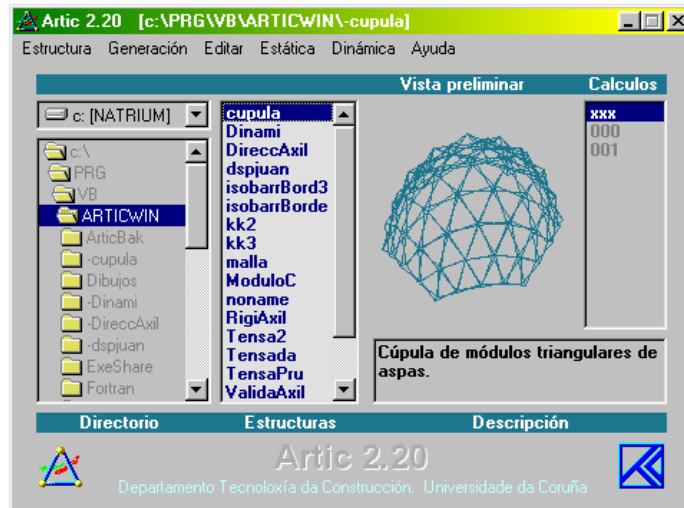


## 7.1.2 USO DEL PROGRAMA.

Hacemos arrancar el programa dando la orden de arranque **Artic**, o haciendo doble click sobre el icono correspondiente:



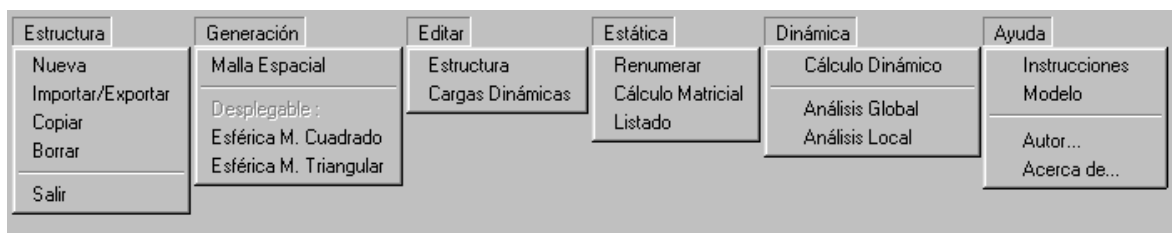
Nos aparece entonces la pantalla inicial con el menú principal desde el que podemos acceder a las distintas opciones del programa:



En la parte central izquierda tenemos el disco y directorio de trabajo, y a su derecha un listado de las estructuras existentes en el directorio de trabajo. De la estructura que se halla seleccionada, nos aparece a su derecha un dibujo de la misma, con una breve descripción y un listado de los cálculos que se han efectuado sobre la misma.

Los datos de cada estructura los almacenamos en una carpeta del mismo nombre precedido por un guión, por lo que nos aparecerá la misma en la lista de carpetas. La idea es que todos los datos, resultados y demás ficheros relacionados con una estructura puedan almacenarse en un mismo lugar y estén fácilmente localizables.

Observamos que el menú principal agrupa las opciones disponibles en seis apartados:



El uso de las distintas opciones es sucintamente el siguiente:

### **Estructura:**

- |                    |  |
|--------------------|--|
| Nueva.             | Crea el directorio para los nuevos datos a generar.            |
| Importar/exportar. | Importa o exporta a otros formatos                             |
| Borrar.            | Borra completamente una estructura.                            |
| Copiar.            | Copia el directorio completo de la estructura con otro nombre. |
| Salir.             | Finaliza el programa.  |

**Generación:**

Malla espacial. Genera varias tipologías de mallas espaciales.  
Desplegable Esférica Módulo Cuadrado.  
Desplegable Esférica Módulo Triangular.

**Editar:**

Estructura Para modificar coordenadas, tipos de barras, cargas... etc.  
Cargas dinámicas. Para modificar las cargas dinámicas establecidas por defecto.

**Estática:**

Renumerar. Renumera los nudos de la estructura.  
Cálculo Matricial. Realiza un cálculo matricial estático de la estructura.  
Listados. Lista datos y resultados a un fichero de texto.

**Dinámica:**

Cálculo Dinámico. Realiza el cálculo dinámico de la estructura.  
Análisis Global. Vista del despliegue y esfuerzos máximos.  
Análisis Local. Estudio de una barra o nudo concreto.

**Ayuda:**

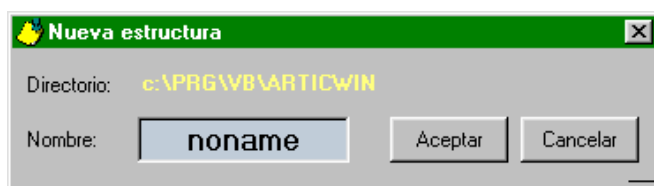
Instrucciones. Breve listado de ayuda  
Modelo. Nociones sobre el modelo empleado en el cálculo dinámico.  
Autor. Datos del autor  
Acerca de... Datos de la versión del programa.

Pasaremos ahora a comentar más detenidamente una por una las distintas opciones.

## ESTRUCTURA

**Nueva.**

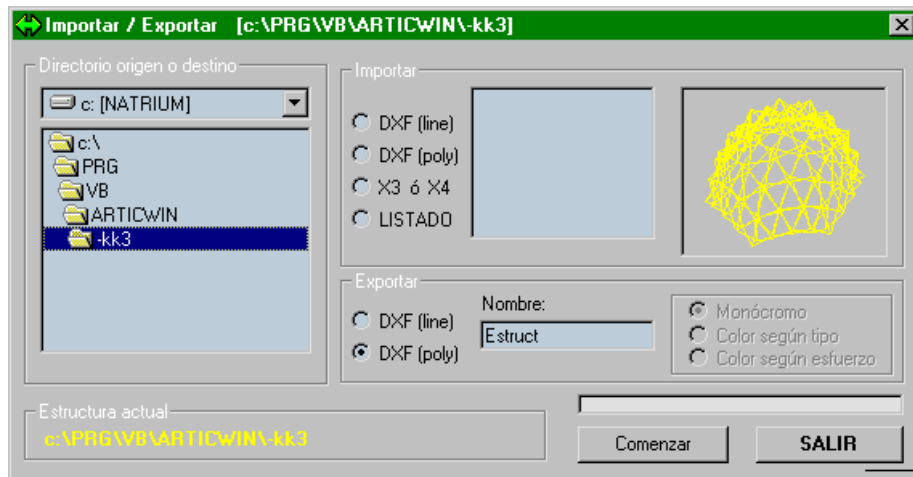
Crea el directorio para los nuevos datos a generar. Esta opción sirve únicamente para dar un nombre a la estructura de trabajo.



Internamente lo que hace es crear un directorio dónde se almacenan todos los datos y cálculos que efectuemos en esta estructura. El primer carácter del directorio será siempre un guión ( - ), para identificar de este modo los directorios que son estructuras de trabajo. Así si por ejemplo le damos a nuestra estructura el nombre MALLA23, se creará un directorio con el nombre -MALLA23.

**Importar exportar.**

Importa o exporta a otros formatos.



En principio puede importar o exportar al formato de intercambio gráfico **DXF**, con líneas o polilíneas. Si se trata de una malla espacial normal basta usar la opción de líneas, si es una desplegable, en la cual nos interesa conservar la posición del nudo central, es indispensable usar la opción de polilíneas, entonces para cada barra se usa una polilínea compuesta de dos líneas, una para cada tramo de barra. Al tratarse este de un formato de intercambio gráfico, solamente podemos exportar o importar la geometría, y ni siquiera especificar que barras se hallan enlazadas (nudo central de las aspas), pero el mismo programa al importar detecta los nudos centrales que están suficientemente próximos y los conecta entre sí.

**X3** y **X4** son dos programas de cálculo matricial desarrollados por el departamento de estructuras de la Universidad de A Coruña. Estos son dos programas de cálculo de estructuras planas, el primero de nudos rígidos, y el segundo de nudos rígidos y articulados. Lógicamente no tiene sentido importar pórticos, pero si cerchas. Se importa geometría, características mecánicas y cargas, pero si hay nudos rígidos, estos pasarán a ser articulados.

**LISTADO** hace referencia a los listados generados por este mismo programa en la opción de Estática/Listados, que se verá más adelante. El listado que se genera contiene una definición total de la estructura (de momento en su vertiente estática) y es un fichero de texto con un contenido similar al siguiente:

Programa: Artic 2.13

1.999 © M. Muñoz Vidal

Proyecto: c:\PRG\VB\ARTICWIN\kkno2

Fecha (d-m-a): 30-08-2001

Hora Listado : 22:12:52

DATOS

Número de puntos.....= 8  
 Número de barras.....= 8  
 N° de tipos de barras.....= 1 (usados: 1 )  
 Peso de la estructura (kg)= 11,31371

CARACTERISTICAS DE LAS BARRAS

Barr.	Inic.	Fin	Tpo	Barr.	Inic.	Fin	Tpo	Barr.	Inic.	Fin	Tpo
1	1	4	1	2	5	8	1	3	1	6	1
4	3	8	1	5	2	3	1	6	6	7	1
7	2	5	1	8	4	7	1				

COORDENADAS DE LOS NODOS (en m.)

Nudo	Abcis.X	Orden.Y	Altur.Z	Nudo	Abcis.X	Orden.Y	Altur.Z
1	0,000	0,000	1,000	2	0,000	0,000	0,000
3	1,000	0,000	1,000	4	1,000	0,000	0,000
5	0,000	1,000	1,000	6	0,000	1,000	0,000
7	1,000	1,000	1,000	8	1,000	1,000	0,000

TIPOS DE BARRAS

N°	Nombre	Area (cm2)	MDI (cm4)	M.Young (kg/cm2)	T.adm (kg/cm2)	PE (kg/m3)
1	Tubo.....40.2	1,00	1,00	1000000	1000	10000

COACCIONES EN LOS NUDOS (en m.)

Nudo	Coacc.X	Coacc.Y	Coacc.Z
1	0,0000	0,0000	LIBRE
2	0,0000	0,0000	0,0000
3	LIBRE	0,0000	LIBRE
4	LIBRE	0,0000	LIBRE
5	0,0000	LIBRE	LIBRE
6	0,0000	LIBRE	LIBRE

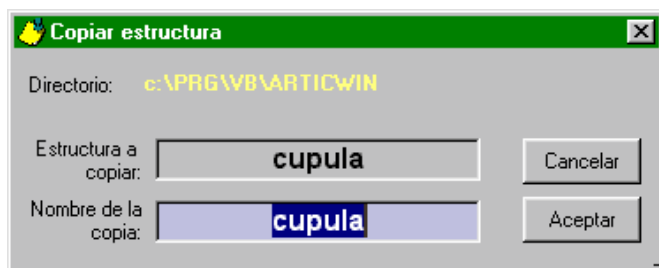
CARGAS EN LOS NUDOS (en tn.)

Nudo	Carga X	Carga Y	Carga Z	Nudo	Carga X	Carga Y	Carga Z
7	0,100	0,100	0,000				

Un listado tipo el anterior, podemos editarlo con un editor de texto puro para cambiar algún dato concreto, coordenadas de los nudos, características de las barras, cargas, etc. Una vez efectuado los cambios iríamos a esta opción de importar listado y los cambios quedarían recogidos en la estructura de trabajo.

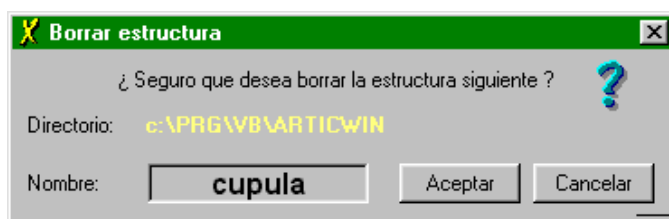
**Copiar.**

Copia el directorio completo de la estructura con otro nombre. Nos permite sacar una copia de un trabajo determinado con otro nombre y así podremos trabajar con la copia sin temor a estropear el trabajo inicial.



**Borrar.**

Con esta opción podemos eliminar del directorio cualquier trabajo que no deseemos conservar. El programa borrará la estructura que tengamos seleccionada en ese momento, aunque antes nos pedirá confirmación:



## Salir.

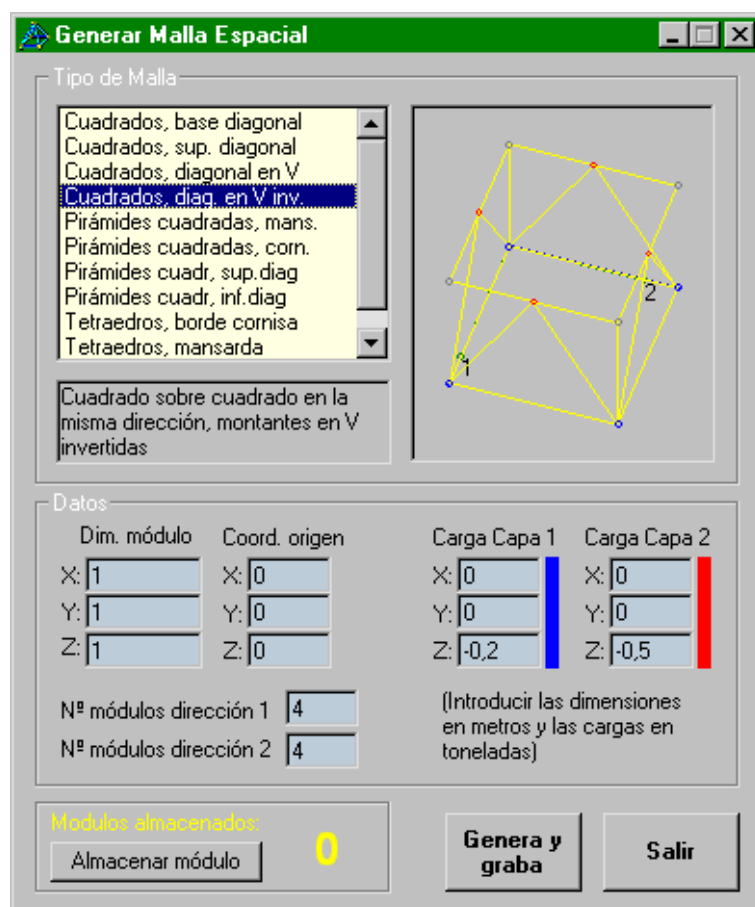
Finaliza el programa.

## GENERACIÓN

### Malla espacial.

Genera varias tipologías de mallas espaciales con barras conectadas únicamente en sus nudos extremos. De este modo podemos generar una malla plana de forma rectangular o romboidal (si se halla definida) con el número de módulos y tamaño que queramos, usando alguno de los tipos de módulos que nos proporciona el programa.

Nada más arrancar esta opción nos aparece una ventana similar a la siguiente:



La forma de actuar será la siguiente: primeramente elegiremos el tipo que queramos de la lista de tipos.

Efectuada la selección introduciremos las dimensiones X, Y, Z, del módulo (largo, ancho y alto) que deseamos que tenga. Luego las coordenadas del nodo inicial de la malla para situar ésta en algún lugar del espacio. Si no tenemos ninguna preferencia podemos dar las coordenadas 0,0,0.

Lo siguiente será introducir las cargas que hay en cada uno de los nudos de la capa 1 y de la capa 2. Generalmente la capa 1 se refiere a la inferior y la 2 a la superior, aunque puede no ser así dependiendo de la definición del módulo. Para mayor claridad los nudos de cada capa se dibujan de un color diferente, y los que no tienen asignado color es en los que no se generarán cargas.

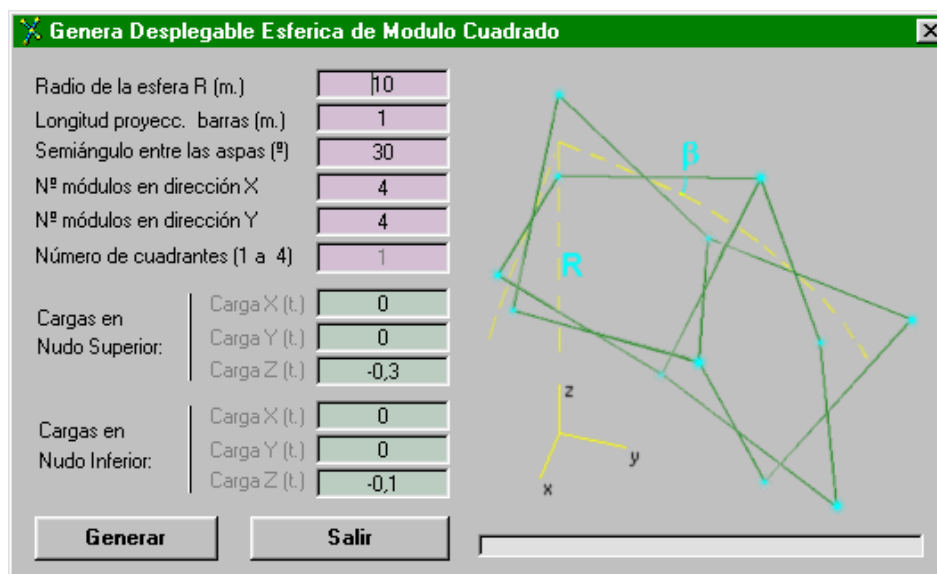
Lo siguiente será definir el tamaño de la malla, para ello tendremos que dar el número de módulos que queremos que tenga en cada dirección. Es preciso apuntar aquí que en algunos casos el tamaño del módulo a repetir no es el mismo del que anteriormente se pedían las dimensiones; esto se especifica en las fichas descriptivas de los módulos que figuran al final de este anexo. Por ejemplo tenemos el caso del tipo tetraédrico, que mientras se solicitan las dimensiones del típico tetraedro, al introducir el número de módulos que queremos en la primera dirección, es preciso hacer constar que en esta dirección el módulo consta de dos tetraedros, no de uno como sería usual; pero esto será el caso excepcional, por lo general no habrá estas distinciones.

Una vez hemos dado todos estos datos podemos dar al botón de generar y grabar y se generaría la malla elegida.

Pero aquí tenemos otra opción: Almacena módulo. Cada vez que pulsemos este botón, el programa guarda la definición de la malla actual y la añade a una lista de mallas definidas. Esta opción es válida para el caso que tengamos que formar una malla por medio de dos o más módulos, como sería por ejemplo una malla para cubierta a dos aguas, o una malla con dos o más niveles, o dos mallas independientes que queremos calcular juntas porque interactúan de alguna forma. Una vez introducidos todos los módulos deseados pulsáramos de nuevo el botón de generar y grabar y se generarían de una vez todas las mallas definidas.

### Desplegable Esférica Módulo Cuadrado.

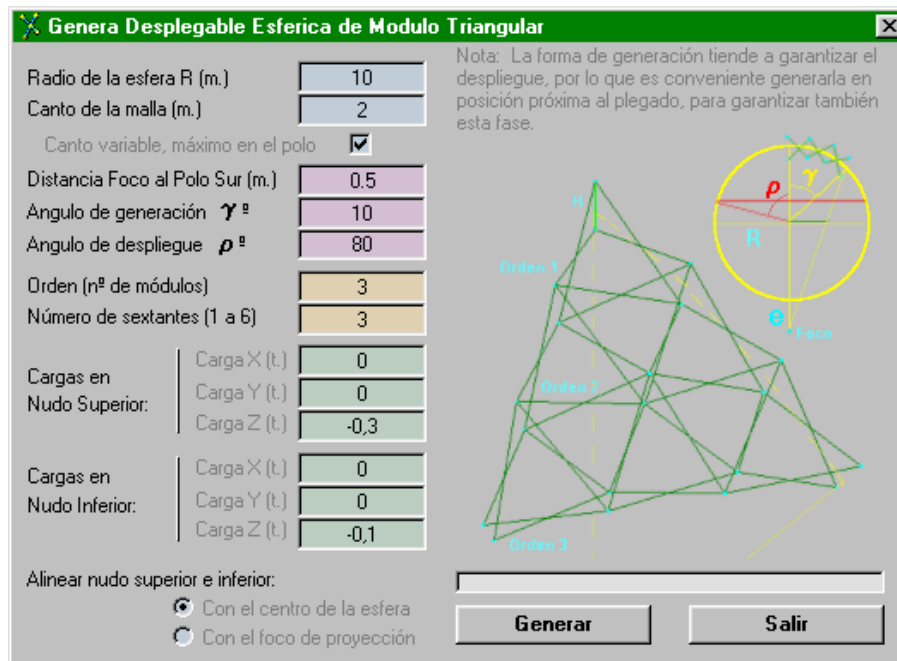
Genera una malla desplegable de módulo cuadrado, según se ha detallado en el apartado de metodología.



En este caso las capas inferior y superior siempre hacen referencia a los nudos interiores y exteriores a la superficie esférica respectivamente.

### Desplegable Esférica Módulo Triangular.

Genera una malla desplegable de módulo triangular, según se ha detallado en el apartado de metodología.



La opción de canto variable está presente por defecto y es la que se explicó profusamente en el apartado de metodología. En caso de desactivar esta opción la malla se genera con canto constante, se ha comprobado que no despliega bien, pero se mantiene esta opción a nivel experimental. Asimismo por defecto los nudos estarán alineados con el centro de la esfera, pero se mantiene la opción de alinearlos con el foco de proyección también con la idea de experimentar con estos parámetros.

En este caso como en el anterior, las capas inferior y superior siempre hacen referencia a los nudos interiores y exteriores a la superficie esférica respectivamente.

## EDITAR

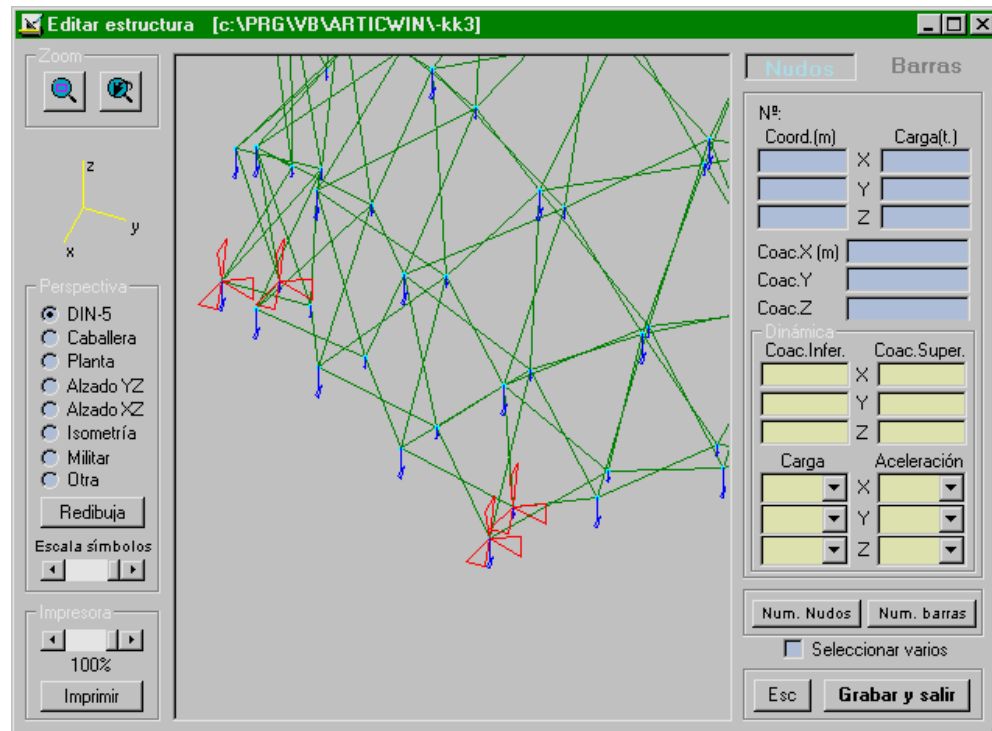
### Estructura.

Para modificar coordenadas, tipos de barras, cargas... etc. Este apartado nos permite unas correcciones generales de la estructura. Es imprescindible pasar por este apartado si una vez realizado una generación automática de la malla, para al menos definir unas coacciones en los nudos antes de pasar a realizar el cálculo. La pantalla que nos encontraremos será del tipo siguiente, en la cual en la parte central tenemos el dibujo de nuestro trabajo en cuestión.

En ella en la parte izquierda están las opciones de visualización. Arriba de todo están los botones de **zoom** con ventana y zoom previo. Debajo de estos un dibujo de la situación de los ejes coordenados según la **perspectiva** actualmente seleccionada, que podemos variar eligiendo otra justo debajo de estos. Debajo de las perspectivas está la opción de **redibujar**, que es útil por ejemplo cuando variamos la coordenada de un nudos y queremos que el dibujo muestre esta actualización.

Un poco más abajo está la opción de **escala símbolos**, que hace referencia al tamaño con que se dibujarán los símbolos que se usan para representar las coacciones (en rojo) y las cargas (en azul) estáticas.

Y abajo de todo está la opción de imprimir el dibujo que estamos viendo en la impresora por defecto. Si no queremos que el dibujo ocupe toda la página (100%) iremos disminuyendo este porcentaje.



Por defecto entramos en la opción de editar nudos como vemos en la parte superior derecha. En este lado derecho de la pantalla están las opciones de modificación de los nudos: coordenadas, cargas coacciones y el apartado específico de dinámica, que solo hará falta rellenar si vamos a efectuar un cálculo dinámico. Para que se activen las opciones es necesario tener seleccionado un nudo o varios previamente.

Para seleccionar un nudo basta hacer click con el ratón sobre el mismo y en la parte derecha aparecerán los datos del mismo. Si en la vista actual aparecen varios nudos superpuestos o muy juntos, bastará con hacer click repetidamente sobre el mismo punto para que vayan apareciendo los datos de los sucesivos nudos que hay superpuestos, cuando pasemos por todos, en el siguiente click se borrarán todos los datos y se repetirá el ciclo.

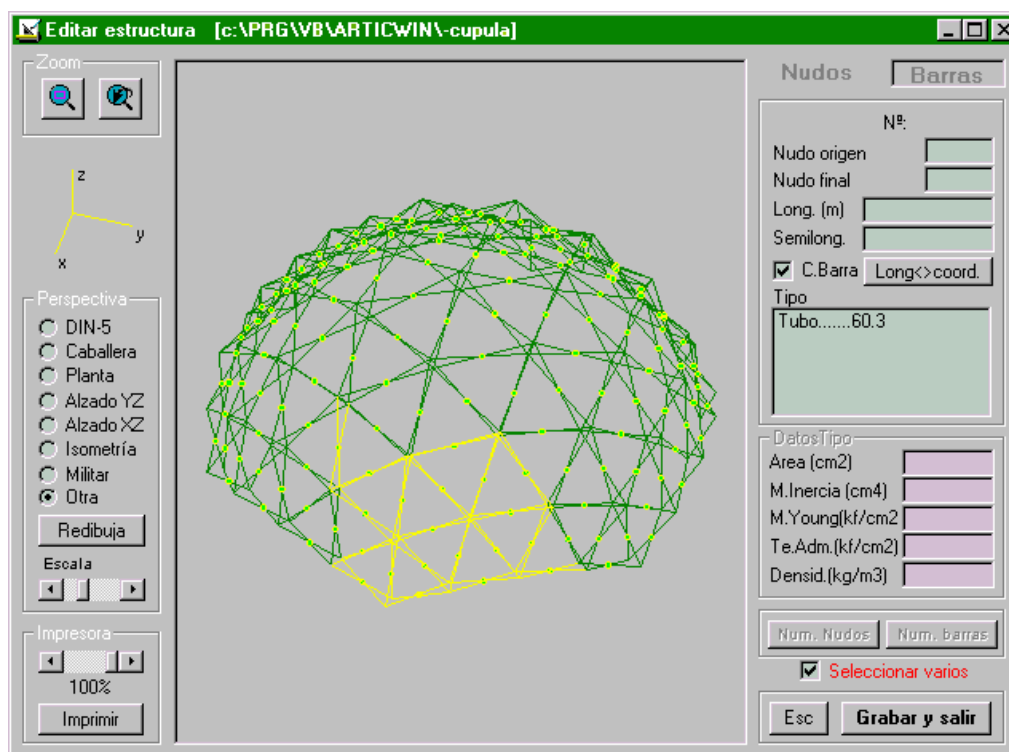
Si queremos aplicar a varios nudos las mismas coordenadas, cargas o coacciones, debemos activar previamente la casilla inferior que pone **Seleccionar varios**, y a continuación iremos haciendo click sobre cada nudo que queramos añadir a nuestro conjunto, y también podemos elegir varios a la vez trazando una ventana sobre el dibujo (pulsar el botón del ratón y dejarlo pulsado mientras lo arrastramos). Cuando tengamos activada esta casilla, las modificaciones que efectuemos se aplicarán a todos los nudos seleccionados en ese momento. Debemos señalar aquí que mientras está activada esta casilla no podemos ir a editar barras, por lo que habrá que desactivarla previamente.

En el apartado de dinámica podemos definir unas coacciones a mayores, una inferior y otra superior, que está pensado para limitar el movimiento de los nudos que no tienen coacciones estáticas. Para cada dirección del espacio se comportan como dos planos elásticos entre los cuales se puede desplazar el nudo al llegar a uno de estos, el nudo rebotaría (dependiendo de la constante elástica definida). Combinado las tres direcciones del espacio se puede restringir el movimiento de un nudo dentro de un prisma recto de base rectangular.

Las cargas y aceleraciones dinámicas se introducen a mayores de las cargas definidas para el nudo pudiendo elegir entre tres funciones rampa, tres funciones seno y tres tablas definidas por el usuario con la opción de Editar / Cargas dinámicas.



Si vamos a la opción de barras, el aspecto del módulo pasa a ser el siguiente:



Solamente varía la parte derecha. Ahora al pulsar sobre una barra nos aparecerán los datos de esta y tendremos la opción de modificarlos. Típicamente son: nudo origen, nudo extremo, longitud inicial (no tiene por que coincidir con la distancia entre nudos origen y extremo si la barra tiene esfuerzos previos), semilongitud y tipo de barra.

La semilongitud hace referencia a la posición del nudo intermedio medida desde el nudo de menor numeración al de mayor numeración. Para ver la numeración de nudos o barras abajo aparecen los botones '**Num. Nudos**' y '**Num. Barras**'.

Si tenemos activada la casilla '**C.Barra**', cuando variemos la semilongitud, automáticamente recalcula las coordenadas del nudo intermedio según ese valor. Si en una malla desplegable queremos introducir tensiones previas, deberemos quitar esta marca para poder variar la semilongitud sin cambiar la posición del nudo.

El botón '**Long<>coord.**' hace que, para las barras seleccionadas, se actualicen sus longitudes según las coordenadas actuales de los nudos, y en semilongitud pondrá la mitad de las anteriores. Si hemos hecho grandes cambios de las coordenadas y no queremos que aparezcan tensiones iniciales debemos pulsar este botón. Hay que tener cuidado si estamos trabajando con mallas desplegadas, pues en muchas el nudo intermedio no está justamente en el centro de la barra, y obtendríamos por tanto una estructura que no funcionaría.

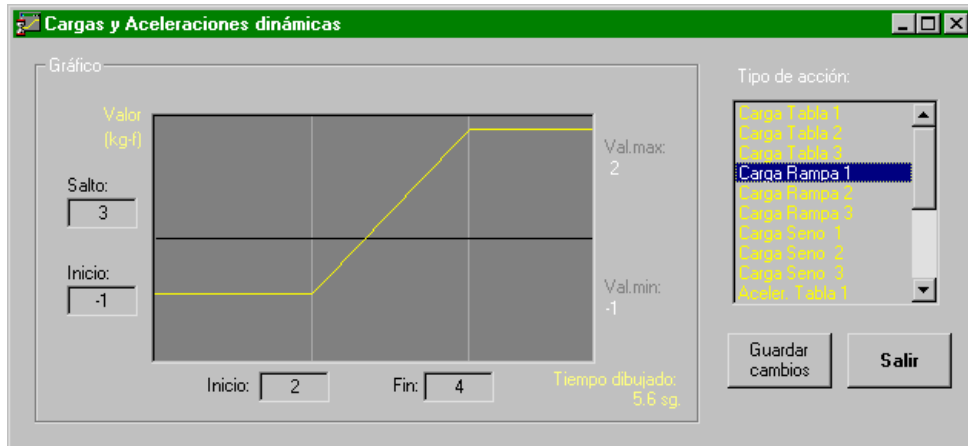
Debajo del tipo de barra aparecerán los datos del tipo de barra que se halla actualmente seleccionada, y también podremos modificar aquí los datos del tipo de barra. Para modificar un tipo de barra debemos por tanto pinchar previamente sobre una barra de dicho tipo. Debemos tener en cuenta obviamente que el cambio de un tipo de barra afectará a todas las barras que compartan dicho tipo, no solo a la seleccionada.

Un opción que no aparece claramente es que si picamos con el **botó derecho del ratón** sobre un nudo y arrastramos hasta otro nudo se introducirá una barra entre ambos nudos, y si ya había una se borrará. Si cuando soltamos no hay un nudo cerca se generará uno.

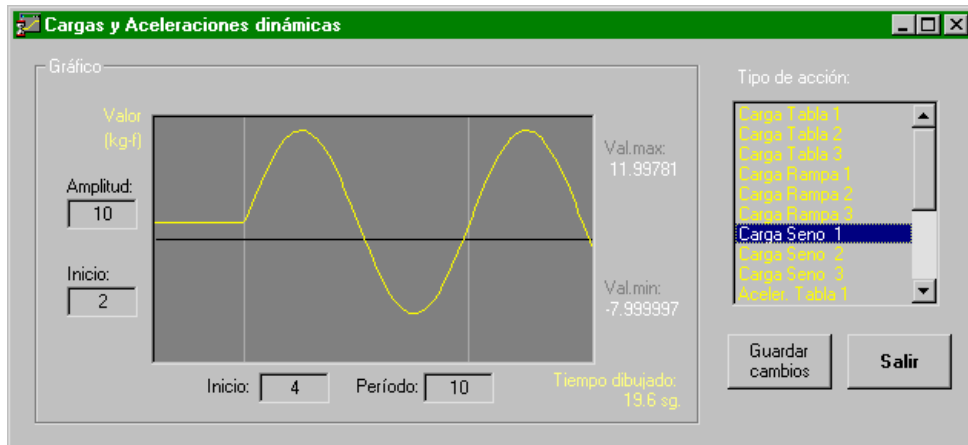
### Cargas dinámicas.

Para crear las cargas o aceleraciones dinámicas entraremos en esta opción. Las cargas o aceleraciones pueden ser de tres tipos:

**Rampa:** valor constante, variación lineal y valor constante. Se definirán el valor inicial y el salto que da la función entre los tiempos inicial y final.



**Seno:** Valor inicial constante y después función seno indefinida. Los valores a dar ahora serán el valor constante inicial, y la amplitud de la función seno; y se especificará el tiempo de arranque de la función seno y su período.



**Tabla:** Se usan los datos numéricos de tablas introducidas por el usuario.



Las tablas 1, 2 y 3 de cargas se llamarán respectivamente CARD1.DAT, CARD2.DAT y CARD3.DAT. Las de aceleraciones AD1.DAT, AD2.DAT y AD3.DAT y son ficheros de texto que estarán en el directorio de la estructura. Estos ficheros de texto tendrán datos equiespaciados en el tiempo de las cargas a aplicar. Por ejemplo:

```
20
Impacto tipo uno
0.1, 0.5, 0.7, 0.6, 0.4, 0.2, 0.1, -0.2, -0.4, -0.5
-0.4, -1, -0.5, 0.2, 0.6, 0.4, 0.5, 0.7, 0.8, 0.1
```

La primera línea nos dice el nº de datos numéricos de que consta la tabla, la segunda es una breve descripción de la tabla y a continuación se da la serie de valores.

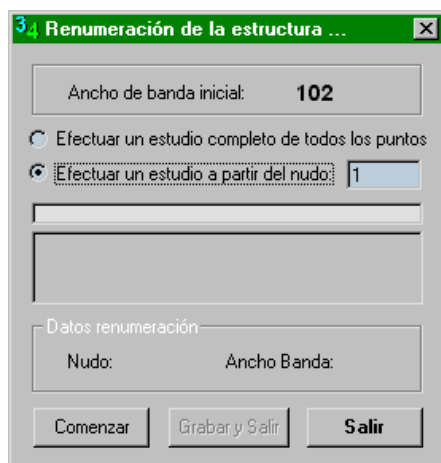
En este caso el valor 'salto' será el multiplicador para todos los valores de la tabla. Si ponemos aquí valor 1 la carga tomará los mismos valores de la tabla más el valor que pusimos en 'inicio'. Después de la tabla el valor de la carga permanece constante e igual al valor final de la tabla más el que pusimos en 'inicio'.

## ESTÁTICA

### Renumerar.

Renumerar los nudos de la estructura. Para almacenar los datos de la matriz global de rigidez de la estructura sólo precisamos una banda de ésta. El ancho de esta banda depende de la numeración de los nodos extremos de las barras, y es igual a la diferencia de numeración de la barra en la cual sea mayor ésta, multiplicada por tres. Por tanto el reducir este "ancho de banda" implicaría necesitar una menor capacidad de almacenamiento y menor tiempo de cálculo; y ese es el fin de la renumeración.

La presentación inicial es:



En el estudio lo que hace es tomar un nodo inicial y a partir de éste renumerar el resto. El número del inicial lo podemos introducir nosotros marcando la segunda opción, y podemos hacer varios tanteos con distintos puntos.

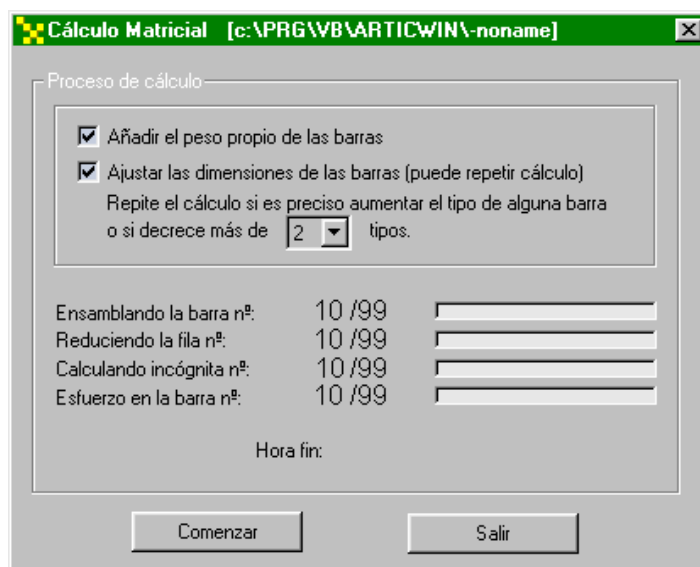
Con la primera opción lo que hace es ir probando esto con todos los puntos de la malla hasta encontrar el punto que origine el menor ancho de banda. Generalmente nos dará bastantes buenos resultados la opción 2; la opción 1 tarda bastante tiempo en completarse y sólo será precisa en casos muy especiales o si deseamos hacer un estudio de qué puntos de las mallas (centro, esquinas...) producen los menores anchos de banda.

Comenzado el estudio en la pantalla nos mostrará los estudios que todavía tienen que completarse (si estamos en el primer caso), y los puntos que nos faltan en cada estudio. En

el caso de encontrarse una mejora nos dirá en qué punto se halló y cual sería el nuevo ancho de banda, y nos pregunta si deseamos que se efectúe la reenumeración, a lo que responderemos con sí o no.

### Cálculo Matricial.

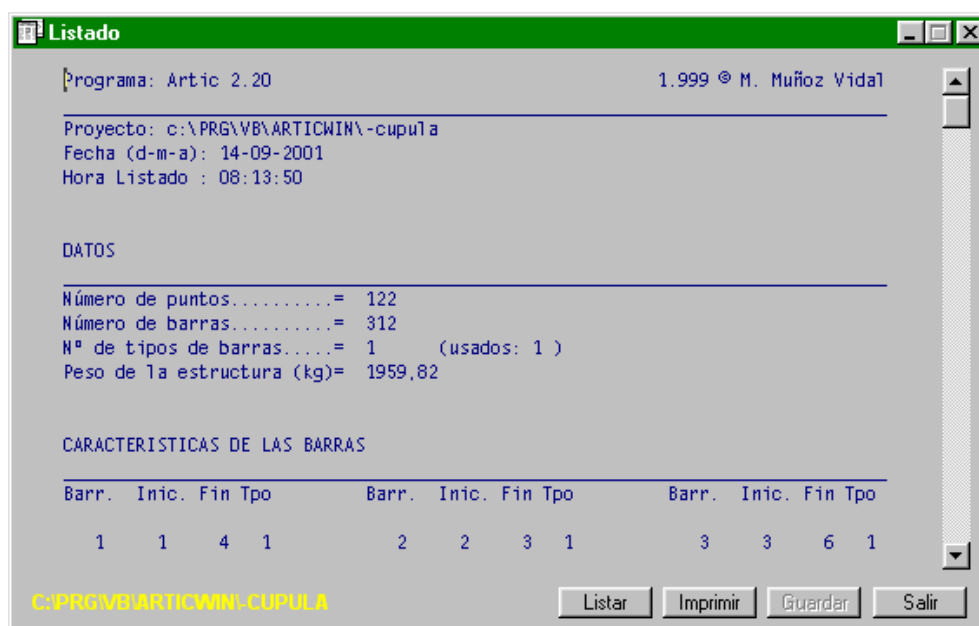
Realiza un cálculo matricial estático de la estructura.



Aparte de hallar los esfuerzos calcula el tipo de tubo (según la tabla vista) que es preciso, teniendo en cuenta el axil y el pandeo. Si la barra calculada no coincide con la inicialmente asignada, se vuelve a iniciar un cálculo con los nuevos tipos de barras; esto se repetirá hasta que no sean precisos más cambios. Para que la función de dimensionado automático funcione es preciso que los tipos de barras estén ordenados de menor a mayor resistencia.

### Listados.

Lista los datos de la estructura a un fichero de texto. Si queremos podemos editar este fichero y luego volver a importar las modificaciones efectuadas.



## DINÁMICA

### Cálculo Dinámico.

Realiza el cálculo dinámico de la estructura.

Arriba a la izquierda deberemos escribir la extensión (tres caracteres) que tendrán todos los ficheros del cálculo que vamos a efectuar, y debajo podemos escribir una descripción de este cálculo. A la derecha vemos una lista de todos los cálculos ya efectuados. Si hacemos doble click sobre un cálculo efectuado se nos mostrarán los parámetros empleados en el mismo. Si señalamos un cálculo ya efectuado y pulsamos el botón 'coge datos', los parámetros de ese cálculo pasarán a esta ventana.

Debajo de estos se puede marcar la casilla 'continuación de', que consiste en que el cálculo se retoma (manteniendo aceleraciones y velocidades) en la posición dejada por el cálculo anterior. Esto es útil si queremos proseguir un cálculo que veamos incompleto, o si queremos subdividir un cálculo grande en varios más cortos. Pero hay que tener en cuenta aquí que el programa no hace una estimación del tiempo de cálculo total teniendo en cuenta los cálculos anteriores (alguno de los cuales podría faltar pues solo precisamos la presencia del último cálculo que queramos proseguir) en el sentido que si tenemos cargas o aceleraciones dinámicas, estas volverán a aplicarse tomando como tiempo cero el tiempo de arranque del cálculo continuación. Entonces si tenemos cargas de este tipo les tenemos que poner como tiempo de inicio el valor de inicio que tenía la carga en el primer cálculo menos el tiempo total transcurrido desde el primer cálculo hasta el arranque del actual.

En **parámetros generales** indicaremos si queremos usar nudo intermedio o no. (si es una malla desplegable debemos poner que sí), si efectuamos corrección por axil y si consideramos el peso propio de la estructura en el cálculo.

En **grabar** seleccionamos qué parámetros queremos grabar. En este apartado es interesante afinar un poco, pues estos tipos de datos son los que más espacio de disco ocuparán, en función del número de iteraciones a efectuar. Normalmente posición y velocidad ocupan cuatro veces más que los distintos esfuerzos. Para no consumir disco en exceso, las aceleraciones las obtendrá el propio programa a partir de las velocidades.

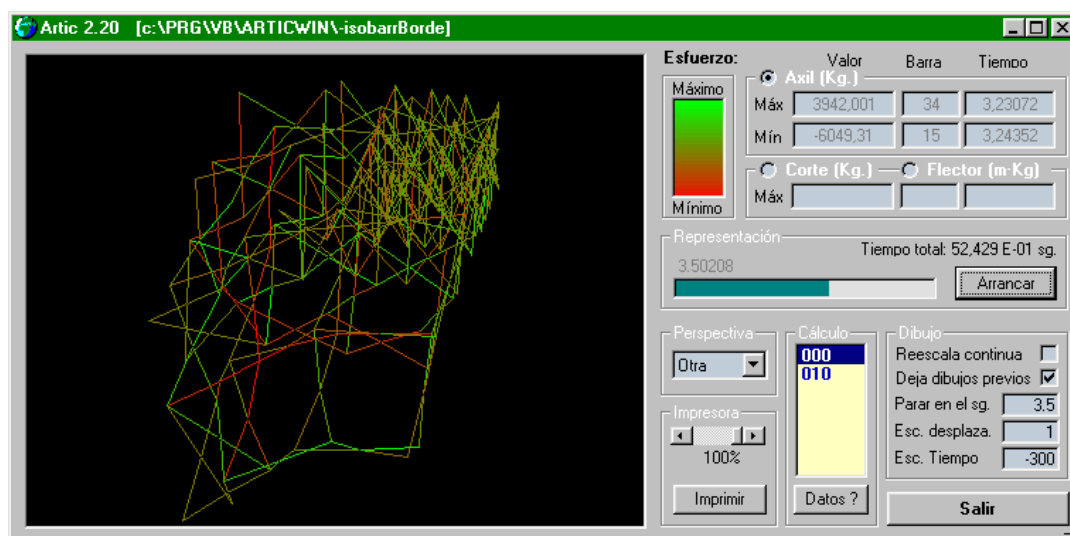
En **iteraciones** definiremos el tiempo de cálculo de cada iteración y el número total de iteraciones. Tiempos que funcionan bien son del orden de  $10^{-5}$  sg. para las estructuras de dimensiones habituales. El número de iteraciones puede ser cualquiera aunque si vamos a efectuar un posterior estudio mediante la Transformada Rápida de Fourier, es conveniente elegir potencias de dos. Para evitar consumir excesivo espacio en disco, en vez de grabar los resultados de todas las iteraciones podemos grabar una de cada varias, fijando este valor en 'relación de grabación'.

Con los datos que fijamos en el apartado anterior, en **Datos de cálculo** nos vendrá dado el tiempo de cálculo total, y dos parámetros que solo serán útiles de cara a una posible TRF como son la frecuencia máxima que podrá analizar esta así como el error de frecuencia esperado, habida cuenta que analizaremos un sistema discretos y por tanto obtendremos un sistema discreto de puntos de frecuencias espaciados el error de frecuencia.

En amortiguamiento definiremos el tipo de amortiguamiento y su valor en  $sg.^{-1}$

### Análisis Global.

Con este módulo analizaremos los resultados obtenidos del cálculo estudiando el modelo mediante una vista del despliegue y obtendremos los esfuerzos máximos de la estructura.



En la parte izquierda tendremos una vista de la estructura en cada momento del cálculo, y a la derecha diversos parámetros: arriba una indicación de que esfuerzo vamos a representar, axil, cortante o flector. Cortantes y flectores normalmente están relacionados por la longitud de la semibarra, por lo que son proporcionales entre sí, y solo obtendremos valores positivos, pues no tenemos criterio para fijar en una barra unidimensional un sistema positivo-negativo. El axil podrá ser positivo o negativo. Una vez que le demos al botón 'arrancar' y pare la representación gráfica, en estas casillas superiores figurarán los esfuerzos máximos analizados, en qué barra se producen y el tiempo de cálculo en que se dan.

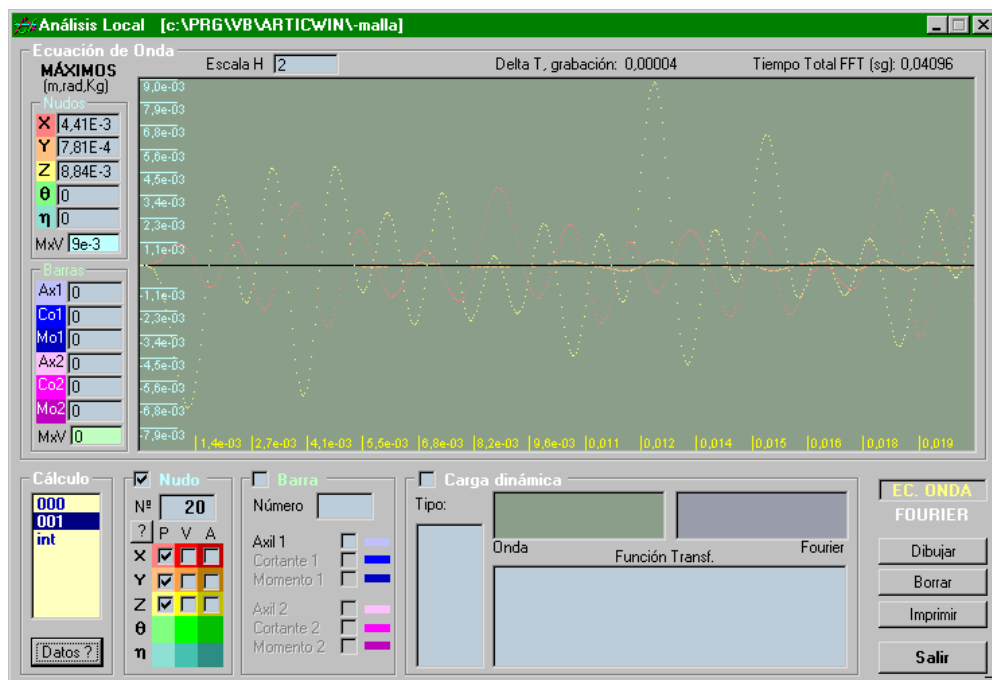
En **perspectiva** podemos elegir otra vista y en cálculo seleccionaremos el cálculo a estudiar. Con el proceso de representación parado podemos dar al botón 'imprimir' para obtener la figura actual por impresora.

En **dibujo** tenemos varias opciones. 'Reescala continua' significa que el dibujo de la estructura se reescalará en todo momento para ocupar el máximo posible dentro del marco de dibujo. 'Deja dibujos previos' si está marcada si está marcada hace que los dibujos previos no se borren. 'Parar en el sg.' fuerza a que el proceso de representación se detenga cuando se alcance dicho segundo. 'Esc. desplazamientos' es útil cuando estudiamos desplazamientos muy pequeños comparados con el tamaño de la estructura y queremos ampliarlos para ver su comportamiento. Cuando estudiamos el despliegue de estructuras, su valor será habitualmente 1. 'Esc. tiempo' es para retardar el proceso de dibujo cuando damos valores altos. Si damos valores negativos se acelerará, saltándose tantas iteraciones como valor se indique aquí (prescindiendo del signo). Combinando esta opción con 'Deja dibujos previos' podemos conseguir un dibujo con varias posiciones de la estructura igualmente espaciadas en el tiempo.

Debemos hacer notar que para que si tenemos activada la opción 'Deja dibujos previos' si se pone otra ventana de Windows sobre esta y luego se aparta, en la parte solapada habrá desaparecido el dibujo de la estructura, por lo que debemos tener cuidado si queremos capturar el dibujo. Esto se debe a las opciones usadas durante el proceso de dibujo para no ralentizar demasiado este.

### Análisis Local.

Es para el estudio de una barra o nudo concreto. Podemos ver la ecuación de onda de nudos o esfuerzos de barras o bien la TRF de estas ondas.



En primer lugar seleccionamos abajo a la izquierda qué cálculo queremos estudiar. Pulsando el botón 'datos' podemos ver los datos del cálculo seleccionado.

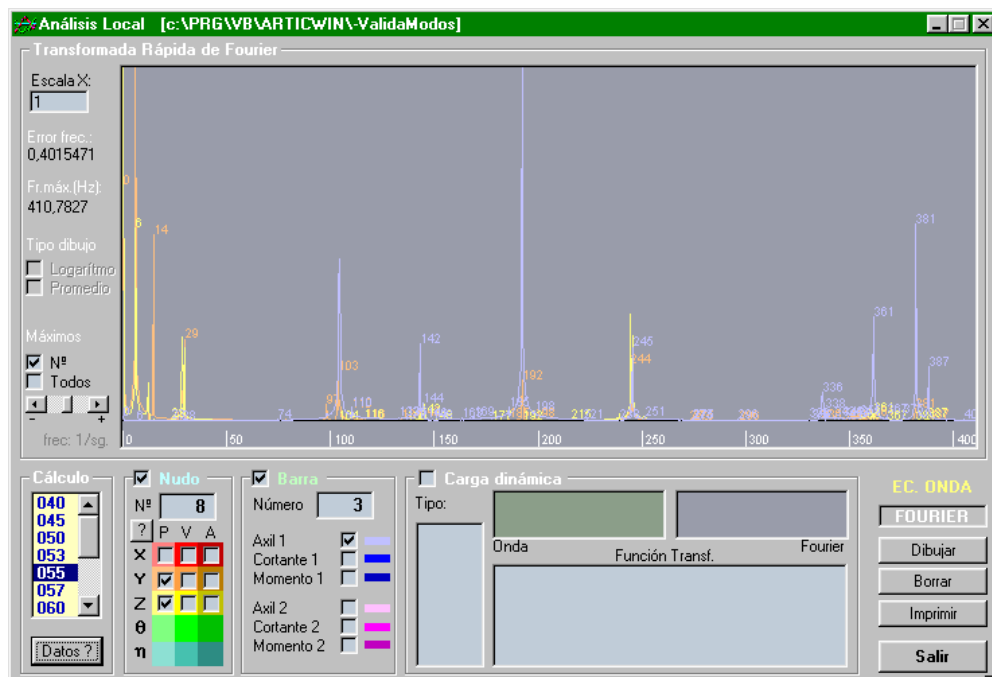
Para seleccionar un nudo debemos marcar la casilla nudo, debajo introducir su número y mas abajo marcamos que parámetros queremos representar: P-posiciones V-Velocidades o A-aceleraciones y en que dirección del espacio X, Y, Z. Las posiciones se dibujan relativas a la posición inicial del nudo, ya que de lo contrario, para desplazamientos muy pequeños en relación con su coordenada no apreciaríamos la onda. Para barras actuaríamos de forma similar.

Una vez elegidos los parámetros a representar apretaremos el botón '**Dibujo**' con lo que se leerán los datos y se dibujarán. En principio las ondas se ajustarán para que encajen dentro de la ventana lo más ajustadas posibles. Los valores máximos de los parámetros elegidos se reflejarán en las casillas del margen izquierdo. Si en las casillas '**MxV**' que existen para nudos introducimos un valor distinto de cero, las escalas verticales se ajustarán para este valor, y podremos comparar gráficamente los valores X,Y,Z, a la vez que nos aparece una escala vertical, a la izquierda de la zona de dibujo para los nudos y a la derecha para las barras.

La escala horizontal de tiempos también la podemos modificar introduciendo un factor multiplicador en la casilla '**Escala H**'.

Podemos comparar las ondas de varios nudos distintos efectuando repetidamente la operación de dibujar para cada uno de ellos, ya que por defecto los gráficos no se borran. Para borrarlos debemos pulsar el botón '**borrar**'.

Si ahora apretamos el rótulo '**FOURIER**' pasaremos a la pantalla que realiza la TRF.



La parte inferior es idéntica a la de ondas y para que aparezca el dibujo de la TRF de la onda debemos volver a pulsar el botón '**Dibujar**'.

En parte izquierda del dibujo también podemos introducir un factor para la escala horizontal de las frecuencias. Debajo aparecen los datos que ya vimos en la pantalla de cálculo, el error de frecuencia y el valor de la frecuencia máxima analizable.

El epígrafe '**Máximos**' hace referencia al valor de los picos de la gráfica, que serán las frecuencias principales. La casilla '**N°**' es para que aparezca un número al lado de cada máximo con el valor de la frecuencia que le corresponde. Si marcamos la casilla '**Todos**' se numerarán todos los máximos que aparezcan. En algunos casos la cantidad de los números que aparecen es tal que debemos efectuar una gradación para que no aparezcan demasiados, pero sí figuren todos los importantes. Para ello tenemos la barra de desplazamiento inferior, desplazándola hacia el + aparecerán más valores, y hacia el otro lado menos.



---

## 7.1.3 ESTRUCTURA DE FICHEROS.

---

Los ficheros que se crean en el directorio de cada estructura, y el significado y contenido de cada uno de ellos lo resumimos a continuación.

### DEFINICIÓN DE LA ESTRUCTURA

#### **DATOS.DAT**

nº puntos (np)

nº Barras(ne)

nº barras enlazadas (nea)

**NOTAS.DAT** Fichero de texto con comentarios acerca de la estructura.

**COORDENA.DAT** Coordenadas de los puntos en metros (3\*np).

x1

x2

...

y1

y2

...

z1

z2

....

**BARRAS.DAT** datos de las barras (3\*ne)

nudo origen 1

nudo final 1

tipo 1

nudo origen 2

nudo final 2

tipo 2

...

**TIPOS.DAT**

título

nº de tipos

Datos tipo 1:

Area.cm2....

Inercia(cm4)...

M+Young.kg/cm2.....

def+.....

M-Young.kg/cm2....

def-.....

T.adm.kg/cm2..

P.esp.kg/m3.....

nombre descriptivo.

**COACCION.DAT** Coacciones en los nudos (3\*np)

coacción x1  
coacción y1  
coacción z1  
coacción x2  
coacción y2  
coacción z2  
....

**CARGAS.DAT** cargas estáticas(Kg) sobre los nudos (3\*np)

Cargas x1  
Cargas y1  
Cargas z1  
Cargas x2  
Cargas y2  
Cargas z2  
....

**BARRASA.DAT** enlaces entre barras (2\*nea)

barra origen 1  
barra destino 1  
barra origen 2  
barra destino 2  
...

**COORDENL.DAT** Coordenadas de los puntos de enlace de las barras (6\*nea)

x1  
y1  
z1  
x2  
y2  
z2  
....

**SLONGITL.DAT** Coordenadas de los puntos de enlace de las barras (2\*nea)

SL·barr·orig1  
SL·barr·dest1  
SL·barr·orig2  
SL·barr·dest2  
SL·barr·orig3  
SL·barr·dest3  
....

**LONGITUD.DAT** Longitudes (mts) en equilibrio de las barras (ne)

longitud de la barra 1  
longitud de la barra 2  
...

**CARGASD.DAT** Tipo de carga dinámica en cada nudo (3\*np)

tipo x1  
tipo y1  
tipo z1  
tipo x2

tipo y2

tipo z2

...

Valores posibles:

-3 - Tabla 3 (CARGD3.DAT)

-2 - Tabla 2 (CARGD2.DAT)

-1 - Tabla 1 (CARGD1.DAT)

0 - Sin fuerza dinámica

1 - Función rampa ampliada

2 - Función rampa ampliada

3 - Función rampa ampliada

4 - Función Seno ampliada

5 - Función Seno ampliada

6 - Función Seno ampliada

## PRQD.DAT

Datos de los 10 tipos de cargas dinámicas (40 valores)

### (tipo fuerza -3)

Valor fuerza en instante cero (kg)

Valor amplitud fuerza en kg. (si valor máx. tabla es +/-1). Se suma a la fuerza inicial.

Tiempo (sg) de duración de una fuerza inicial cte. de valor el primer nº de la tabla.

Tiempo (sg) de duración total de la tabla. (luego la fuerza final permanece cte.)

### (tipo fuerza -2)

Valor fuerza en instante cero (kg)

Valor amplitud fuerza en kg. (si valor máx. tabla es +/-1). Se suma a la fuerza inicial.

Tiempo (sg) de duración de una fuerza inicial cte. de valor el primer nº de la tabla.

Tiempo (sg) de duración total de la tabla. (luego la fuerza final permanece cte.)

### (Tipo fuerza -1)

Valor fuerza en instante cero (kg)

Valor amplitud fuerza en kg. (si valor máx. tabla es +/-1). Se suma a la fuerza inicial.

Tiempo (sg) de duración de una fuerza inicial cte. de valor el primer nº de la tabla.

Tiempo (sg) de duración total de la tabla. (luego la fuerza final permanece cte.)

### (Tipo 0)

0

0

0

0

### (Tipo fuerza 1)

Valor fuerza en instante cero (kg)

Valor final fuerza en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la rampa.

### (Tipo fuerza 2)

Valor fuerza en instante cero (kg)

Valor final fuerza en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la rampa.

### **(Tipo fuerza 3)**

Valor fuerza en instante cero (kg)

Valor final fuerza en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la rampa.

### **(Tipo fuerza 4)**

Valor fuerza en instante cero (kg)

Amplitud función seno en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la función seno.

### **(Tipo fuerza 5)**

Valor fuerza en instante cero (kg)

Amplitud función seno en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la función seno.

### **(Tipo fuerza 6)**

Valor fuerza en instante cero (kg)

Amplitud función seno en kg., que se suma a la fuerza inicial.

Tiempo (sg) de duración de la fuerza inicial cte.

Tiempo (sg) de duración total de la función seno.

**CARGD1.DAT** Tabla 1 de carga dinámica. (los valores deberían estar entre -1 y 1)

Nº de puntos de la tabla.

Descripción de la tabla (texto)

valor 1

valor 2

valor 3

...

**CARGD2.DAT** Tabla 2 de carga dinámica. (los valores deberían estar entre -1 y 1)

Nº de puntos de la tabla.

Descripción de la tabla (texto)

valor 1

valor 2

valor 3

...

**CARGD3.DAT** Tabla 3 de carga dinámica. (los valores deberían estar entre -1 y 1)

...

**ACELED.DAT** (similar a CARGAD.DAT... pero para aceleraciones)

...

**ARQD.DAT** Datos de los 10 tipos de aceleraciones (40 valores) (similar a PRQD.DAT...)

**AD1.DAT** (Tablas de aceleraciones...)

**AD2.DAT** ...

**AD3.DAT** ...

**YOUNG.DAT** Tabla que define la gráfica tensión-deformación del material.

Nº de puntos de que consta la tabla (equidistantes), con valor entre 0 y 1

título (Texto)

valor 1

valor 2

...

## RESULTADOS DEL CÁLCULO

(Todos la extensión genérica: EXT)

**PARAMETR.EXT** Datos generales del cálculo:

Descripción datos (texto)

Delta tiempo (sg)

Relación de iteración/grabación (itergrb)

Tipo de amortiguamiento (texto)

Constante de amortiguamiento (sg-1)

Nº máximo de iteraciones (itermax)

Velocidad máxima admisible (m/s)

Si se considera el peso propio (texto)

Si de usa nudo intermedio (texto)

Extensión del cálculo anterior (texto)

Si es continuación cálculo anterior (texto)

Si Calcula corrección por axil (texto)

Breve texto descriptivo del calculo (texto)

Si se graban posiciones (texto)

Si se graban velocidades (texto)

Si se graban axiles (texto)

Si se graban cortantes (texto)

Si se graban flectores (texto)

Frecuencia máxima analizable (Hz)

Error de frecuencia (Hz)

Tiempo total de cálculo (sg)

**RESULTC.EXT** Datos de las posiciones de los nudos a lo largo del proceso. (binario)

Sin nudo intermedio: (nº de datos=  $3*(np)*itermax/itergrb$ )

(iteración 1) x1 y1 z1

x2 y2 z2

...

(iteración 2) x1 y1 z1

x2 y2 z2

...

con nudo intermedio: (nº de datos=  $3*(np)*itermax/itergrb+(5*(ne)*itermax/itergrb)$ )

(iteración 1) x1 y1 z1

x2 y2 z2

...

theta(np+1) eta(np+1)

theta(np+2) eta(np+2)

...

(iteración 2) x1 y1 z1

x2 y2 z2

...  
theta(np+1) eta(np+1)  
theta(np+2) eta(np+2)  
...

**RESULTV.EXT** Datos de las velocidades de los nudos a lo largo del proceso. (binario)

Sin nudo intermedio: (nº de datos=  $3*(np)*itermax/itergrb$ )

(iteración 1) Vx1 Vy1 Vz1  
Vx2 Vy2 Vz2

...  
(iteración 2) Vx1 Vy1 Vz1  
Vx2 Vy2 Vz2

...

con nudo intermedio: (nº de datos=  $3*(np)*itermax/itergrb)+(5*(ne)*itermax/itergrb$ )

(iteración 1) Vx1 Vy1 Vz1  
Vx2 Vy2 Vz2

...  
Vtheta(np+1) Veta(np+1)  
Vtheta(np+2) Veta(np+2)

...  
(iteración 2) Vx1 Vy1 Vz1  
Vx2 Vy2 Vz2

...  
Vtheta(np+1) Veta(np+1)  
Vtheta(np+2) Veta(np+2)

...

**AXIL.EXT** Axiles de las semibarras en función del tiempo. (binario)

**CORT.EXT** Cortante de las semibarras en función del tiempo.(binario)

**FLECT.EXT** Flector(binario)

---

## 7.2 EL PROCESO DE GENERACIÓN DE MALLAS ESPACIALES

---

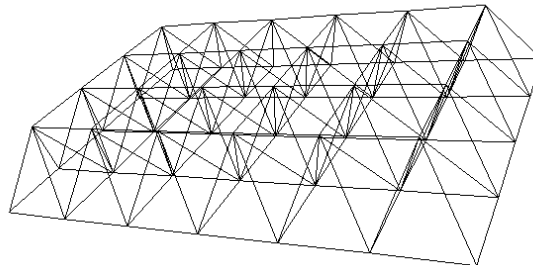
Como anexo independiente, dentro de lo que es el programa también debemos destacar este apartado. No es específico de la generación de mallas desplegadas, que ya forman parte del cuerpo central de este estudio, sino de mallas espaciales planas de una o varias capas de nudos. Por su interés lo destacamos y comentamos con un poco más de detalle.

---

### 7.2.1 GENERACIÓN

---

El tipo de mallas que podemos generar automáticamente con este programa se refiere a mallas planas compuestas por módulos que se repitan según dos direcciones principales.



El proceso básico de generación consiste en la repetición de un módulo que hayamos predefinido en las dos direcciones que deseemos  $N$  veces en una y  $M$  veces en la otra. Los vectores dirección, aparte de indicarnos ésta, también nos suministran la información del desplazamiento que debemos darle al módulo entre una repetición y la siguiente para que los nodos finales de uno coincidan con los iniciales del módulo siguiente. Pero esta concepción nos sirve sólo para el caso más sencillo y prácticamente imposible de que los elementos del borde no difieran del resto del interior. Pues lo más usual es que nos sobrasen o nos faltasen algunas barras del módulo tipo al llegar a los bordes, y también puede ocurrir, como en el caso de la malla tetraédrica que los remates de borde sean substancialmente distintos de los del interior, y más todavía los casos singulares de las esquinas.

Por todo ello la solución más completa que hemos hallado aquella en la que definimos todas estas particularidades. Tendremos pues un total de nueve módulos distintos (o prácticamente iguales) a definir, que serán el módulo tipo del interior, uno para cada uno de los bordes y otro para cada una de las cuatro esquinas. El proceso será entonces casi igual al inicialmente descrito, variando únicamente en que al llegar a una singularidad (bordes o esquinas), sustituye el módulo tipo por el que corresponda en cada caso.

Como decíamos al principio este método nos permitirá generar mallas espaciales cuyo contorno tendrá forma de paralelogramo en planta: cuadrado, rectángulos, rombos y trapecios. Con los módulos ya es más flexible, ya que tipos que en teoría seguirían tres direcciones principales, como los tetraédricos, son admitidos para la generación de mallas sin mayores problemas.

Si nos proponemos incorporar nosotros mismos un nuevo módulo conviene que aparte de que como antes decíamos los nodos finales y origen coincidan, también habrá que tener presente que la repetición de los módulos no conlleve que se produzcan duplicidades de barras uniendo los mismos nodos; pues el programa no detectará si hay más de una barra uniendo dos nodos. Para una explicación más detallada acudir al apartado de DEFINICIÓN DE NUEVOS MÓDULOS.

Una vez efectuada y comprobada la definición del módulo que deseemos, el programa se encarga de la creación de la malla, y en síntesis los pasos principales son los siguientes:

## 1) LECTURA DE LOS DATOS DEL MÓDULO

Se realiza en las líneas que siguen. El significado de las distintas variables se comenta en capítulos posteriores:

```

○ If encontrado Then
○   Line Input #1, De$
○   Input #1, NCP
○   For i = 1 To 2: For j = 1 To 3: Input #1, Dire(i, j): Next j: Next i
○   Input #1, b, p, BA1OR, BA1EX, BA2OR, BA2EX, BEOO, BEEO, BEOE, BEEE
○   ReDim MB(b, 2), MP(p, 4), cp(p, 2), BA1ORm(BA1OR, 2), BA1EXm(BA1EX, 2)
○   ReDim BA2ORm(BA2OR, 2), BA2EXm(BA2EX, 2)
○   ReDim BEOOm(BEOO, 2), BEEOm(BEEO, 2), BEOEm(BEOE, 2), BEEEm(BEEE, 2)
○   ReDim carNCP(NCP, 3)
○   For i = 1 To b: For j = 1 To 2: Input #1, MB(i, j): Next j: Next i
○   For i = 1 To p: For j = 1 To 4: Input #1, MP(i, j): Next j: Next i
○   For i = 1 To BA1OR: For j = 1 To 2: Input #1, BA1ORm(i, j): Next j: Next i
○   For i = 1 To BA1EX: For j = 1 To 2: Input #1, BA1EXm(i, j): Next j: Next i
○   For i = 1 To BA2OR: For j = 1 To 2: Input #1, BA2ORm(i, j): Next j: Next i
○   For i = 1 To BA2EX: For j = 1 To 2: Input #1, BA2EXm(i, j): Next j: Next i
○   For i = 1 To BEOO: For j = 1 To 2: Input #1, BEOOm(i, j): Next j: Next i
○   For i = 1 To BEEO: For j = 1 To 2: Input #1, BEEOm(i, j): Next j: Next i
○   For i = 1 To BEOE: For j = 1 To 2: Input #1, BEOEm(i, j): Next j: Next i
○   For i = 1 To BEEE: For j = 1 To 2: Input #1, BEEEm(i, j): Next j: Next i
○   Input #1, ESCX, ESCY, ESCZ
○ End If

```

Además de la lectura de los datos de los módulos y direcciones principales, también están incluidos los datos de las cargas en cada una de las capas que la componen.

## 2) NÚCLEO DE LA GENERACIÓN

```

○ MODULO = 0
○ Do
○   Modul.Caption = Str$(MODULO)
○   LeeModulo (MODULO)
○
○   For i = 1 To M1: For j = 1 To M2
○     O1 = ORX + Dire(1, 1) * (i - 1) + Dire(2, 1) * (j - 1) ' Nuevo origen
○     O2 = ORY + Dire(1, 2) * (i - 1) + Dire(2, 2) * (j - 1) ' de cada
○     O3 = ORZ + Dire(1, 3) * (i - 1) + Dire(2, 3) * (j - 1) ' módulo
○     For R = 1 To p 'Para cada punto del modulo.
○       buscapunto R
○     Next R
○
○     Caso = Abs((10 * (i = 1) + 20 * (i = M1)) + (1 * (j = 1) + 2 * (j = M2)))
○     Select Case Caso
○       Case 11 '##### REMATE ESQUINA O,O #####
○         For R = 1 To BEOO
○           n1 = BEOOm(R, 1): n2 = BEOOm(R, 2)
○           OtraBarra n1, n2
○         Next R
○       Case 21 '##### REMATE ESQUINA E,O #####
○         For R = 1 To BEEO
○           n1 = BEEOm(R, 1): n2 = BEEOm(R, 2)

```



```

O      OtraBarra n1, n2
O      Next R
O      Case 12 '##### REMATE ESQUINA O,E #####
O      For R = 1 To BEOE
O      n1 = BEOEm(R, 1): n2 = BEOEm(R, 2)
O      OtraBarra n1, n2
O      Next R
O      Case 22 '##### REMATE ESQUINA E,E #####
O      For R = 1 To BEEE
O      n1 = BEEEm(R, 1): n2 = BEEEm(R, 2)
O      OtraBarra n1, n2
O      Next R
O      Case 1 '##### REMATE BORDE DIR. 1, ORIGEN #####
O      For R = 1 To BA1OR
O      n1 = BA1ORm(R, 1): n2 = BA1ORm(R, 2): OtraBarra n1, n2
O      Next R
O      Case 2 '##### REMATE BORDE DIR. 1, EXTREMO #####
O      For R = 1 To BA1EX
O      n1 = BA1EXm(R, 1): n2 = BA1EXm(R, 2): OtraBarra n1, n2
O      Next R
O      Case 10 '##### REMATE BORDE DIR. 2, ORIGEN #####
O      For R = 1 To BA2OR
O      n1 = BA2ORm(R, 1): n2 = BA2ORm(R, 2): OtraBarra n1, n2
O      Next R
O      Case 20 '##### REMATE BORDE DIR. 2, EXTREMO #####
O      For R = 1 To BA2EX
O      n1 = BA2EXm(R, 1): n2 = BA2EXm(R, 2): OtraBarra n1, n2
O      Next R
O      Case 0 '##### MODULO TIPO #####
O      For R = 1 To b
O      n1 = MB(R, 1): n2 = MB(R, 2): OtraBarra n1, n2
O      Next R
O      Case Else
O      Print "error": Stop
O      End Select
O
O      DoEvents
O      Next j: Next i
O      MODULO = MODULO + 1
O      Loop Until fin = 1

```

Primero para todos los puntos del módulo, realiza una llamada a la subrutina que busca si existe con las coordenadas dadas otro punto, en cuyo caso le daría su numeración, y si no existiese incrementaría el nº de barras de la malla en uno y le daría ese nº a este nodo:

```

O Private Sub buscapunto(R)
O '***** Subrutina busqueda. Numeracion de puntos *****
O Dim S As Long, DMS As Single, D1S As Single, D2S As Single, D3S As Single
O Dim encontrado As Integer
O Dim cp As Integer
O '*****
O encontrado = 0
O cc(R, 1) = MP(R, 1) + O1
O cc(R, 2) = MP(R, 2) + O2
O cc(R, 3) = MP(R, 3) + O3
O S = 0
O DMS = 0.005 'distancia para considerar dos puntos coincidentes
O Do While S < np
O S = S + 1 'Busca si los puntos ya existen...
O D1S = Abs(C(S, 1) - cc(R, 1)): D2S = Abs(C(S, 2) - cc(R, 2)): D3S = Abs(C(S, 3) -
O cc(R, 3))
O If D1S < DMS Then If D2S < DMS Then If D3S < DMS Then encontrado = 1: Exit Do
O Loop
O If encontrado = 1 Then
O 'el punto ya existe.....
O cc(R, 4) = S
O Else
O 'añadir un nuevo punto.....
O np = np + 1
O cp = MP(R, 4)
O RR(np, 1) = carNCP(cp, 1)
O RR(np, 2) = carNCP(cp, 2)
O RR(np, 3) = carNCP(cp, 3)
O C(np, 1) = cc(R, 1)
O C(np, 2) = cc(R, 2)
O C(np, 3) = cc(R, 3)

```

```

○      cc(R, 4) = np
○ End If
○ End Sub

```

Después para cada modulo lo que hace es que para cada barra introduce el número del nodo inicial y final de la misma en las variables N1 y N2, y llama a la subrutina de añadir barras:

```

○ Private Sub OtraBarra(n1, n2)
○ '***** Subrutina para anadir una nueva barra *****
○ Dim L1 As Long, L2 As Long
○ L1 = cc(n1, 4)
○ L2 = cc(n2, 4)
○ If L1 > L2 Then Swap L1, L2
○ ne = ne + 1
○ L(ne, 1) = L1
○ L(ne, 2) = L2
○ DibBarra ne
○ End Sub

```

Como antes comentábamos, esta subrutina no comprueba si entre los nodos a unir ya existía otra barra, sino que simplemente añade los datos de los nudos en las matrices correspondientes introduciéndolos de menor a mayor numeración e incrementa el nº de barras de la malla en uno.

---

## 7.2.2 DEFINICIÓN DE NUEVOS MÓDULOS

---

Si queremos introducir un nuevo módulo de malla espacial que no se halle predefinido basta con editar el fichero de texto MALLAS.DAT que se halla en el directorio del programa para poder introducir los datos de nuestro módulo como pasamos a describir:

### DESCRIPCIÓN DE UN MÓDULO.

Cada tipo de módulo tiene una definición similar a la siguiente:

```
xxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Tipo05 Pirámides cuadradas, corn.
Pirámides de base cuadrada, con borde en cornisa
2,1,0,0,0,1,0
8,7,8,8,8,8,8,8,8,8
1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
0,0,1,2, 0,1,1,2, 1,0,1,2, 1,1,1,2, .5,.5,0,1, .5,1.5,0,1, 1.5,.5,0,1
1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7
1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4
1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4
1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7
1,3,1,2,1,5,5,2,5,4,5,3,2,4,3,4
1,1, .707106781
```

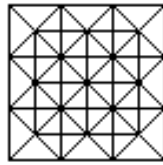
La primera línea es solo orientativa de cual debe ser la longitud del texto de la línea siguiente, por lo que podríamos prescindir de ella. Para cada nuevo tipo de módulo que introduzcamos es básico en la línea siguiente poner 'TipoNN' seguido de una breve descripción del módulo, que es la que aparecerá en la lista. En 'Tipo' debemos respetar las mayúsculas y minúsculas, y NN son dos caracteres cualquiera, que serán los que identifique unívocamente nuestro módulo, por lo que debemos tener cuidado de no repetirlos. Según esto, contando que el juego de caracteres nos permite unos 224 distintos, tenemos opción a definir 224x224 módulos distintos combinando dos caracteres cualesquiera, lo que parece más que suficiente para nuestras necesidades.

En la línea siguiente podemos efectuar una descripción más larga, que nos aparecerá cuando tengamos seleccionado este módulo, y en las siguientes es la descripción numérica concreta de nuestro módulo, que se dispondrán como describimos a continuación.

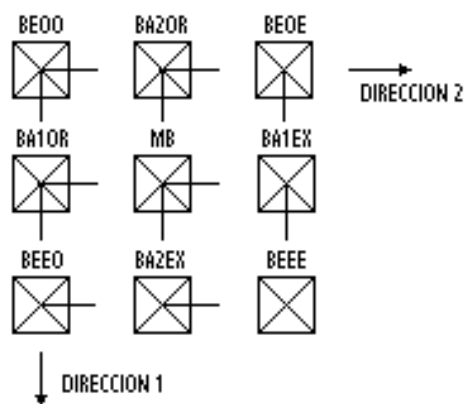
### DATOS NUMÉRICOS DEL MÓDULO.

Para ver como podemos hacer para definir un módulo, tomemos por ejemplo un caso sencillo como es el de pirámides de base cuadrada, pues en otros casos como el de módulos tetraédricos, la descripción se complicaría.

Lo más práctico para comenzar es realizar un dibujo de la expresión mínima de la malla en donde se den todos los casos particulares (remate de bordes y de esquinas):



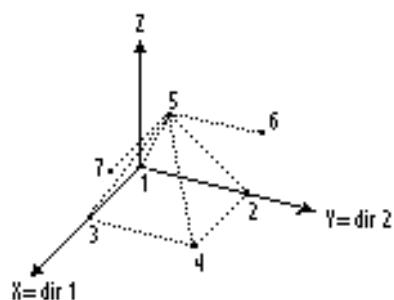
Una vez realizado esto deberemos es preciso descomponerla en los nueve módulos precisos cuya repetición nos asegure la formación de la malla deseada. La forma de definir la malla consiste en realidad en definir nueve módulos, que serán los siguientes: un módulo tipo a repetir en todo el interior de la malla, un módulo más por cada uno de los bordes, o sea cuatro más; y otro más por cada esquina, o sea otros cuatro. De esta forma vemos que no tendremos ningún problema en rematar los bordes como queramos y resolveremos las singularidades de las esquinas como creamos más conveniente.



Hecho esto definiremos el conjunto de puntos necesario y suficiente para que con él podamos representar cualquiera de los nueve módulos anteriores, éste lo almacenaremos en la matriz MP. Los datos que la componen serán las coordenadas de los puntos, y la capa a la que asignamos cada uno de los puntos. Esto nos servirá para a la hora de introducir las cargas en cada capa, el programa sepa a qué puntos nos referimos. De esta forma podemos definir las capas como normalmente se entienden, o sea, por niveles; o bien para diferenciar los nudos que van cargados de otros que normalmente no llevan carga o no interesa que la lleven por no poder realizarse el equilibrio de cargas (como en el caso de un nudo en medio de una barra con alguna componente de la carga perpendicular a ella). Esto se puede hacer por ejemplo definiendo estos puntos como pertenecientes a la capa cero, con lo cual el programa no nos preguntará la carga que les asignamos y le dará valor cero en las direcciones X,Y,Z ; pero si queremos si queremos poder introducir cargas en alguna dirección por ser factible el equilibrio podemos definir estos nudos como pertenecientes a una capa más de las que normalmente consta la malla y aclarando suficientemente esta condición especial.

En el caso de los módulos de los tipos que hemos predefinido y designado como A2 y A3 hemos preferido la primera de las opciones (capa 0), pues en los nudos que constituyen el encuentro de cuatro barras horizontales sólo se puede producir el equilibrio si introducimos cargas horizontales, y estas son poco usuales que se den en la realidad en los nudos intermedios de una malla plana.

En el caso que estábamos estudiando, el módulo puede quedar de la forma:



Para los distintos módulos de barras sólo será preciso dar las barras que componen dicho módulo, definidas éstas al indicar su nodo inicial y nodo final, siempre referidos a esta numeración de nudos.

La forma de definir los nueve módulos es mediante las barras que los integran, dando su nudo origen y su nudo extremo, y todos los módulos deben compartir el mismo conjunto de puntos, por tanto al definir el conjunto de puntos debemos colocar los suficientes y en los lugares adecuados para asegurarnos que con ellos podemos formar cualquiera de los nueve módulos de barras antes mencionados.

El orden y los datos a poner son los siguientes:

- NCP** (un dato) Número de capas de que consta la malla, generalmente serán dos; esto nos sirve a la hora de dar los valores de las cargas en los nudos para poder hacer que estos sean diferentes para cada capa.
- DI(2,3)** (seis datos) Vectores que indican las dos direcciones principales de desarrollo de la malla. Figuran primeramente los valores X,Y,Z de la dirección 1 y luego de la dirección 2. Aparte de la dirección también suministran la información del paso o incremento que se produce en cada dirección para situar la nueva fila de módulos, por tanto su dirección y magnitud estarán en función de la definición que realicemos del módulo. No tienen por qué ser direcciones ortogonales, por lo que podremos generar mallas en forma de romboide y superficies inclinadas.
- B** (un dato) Número de barras de que consta el módulo. En este caso no cuentan las barras a añadir o quitar para el remate de los bordes.
- P** (un dato) Número de puntos de que consta el módulo que hemos definido. Conviene recordar que se deben introducir también los puntos que únicamente sirvan de apoyo para barras de remate de un borde.
- BA1OR** (un dato) Numero de barras añadir en el remate del borde origen de la malla que tiene dirección paralela a la que hemos definido como 1.
- BA1EX** (un dato) Numero de barras añadir en el remate del borde extremo de la malla que tiene dirección paralela a la que hemos definido como 1.
- BA2OR** (un dato) Numero de barras añadir en el remate del borde origen de la malla que tiene dirección paralela a la que hemos definido como 2.
- BA2EX** (un dato) Numero de barras añadir en el remate del borde extremo de la malla que tiene dirección paralela a la que hemos definido como 2.
- BEOO** (un dato) Numero de barras añadir en la esquina origen, origen de ambas direcciones.

<b>BEE0</b>	(un dato) Numero de barras añadir en la esquina extremo de la dirección 1 y origen de la dirección 2.
<b>BEOE</b>	(un dato) Numero de barras añadir en la esquina origen de la dirección 1 y extremo de la dirección 2.
<b>BEEE</b>	(un dato) N° barras añadir en la esquina extremo, extremo de ambas direcciones.
<b>MB(B,2)</b>	(n° de barras *2 datos) Datos de las barras del módulo tipo, se pondrán primero el nodo inicial y el nodo final de la barra que hemos numerado como 1, luego la número 2, y así sucesivamente.
<b>MP(P,4)</b>	(n° de puntos *4 datos) Datos de los nudos del módulo, se pondrán la coordenada X, la coordenada Y, la coordenada Z y el número de la capa a que pertenece el punto 1, luego el número 2, etc. hasta llegar a P puntos.
<b>BA1OR(BA1OR,2)</b>	(n° de barras a añadir en dir.1, borde origen *2 datos) Datos de las barras que se añadirán en el borde origen que tiene la dirección que hemos definido como 1, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BA1EX(BA1EX,2)</b>	(n° de barras a añadir en dir.1, borde extremo *2 datos) Datos de las barras que se añadirán en el borde extremo que tiene la dirección que hemos definido como 1, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BA2OR(BA2OR,2)</b>	(n° de barras a añadir en dir.2, borde origen *2 datos) Datos de las barras que se añadirán en el borde origen que tiene la dirección que hemos definido como 2, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BA2EX(BA2EX,2)</b>	(n° de barras a añadir en dir.2, borde extremo *2 datos) Datos de las barras que se añadirán en el borde extremo que tiene la dirección que hemos definido como 2, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BEO0(BEO0,2)</b>	(n° de barras a añadir en esquina origen, origen *2 datos) Datos de las barras que se añadirán en la esquina origen, origen de ambas direcciones, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BEE0(BEE0,2)</b>	(n° de barras a añadir en esquina extremo, origen *2 datos) Datos de las barras que se añadirán en la esquina extremo de la dir.1 y origen de la dir.2, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.
<b>BEOE(BEOE,2)</b>	(n° de barras a añadir en esquina origen, extremo *2 datos) Datos de las barras que se añadirán en la esquina origen de la dir.1 y extremo de la dir.2, se pondrán el nodo inicial y el nodo final de cada una de las barras que

queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.

**BEEE(BEEE,2)** (nº de barras a añadir en esquina extremo, extremo \*2 datos) Datos de las barras que se añadirán en la esquina extremo, extremo de ambas direcciones, se pondrán el nodo inicial y el nodo final de cada una de las barras que queremos añadir, siempre de acuerdo con la numeración de nudos que hemos hecho en el módulo.

Una vez definido el módulo, antes de nada es preciso comprobar que su repetición produce la malla deseada, para ello iremos al programa de generación y generaremos una malla. Con ello podemos ver que no se dé ningún tipo de erro y que se genera la malla deseada. Otra cosa que debemos comprobar aquí es que no se dan duplicidades de barras, y si todo esto resulta satisfactorio ya tendremos definido un nuevo módulo.

## 7.2.3 DESCRIPCIÓN DE LOS MÓDULOS PREDEFINIDOS

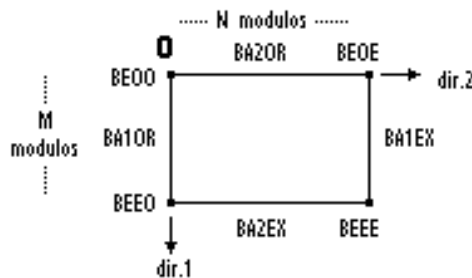
Pasaremos ahora a realizar una descripción más detallada de los módulos que tenemos predefinidos en el programa, así como del tipo de malla que generarán.

El significado de los distintos tipos de línea que se emplea en el dibujo de los módulos en el programa es el siguiente:

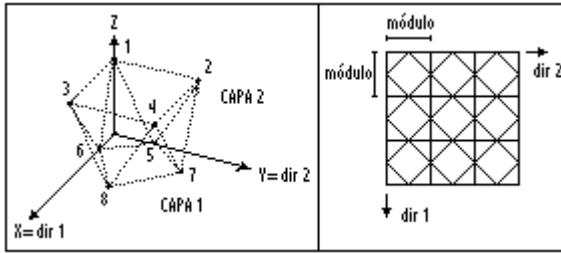
1	—————	3	.....
2	-----	4	.....

- 1.- barra tipo
- 2.- barras a añadir en los remates de los bordes
- 3.- ejes coordenados X,Y,Z
- 4.- direcciones principales de generación de malla

Los datos que se emplearán en la definición del módulo se explicarán mejor con la siguiente figura:



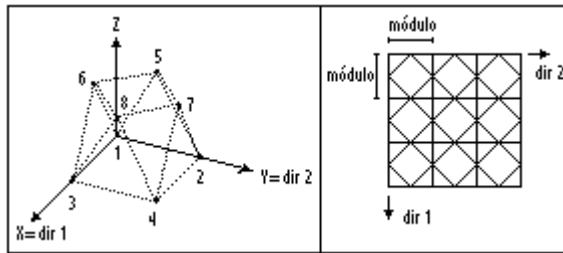
## ■ TIPO A0. PRISMAS CUADRADOS CON BASES EN DIAGONAL.



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	12
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	11
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	14
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	11
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	14
<b>BEEO</b>	Nº barras esquina origen,origen	12
<b>BEE0</b>	Nº barras esquina extremo,origen	14
<b>BE0E</b>	Nº barras esquina origen,extremo	14
<b>BE0E</b>	Nº barras esquina origen,extremo	14
<b>BEEE</b>	Nº barras esquina extremo,extremo	16
<b>MB(B,2)</b>	Datos de las barras, módulo tipo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8	
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos 0,0,1,2, 0,1,1,2, 1,0,1,2, 1,1,1,2, 0,.5,0,1, .5,0,0,1, .5,1,0,1, 1,.5,0,1, 0,1.5,0,1, 1.5,0,0,1, 1,1.5,0,1, 1.5,1,0,1	
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,6,10	
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,2,4,2,7,4,7,7,12	
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,5,9	
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,8,11	
<b>BE00(BE00,2)</b>	Datos barras esquina origen,origen 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,5,9,6,10	
<b>BEE0(BEE0,2)</b>	Datos barras esquina extremo,origen 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,8,11	
<b>BE0E(BE0E,2)</b>	Datos barras esquina origen,extremo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,2,4,2,7,4,7,7,12	
<b>BE0E(BE0E,2)</b>	Datos barras esquina origen,extremo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,2,4,2,7,4,7,7,12	
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo 1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,2,4,2,7,4,7	

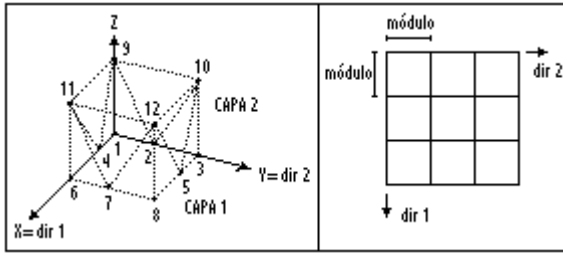


## ■ TIPO A1. PRISMAS CUADRADOS, CARA SUPERIOR DIAGONAL



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	12
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	11
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	14
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	11
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	14
<b>BEOO</b>	Nº barras esquina origen,origen	12
<b>BEEO</b>	Nº barras esquina extremo,origen	14
<b>BEOE</b>	Nº barras esquina origen,extremo	14
<b>BEEE</b>	Nº barras esquina extremo,extremo	16
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	0,0,0,1, 0,1,0,1, 1,0,0,1, 1,1,0,1, 0,.,5,1,2, .5,0,1,2, .5,1,1,2, 1,.,5,1,2, 0,1,5,1,2, 1,5,0,1,2, 1,1,5,1,2, 1,5,1,1,2
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,6,10
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,2,4,2,7,4,7,7,12
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,5,9
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,8,11
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,5,9,6,10
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,8,11
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,2,4,2,7,4,7,7,12
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	1,2,1,3,1,5,1,6,2,5,3,6,5,6,5,7,6,8,7,8,3,4,3,8,4,8,2,4,2,7,4,7

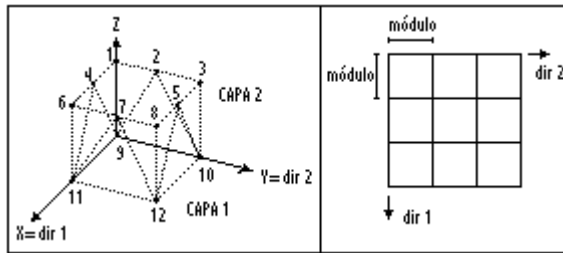
## ■ TIPO A2. PRISMAS CUADRADOS, DIAGONALES EN 'V'.



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	12
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	10
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	15
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	10
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	15
<b>BEOO</b>	Nº barras esquina origen,origen	11
<b>BEEO</b>	Nº barras esquina extremo,origen	16
<b>BEOE</b>	Nº barras esquina origen,extremo	16
<b>BEEE</b>	Nº barras esquina extremo,extremo	21
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	0,0,0,0, 0,5,0,1, 0,1,0,0, 5,0,0,1, 5,1,0,1, 1,0,0,0, 1,5,0,1, 1,1,0,0, 0,0,1,2, 0,1,1,2, 1,0,1,2, 1,1,1,2
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,1,9
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,11,12,7,11,7,12,7,8,7,6,6,11
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8,3,10
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8,11,12,7,11,7,12,7, 8,7,6,8,12

NOTA: en el caso de los nudos que son el encuentro de cuatro barras horizontales, aunque pertenecen a la capa 2 física, el programa no les pondrá la carga que hemos dado para esta capa, sino que será cero, pues las cargas verticales no se podrían equilibrar y las horizontales es poco usual que se den.

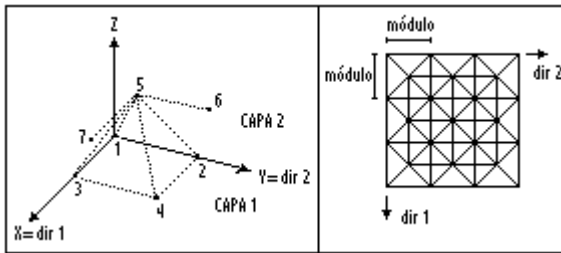
## ■ TIPO A3. PRISMAS CUADRADOS, DIAGONALES 'V' INVERTIDA



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	12
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	10
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	15
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	10
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	15
<b>BEOO</b>	Nº barras esquina origen,origen	11
<b>BEEO</b>	Nº barras esquina extremo,origen	16
<b>BEOE</b>	Nº barras esquina origen,extremo	16
<b>BEEE</b>	Nº barras esquina extremo,extremo	21
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	0,0,1,0,0,5,1,2,0,1,1,0,5,0,1,2,5,1,1,2,1,0,1,0,1,5,1,2,1,1,1,0,0,0,0,1,0,1,0,1,1,0,0,1,1,1,0,1
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,1,9
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,11,12,7,11,7,12,7,8,7,6,6,11
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8,3,10
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	9,10,9,11,9,2,9,4,10,2,11,4,1,2,1,4,2,3,4,6,10,12,5,10,5,12,5,3,5,8,11,12,7,11,7,12,7,8,7,6,8,12

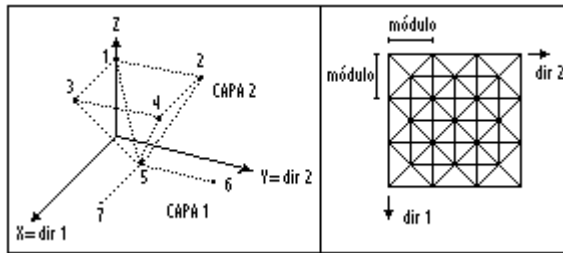
NOTA: en el caso de los nudos que son el encuentro de cuatro barras horizontales, aunque pertenecen a la capa 2 física, el programa no les pondrá la carga que hemos dado para esta capa, sino que será cero, pues las cargas verticales no se podrían equilibrar y las horizontales es poco usual que se den.

## ■ TIPO A4. PIRÁMIDES BASE CUADRADA, BORDE MANSARDA



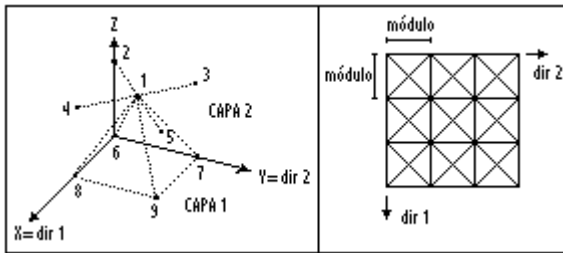
<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	8
<b>P</b>	Número de puntos	7
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	8
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	8
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	8
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	8
<b>BEOO</b>	Nº barras esquina origen,origen	8
<b>BEEO</b>	Nº barras esquina extremo,origen	8
<b>BEOE</b>	Nº barras esquina origen,extremo	8
<b>BEEE</b>	Nº barras esquina extremo,extremo	8
<b>MB(B,2)</b>	Datos de las barras, módulo tipo 1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7	
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos 0,0,0,1, 0,1,0,1, 1,0,0,1, 1,1,0,1, .5,.5,1,2, .5,1.5,1,2, 1.5,.5,1,2	
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen 1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7	
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo 1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7	
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen 1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7	
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo 1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4	
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen 1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7	
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen 1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4	
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo 1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7	
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo 1,3,1,2,1,5,5,2,5,4,5,3,2,4,3,4	

## ■ TIPO A5. PIRÁMIDES BASE CUADRADA, BORDE EN CORNISA



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	8
<b>P</b>	Número de puntos	7
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	8
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	8
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	8
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	8
<b>BEOO</b>	Nº barras esquina origen,origen	8
<b>BEEO</b>	Nº barras esquina extremo,origen	8
<b>BEOE</b>	Nº barras esquina origen,extremo	8
<b>BEEE</b>	Nº barras esquina extremo,extremo	8
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	0,0,1,2, 0,1,1,2, 1,0,1,2, 1,1,1,2, .5, .5,0,1, .5,1.5,0,1, 1.5, .5,0,1
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	1,3,1,2,1,5,5,2,5,4,5,3,5,6,5,7
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	1,3,1,2,1,5,5,2,5,4,5,3,5,6,3,4
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	1,3,1,2,1,5,5,2,5,4,5,3,2,4,5,7
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	1,3,1,2,1,5,5,2,5,4,5,3,2,4,3,4

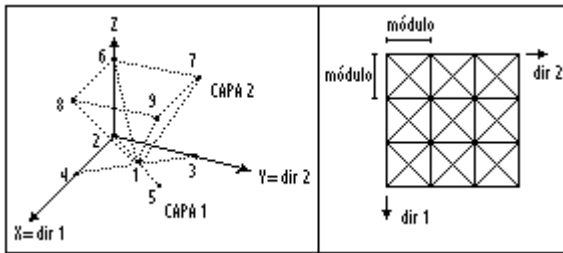
## ■ TIPO A6. PIRÁMIDES CUADRADAS, DIAGONAL SUPERIOR.



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	9
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	8
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	9
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	8
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	9
<b>BEOO</b>	Nº barras esquina origen,origen	7
<b>BEEO</b>	Nº barras esquina extremo,origen	8
<b>BEOE</b>	Nº barras esquina origen,extremo	8
<b>BEEE</b>	Nº barras esquina extremo,extremo	9
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	1,2,1,3,1,4,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	.5,.5,1,2, 0,0,1,0, 0,1,1,0, 1,0,1,0, 1,1,1,0, 0,0,0,1, 0,1,0,1,1,0,0,1,1,1,0,1
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	1,3,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	1,2,1,4,1,6,1,7,1,8,1,9,6,7,6,8,7,9
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	1,4,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	1,2,1,3,1,6,1,7,1,8,1,9,6,7,6,8,8,9
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	1,3,1,6,1,7,1,8,1,9,6,7,6,8,8,9
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	1,4,1,6,1,7,1,8,1,9,6,7,6,8,7,9
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	1,4,1,6,1,7,1,8,1,9,6,7,6,8,7,9

NOTA: en el caso de los nudos que son el encuentro de cuatro barras horizontales, aunque pertenecen a la capa 2 física, el programa no les pondrá la carga que hemos dado para esta capa, sino que será cero, pues las cargas verticales no se podrían equilibrar y las horizontales es poco usual que se den.

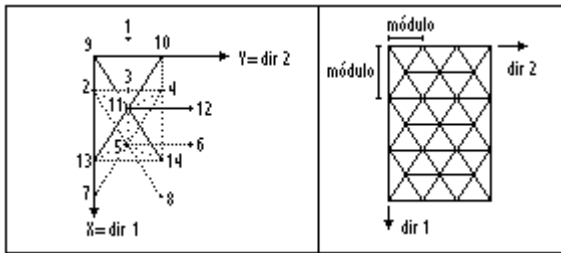
## ■ TIPO A7. PIRÁMIDES CUADRADAS, DIAGONAL INFERIOR.



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	1,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	10
<b>P</b>	Número de puntos	9
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	8
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	9
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	8
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	9
<b>BEOO</b>	Nº barras esquina origen,origen	7
<b>BEEO</b>	Nº barras esquina extremo,origen	8
<b>BEOE</b>	Nº barras esquina origen,extremo	8
<b>BEEE</b>	Nº barras esquina extremo,extremo	9
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	1,2,1,3,1,4,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	.5,.5,0,1, 0,0,0,0, 0,1,0,0, 1,0,0,0, 1,1,0,0, 0,0,1,2, 0,1,1,2, 1,0,1,2, 1,1,1,2
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	1,3,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	1,2,1,4,1,6,1,7,1,8,1,9,6,7,6,8,7,9
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	1,4,1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	1,2,1,3,1,6,1,7,1,8,1,9,6,7,6,8,8,9
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	1,5,1,6,1,7,1,8,1,9,6,7,6,8
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	1,3,1,6,1,7,1,8,1,9,6,7,6,8,8,9
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	1,4,1,6,1,7,1,8,1,9,6,7,6,8,7,9
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	1,4, 1,6,1,7,1,8,1,9,6,7,6,8,7,9

NOTA: en el caso de los nudos que son el encuentro de cuatro barras horizontales, aunque pertenecen a la capa 1 física, el programa no les pondrá la carga que hemos dado para esta capa, sino que será cero, pues las cargas verticales no se podrían equilibrar y las horizontales es poco usual que se den.

## ■ TIPO A8. TETRAEDROS CON BORDE EN CORNISA

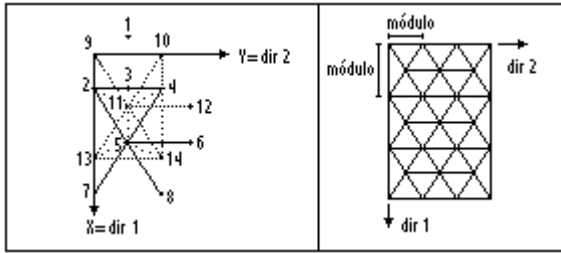


<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	2,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	18
<b>P</b>	Número de puntos	14
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	16
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	13
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	18
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	17
<b>BEOO</b>	Nº barras esquina origen,origen	19
<b>BEEO</b>	Nº barras esquina extremo,origen	16
<b>BEOE</b>	Nº barras esquina origen,extremo	15
<b>BEEE</b>	Nº barras esquina extremo,extremo	14
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	9,10,9,11,10,11,11,12,11,13,11,14,2,4,2,5,4,5,5,6,5,7,5,8,4,10,2,11, 11,4,11,5,5,13,5,14
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	-,334,5,0,1, .666,0,0,1, .666,5,0,1, .666,1,0,1, 1.666,5,0,1, 1.666,1,5,0,1, 2.666,0,0,1, 2.666,1,0,1, 0,0,1,2, 0,1,1,2, 1,5,1,2, 1,1,5,1,2, 2,0,1,2, 2,1,1,2
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13, 1,5,5,4,5,6,5,8, 4,10,11,4,11,5,5,13,5,14
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	9,10,9,11,10,11,10,14,11,13,11,14, 1,2,1,5,2,5, 2,11,11,5,5,13,5,14
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	9,10,9,11,10,11,11,12,11,13,11,14,2,4,2,5,4,5,5,6,5,7,5,8, 4,10,2,11,11,4,11,5,5,13,5,14
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	9,10,9,11,10,11,11,12,11,13,11,14,13,14,2,4,2,5,4,5,5,6,4,10,2,11,11,4,11,5,5,13,5,14
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13,3,4,3,5,4,5,5,6,5,8,4,10,3,11,11,4,11, 5,5,13,5,14,3,9
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13,13,14,5,1,5,4,5,6, 4,10, 11,4,11,5,5,13,5,14
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	9,10,9,11,10,11,10,14,11,13,11,14, 2,3,3,5,2,5, 10,3,3,11,11,2,11,5,5,13,5,14
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	9,10,9,11,10,11,10,14,11,13,11,14,13,14, 1,2,1,5,2,5, 2,11,11,5,5,13,5,14

NOTA: Aunque en la dirección 1 la longitud del módulo equivale a dos triángulos, la longitud del módulo que solicita el programa se refiere, en esta dirección, a la altura del triángulo; mientras que en la otra dirección lo que pide es el largo de la base.



## ■ TIPO A9. TETRAEDROS CON BORDE EN MANSARDA



<b>NCP</b>	Número de capas	2
<b>DI(2,3)</b>	Direcciones principales	2,0,0,0,1,0
<b>B</b>	Número de barras del módulo tipo	18
<b>P</b>	Número de puntos	14
<b>BA1OR</b>	Nº barras añadir, dir. 1, borde origen	16
<b>BA1EX</b>	Nº barras añadir, dir. 1, borde extremo	13
<b>BA2OR</b>	Nº barras añadir, dir. 2, borde origen	18
<b>BA2EX</b>	Nº barras añadir, dir. 2, borde extremo	17
<b>BEOO</b>	Nº barras esquina origen,origen	19
<b>BEEO</b>	Nº barras esquina extremo,origen	16
<b>BEOE</b>	Nº barras esquina origen,extremo	15
<b>BEEE</b>	Nº barras esquina extremo,extremo	14
<b>MB(B,2)</b>	Datos de las barras, módulo tipo	9,10,9,11,10,11,11,12,11,13,11,14,2,4,2,5,4,5,5,6,5,7,5,8,4,10,2,11, 11,4,11,5,5,13,5,14
<b>MP(P,4)</b>	Datos de las coordenadas de los nudos	- .334, .5,1,2, .666,0,1,2, .666, .5,1,2, .666,1,1,2, 1.666, .5,1,2, 1.666,1.5,1,2, 2.666,0,1,2, 2.666,1,1,2, 0,0,0,1, 0,1,0,1, 1, .5,0,1, 1,1.5,0,1, 2,0,0,1, 2,1,0,1
<b>BA1OR(BA1OR,2)</b>	Datos barras añadir, dir.1, borde origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13,1,5,5,4,5,6,5,8,4,10,11,4,11,5,5,13,5,14
<b>BA1EX(BA1EX,2)</b>	Datos barras añadir, dir.1, borde extremo	9,10,9,11,10,11,10,14,11,13,11,14,1,2,1,5,2,5,2,11,11,5,5,13,5,14
<b>BA2OR(BA2OR,2)</b>	Datos barras añadir, dir.2, borde origen	9,10,9,11,10,11,11,12,11,13,11,14,2,4,2,5,4,5,5,6,5,7,5,8,4,10,2,11,11,4,11,5,5,13,5,14
<b>BA2EX(BA2EX,2)</b>	Datos barras añadir, dir.2, borde extremo	9,10,9,11,10,11,11,12,11,13,11,14,13,14,2,4,2,5,4,5,5,6,4,10,2,11,11,4,11,5,5,13,5,14
<b>BEOO(BEOO,2)</b>	Datos barras esquina origen,origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13,3,4,3,5,4,5,5,6,5,8,4,10,3,11,11,4,11, 5,5,13,5,14,3,9
<b>BEEO(BEEO,2)</b>	Datos barras esquina extremo,origen	9,10,9,11,10,11,11,12,11,13,11,14,9,13,13,14,5,1,5,4,5,6, 4,10, 11,4,11,5,5,13,5,14
<b>BEOE(BEOE,2)</b>	Datos barras esquina origen,extremo	9,10,9,11,10,11,10,14,11,13,11,14, 2,3,3,5,2,5, 10,3,3,11,11,2,11,5,5,13,5,14
<b>BEEE(BEEE,2)</b>	Datos barras esquina extremo,extremo	9,10,9,11,10,11,10,14,11,13,11,14,13,14, 1,2,1,5,2,5, 2,11,11,5,5,13,5,14

NOTA: Aunque en la dirección 1 la longitud del módulo equivale a dos triángulos, la longitud del módulo que solicita el programa se refiere, en esta dirección, a la altura del triángulo; mientras que en la otra dirección lo que pide es el largo de la base.

## 7.3 LISTADOS DEL PROGRAMA.

### 7.3.1 MÓDULO PRINCIPAL DE CÁLCULO DINÁMICO.



```

O
O '::::::::::::::::::::::::::::: Dinami.FRM:::::::::::::::::::::::::::::
O ' version xx07.frm adaptada a las subrutinas DiscoX de Vb      25-06-01
O ' versión sin giro de matrices.....fecha finalización versión 22-07-01
O '
O 'Programa con Runge-Kutta en las cuatro primeras iteraciones y Adams-Moulton en las
O   siguientes
O 'Considera el axil de las barras paralelo a la barra sin flexionar.
O '
O 'Versión que intercala o no (nnd$), un nudo intermedio durante el cálculo,
O 'Los nudos añadidos (intermedios) se comportan como barras
O 'Los nudos extremos se comportan como puntos materiales
O '.....
O '
O ' alon - longitud en reposo de las barras
O ' qst - carga estática en nudo i con dirección x,y,x en Kgf
O ' dtqd - intervalo de tiempo de la tabla de la carga dinámica
O ' nc - nº de puntos de la tabla de la carga dinámica
O ' tipoq$ - tipo de carga dinámica
O ' nq - tipo de fuerza dinámica
O ' amas - masa de los nudos
O ' nll - nudos que delimitan las barras y tipo de barra
O ' nx - coacciones
O '====
O ' anx - coacciones inferiores
O ' bnx - coacciones superiores
O ' kc - constante de elasticidad de muelle de las coacciones
O
O Option Explicit
O DefDbl A-H
O DefLng I-N
O DefDbl O-Z
O
O 'doble precisión
O Dim CA(), va(), anx(), bnx()
O Dim p(), alon(), slon1(), amas(), qst(), ainer(), C(), v(), tabqd1(), tabqd2(),
O   tabqd3(), tabad1(), tabad2(), tabad3()
O Dim f0(-3 To 6), f1(-3 To 6), t0(-3 To 6), t1(-3 To 6)
O Dim af0(-3 To 6), af1(-3 To 6), at0(-3 To 6), at1(-3 To 6)
O Dim dt, alambda, velmax, vCritica, G, pi, kc As Double
O Dim tabym(), C0()
O 'enteros 'np, ne '<--'variables globales
O Dim np2, ne2, ndim, ntp, npqd(3), itergrb, npad(3), npym
O Dim nq(), nll(), nx(), NA()
O Dim itermax, nParar
O 'cadenas
O Dim pp$( ), ppropio$, nnd$, axil$, cont$, xpos$, xvel$, xaxi$, xcor$, xfle$
O Dim extension2$, amortt$, tipotit$, comentario$
O 'Dim extension$ , nom$

```

```

O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O Private Sub Form_Activate()
O CalculoDinamico
O End Sub
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O Private Sub cmdCancel_Click()
O nParar = 3
O End Sub
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O Sub CalculoDinamico()
O Call Lectura_Datos
O Call Nucleo1
O ReDim CA(0), va(0), anx(0), bnx(0), p(0), alon(0), slon1(0)
O ReDim amas(0), qst(0), ainer(0), C(0), v(0)
O ReDim tabqd1(0), tabqd2(0), tabqd3(0), tabad1(0), tabad2(0), tabad3(0)
O ReDim tabym(0), C0(0)
O ReDim nq(0), nll(0), nx(0), NA(0)
O ReDim pp$(0)
O End Sub
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O Sub Lectura_Datos()
O
O 'Dobles
O Dim alfa, beta, coalfa, sialfa, cobeta, sibeta, vu, acos1
O Dim x1, x2, y1, y2, z1, z2, al, ale, amasabar, xp, yp, zp
O Dim qq, CfReparto, CfRepartoPr
O 'enteros ' nea ->'en declarac.globales
O Dim inic, nfina, ntipo, medio, i, j, k, n1, n2, npa ', nea ', mPos
O 'cadenas
O Dim a$, ab$, tipoq$(3), tipoym$, tipoa$(3)
O
O vCritica = 0.0001 ' Velocidad que en amort. por rozam. dinám.pasa a estático
O pi = 4 * Atn(1)
O G = 9.80665
O
O 'Lectura de los datos generales.....
O 'Lee np,ne,nea...
O DiscoX "datos.dat", "L", "-", iii, sss, ddd, xxx, ttt$
O ' np: n° de nodos extremos
O ' ne: n° de barras
O ' nea: n° de barras añadidas (barras enlace)
O
O 'Lectura de los tipos de barras.....
O DiscoX "TIPOS.DAT", "L", "DX", iii, sss, p(), pp$(0), ttt$
O 'p(ntp, *) Prop.barras (área cm2,m.iner cm4,mod.Young Kg/cm2,t.adm kg/cm2,p.esp.kg/m3)
O 'pp$(ntp) Nombres descriptivos de los tipos de barras
O
O 'Datos generales de este cálculo dinámico...
O ab$ = Trim(nom$) + "\" + "Parametr." + extension$
O Open ab$ For Input As #2
O Input #2, a$ ' Barra de títulos
O Input #2, dt, itergrb, amortt$, alambda, itermax, velmax, ppropio$, nnd$, extension2$
O Input #2, cont$, axil$, comentario$, xpos$, xvel$, xaxi$, xcor$, xfle$
O Close #2
O
O If (nnd$ = "Si") Then 'preparamos para añadir un nodo intermedio en cada barra...
O npa = ne ' npa: n° de nodos añadidos (medios de barras)
O ndim = 5
O Else
O npa = 0
O ndim = 3
O End If
O np2 = np + npa ' np2: n° de nodos totales
O ne2 = ne + nea ' ne2: n° de barras totales
O
O ReDim nll(ne2, 3) 'Datos de las barras (nodo or., fin, tipo)
O ReDim alon(ne2) 'Longitudes iniciales de las barras en m.

```

```

○ ReDim slon1(ne2)          'Semilongitud barra
○ ReDim amas(np2)          'Masa de los nudos
○ ReDim C(np2, 3)          'Coordenadas de los nodos, en m.
○ ReDim CA(np2, 2)         'Orientación de las barras nudos medios 1-alfa,2-beta
○ ReDim C0(np2, 3)         'Velocidad de cada nodo en m/s
○ ReDim v(np2, 3)          'Velocidad angular nudos medios en m/s
○ ReDim va(np2, 2)         'Coacciones en las posiciones de los nodos.
○ ReDim nx(np2, 3)         'Coacciones inferiores en las posiciones de los nodos.
○ ReDim anx(np2, 3)        'Coacciones superiores en las posiciones de los nodos.
○ ReDim bnx(np2, 3)        'Aceleraciones de las cargas
○ ReDim NA(np2, 3)         'Cargas estáticas en nudos, en Kg.
○ ReDim nq(np2, 3)         'Momento de Inercia longitudinal del nudo intermedio
○ ReDim qst(np2, 3)
○ ReDim ainer(np2)
○
○ '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
○ '          Lectura de valores iniciales
○ '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
○
○ ' LONGITUDES BARRAS En equilibrio inicial sin tensión.....
○ DiscoX "LONGITUD.DAT", "LNOREDIM", "D", iii, sss, alon(), xxx, ttt$
○
○ 'Lectura de los datos enteros de las barras
○ DiscoX "BARRAS.DAT", "LNOREDIM", "L", nll(), sss, ddd, xxx, ttt$
○
○ If (nea > 0) Then
○   'DiscoX "BARRASA.DAT", "LNOREDIM", "L", nll(), sss, ddd, xxx, ttt$
○   ab$ = Trim(nom$) + "\" + "Barrasa.Dat"
○   Open ab$ For Input As #1
○     For k = ne + 1 To ne2
○       For j = 1 To 2
○         Input #1, i: nll(k, j) = i + np 'añadiremos los nudos al final
○       Next j
○     Next k
○   Close #1
○ End If
○
○ 'Lectura de las cargas estáticas externas de los puntos.....
○ ab$ = Trim(nom$) + "\" + "Cargas.Dat"
○ Open ab$ For Input As #1
○   For i = 1 To np
○     For j = 1 To 3
○       Input #1, qq: qst(i, j) = qq * 1000 'Paso de Tn a Kg
○     Next j
○   Next i
○ Close #1
○
○ 'Lectura de las coacciones y aceleraciones exteriores de los nudos .....
○ ab$ = Trim(nom$) + "\" + "Coaccion.Dat"
○ Open ab$ For Input As #1
○ For i = 1 To np
○   For j = 1 To 3
○     Input #1, nx(i, j)
○   Next j
○ Next i
○ Close #1
○
○ ab$ = Trim(nom$) + "\" + "Coacclmt.Dat"
○ Open ab$ For Input As #1
○ For i = 1 To np
○   For j = 1 To 3
○     Input #1, anx(i, j), bnx(i, j)
○   Next j
○ Next i
○ Close #1
○
○ ' los nudos intermedios no se coaccionan
○ If (ndim = 5) Then
○   For i = np + 1 To np2
○     For j = 1 To 3
○       nx(i, j) = 1234
○     Next j
○   Next i
○ End If
○
○ 'lectura de las aceleraciones dinámicas
○ ab$ = Trim(nom$) + "\" + "Aceled.Dat"
○ Open ab$ For Input As #1
○ For i = 1 To np

```

```

O   For j = 1 To 3
O     Input #1, NA(i, j)
O   Next j
O Next i
O Close #1
O
O ' LOCALIZACIÓN DE LAS CARGAS DINÁMICAS
O ab$ = Trim(nom$) + "\" + "Cargasd.Dat"
O Open ab$ For Input As #1
O For i = 1 To np
O   For j = 1 To 3
O     Input #1, nq(i, j)
O   Next j
O Next i
O Close #1
O
O 'Lectura de las cargas y aceleraciones DINAMICAS de los puntos.....
O
O 'Para la carga i (-3 a 6):
O 'Prqd(i) - fichero con las propiedades Generales de las cargas dinámicas
O 'npqd(i) - n° de puntos de la tabla de la carga dinámica
O 'tipoq$(i) - tipo de tabla de carga dinámica
O 'tabqdi() - tablas de cargas dinámicas
O
O 'Datos de las CARGAS DINÁMICAS en f0 y f1 (Kg ) y t0 y t1 (seg).....
O ab$ = nom$ + "\" + "Prqd.Dat"
O Open ab$ For Input As #1
O For i = -3 To 6
O   Input #1, f0(i), f1(i), t0(i), t1(i) ' fuerza en Kgf, tiempo en seg
O Next i
O Close #1
O
O 'Tabla 1
O ab$ = Trim(nom$) + "\" + "Cargd1.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(1), tipoq$(1)
O ReDim tabqdi(npqd(1))
O For i = 1 To npqd(1)
O   Input #1, tabqdi(i)
O Next i
O Close #1
O
O 'Tabla 2
O ab$ = Trim(nom$) + "\" + "Cargd2.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(2), tipoq$(2)
O ReDim tabqdi(npqd(2))
O For i = 1 To npqd(2)
O   Input #1, tabqdi(i)
O Next i
O Close #1
O
O 'Tabla 3
O ab$ = Trim(nom$) + "\" + "Cargd3.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(3), tipoq$(3)
O ReDim tabqdi(npqd(3))
O For i = 1 To npqd(3)
O   Input #1, tabqdi(i)
O Next i
O Close #1
O
O 'Datos de las Aceleraciones en m/s*s f0 y f1 y t0 y t1 seg.....
O ab$ = Trim(nom$) + "\" + "Arqd.Dat"
O Open ab$ For Input As #1
O For i = -3 To 6
O   Input #1, af0(i), af1(i), at0(i), at1(i) ' fuerza en Kgf, Kgf, tiempo en seg, seg
O Next i
O Close #1
O
O 'Tabla 1
O ab$ = Trim(nom$) + "\" + "Ad1.Dat"
O Open ab$ For Input As #1
O Input #1, npad(1), tipoa$(1)
O ReDim tabad1(npad(1))
O For i = 1 To npad(1)
O   Input #1, tabad1(i)
O Next i
O Close #1

```

```

○
○
○ 'Tabla 2
○ ab$ = Trim(nom$) + "\" + "Ad2.Dat"
○ Open ab$ For Input As #1
○ Input #1, npad(2), tipoa$(2)
○ ReDim tabad2(npad(2))
○ For i = 1 To npad(2)
○   Input #1, tabad2(i)
○ Next i
○ Close #1
○
○ 'Tabla 3
○ ab$ = Trim(nom$) + "\" + "Ad3.Dat"
○ Open ab$ For Input As #1
○ Input #1, npad(3), tipoa$(3)
○ ReDim tabad3(npad(3))
○ For i = 1 To npad(3)
○   Input #1, tabad3(i)
○ Next i
○ Close #1
○
○ '.....
○ 'Inicialización de variables de posición, velocidad y aceleración
○ DiscoX "Coordena.Dat", "LNOREDIM", "D3A", iii, sss, C(), xxx, ttt$
○
○ If (mnd$ = "Si") Then 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX3nudos
○   'Coordenadas del punto intermedio. Los nuevos ne
○   'puntos se introducen después de los np existentes
○   ab$ = Trim(nom$) + "\" + "Coordenl.Dat"
○   Open ab$ For Input As #1
○   For i = 1 To ne
○     For j = 1 To 3
○       Input #1, C(np + i, j)
○     Next j
○   Next i
○   Close #1
○   DiscoX "Slongit.Dat", "LNOREDIM", "D", iii, sss, slonl(), xxx, ttt$
○
○   If (cont$ = "No") Then 'si es CÁLCULO NUEVO.....
○     MaCero v: MaCero va
○     End If
○
○   If (cont$ = "Si") Then 'si es CÁLCULO CONTINUACION.....
○     ab$ = Trim(nom$) + "\" + "contin." + extension2$
○     Open ab$ For Binary As #1
○     For j = 1 To 3
○       For i = 1 To np2
○         Get #1, , C(i, j) 'Lee despl. y velocidades
○         Get #1, , v(i, j) 'de los nodos
○       Next i
○     Next j
○     For j = 1 To 2
○       For i = 1 To ne
○         Get #1, , CA(i, j) 'Lee despl. y velocidades
○         Get #1, , va(i, j) 'angulares los nodos medios
○       Next i
○     Next j
○     Close #1
○   End If
○   mIguala C0, C
○
○
○
○ 'Calcula los ángulos iniciales de los vectores Extremo-->Inicio ===
○ For k = 1 To ne
○   medio = np + k 'nudo intermedio
○   inic = nll(k, 1)
○   nfina = nll(k, 2)
○   ntipo = nll(k, 3) 'tipo barra
○
○   If (cont$ = "No") Then 'si es CÁLCULO NUEVO.....
○     x1 = C(inic, 1)
○     y1 = C(inic, 2)
○     z1 = C(inic, 3)
○     x2 = C(nfina, 1)
○     y2 = C(nfina, 2)
○     z2 = C(nfina, 3)
○     vu = Sqr((x1 - x2) * (x1 - x2) + (y1 - y2) *
○     (y1 - y2) + (z1 - z2) * (z1 - z2)) 'longitud barra
○     acos1 = (z1 - z2) / vu 'cos con Z

```

```

O      CA(k, 1) = ACos(acos1)          'ang. con eje Z '
O      'CA(k, 1) = ACos(acos1) + 0.02    'para validacion modos 2 transversales....
O
O      End If      '.....'
O
O      'Pone la masa de las barras en los nudos
O      al = alon(k)
O      If (pp$(ntipo) = "Sin.definir") Then Stop
O      amasabar = p(ntipo, 1) * al * p(ntipo, 5) / 10000# ' en Kg-m....
O      CfReparto = 0.55
O      CfRepartoPr = 0.41
O      amas(inic) = (1 - CfReparto) * slon1(k) * amasabar / al          'Distrib. masas 1
O      amas(medio) = CfReparto * amasabar          'Distrib. masas
O      amas(nfina) = (1 - CfReparto) * (al - slon1(k)) * amasabar / al 'Distrib. masas 2
O      'Pone el momento de inercia longitudinal
O      ainer(medio) = amas(medio) * al * al * CfRepartoPr * CfRepartoPr / 12
O      Next k
O
O      Else 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX 2nudos
O      If (cont$ = "No") Then 'CALCULO NUEVO
O      MaCero v
O      Else
O      ab$ = Trim(nom$) + "\" + "contin." + extension2$
O      Open ab$ For Binary As #1
O      For j = 1 To 3
O      For i = 1 To np2
O      Get #1, , C(i, j)
O      Get #1, , v(i, j) ' desplazamientos y velocidades..
O      Next i
O      Next j
O      Close #1
O      End If
O
O      For k = 1 To ne
O      'Pone la masa de las barras en los nudos.....
O      ntipo = nll(k, 3)
O      al = alon(k)
O      n1 = nll(k, 1)          ' nudo inicial
O      n2 = nll(k, 2)          ' nudo final
O      If (pp$(ntipo) = "Sin.definir") Then Stop
O      amasabar = p(ntipo, 1) * al * p(ntipo, 5) / 10000#
O      amas(n1) = 0.5 * amasabar ' en Kg...
O      amas(n2) = 0.5 * amasabar
O      Next k
O
O      End If 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXfin263nudos
O
O      'Consideración del peso propio.....
O      If (pppropio$ = "Si") Then
O      For i = 1 To np2
O      qst(i, 3) = qst(i, 3) - amas(i) ' Kgf
O      Next i
O      End If
O
O      End Sub
O
O      '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O      '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O      Sub Nucleo1()
O
O      Dim cpa(), vpa(), cka(), vka()
O      Dim AKaa(), AKba() 'nodos centrales
O      Dim AKa(), AKb()
O      Dim apa(), aca()
O      Dim ap(), ac()
O      Dim cp(), vp(), ck(), vk(), fueraxil(), fuercort(), flec()
O      'Doubles
O      ReDim cpa(ne, 2, 4), vpa(ne, 2, 4), cka(ne, 2), vka(ne, 2)
O      ReDim AKaa(ne, 2, 4), AKba(ne, 2, 4) 'nodos centrales
O      ReDim AKa(np2, 3, 4), AKb(np2, 3, 4)
O      ReDim apa(ne, 2, 4), aca(ne, 2)
O      ReDim ap(np2, 3, 4), ac(np2, 3)
O      ReDim cp(np2, 3, 4), vp(np2, 3, 4), ck(np2, 3), vk(np2, 3), fueraxil(ne, 2),
O      fuercort(ne, 2), flec(ne, 2)
O      Dim tiempo, desp, vabs, xa
O      Dim ul, AUXaa, AUXba, AUXa, AUXb
O      Dim desplaz

```

```

O 'enteros largos
O Dim i, j, k, ncim1, nCi, ncil, nci2, nci3
O Dim itete, litergrb, iterac
O 'cadenas
O Dim a$, b$, d$, e$, f$, ab$
O
O 'comienzo de las iteraciones.....
O iterac = 1 ' n° de la iteración actual.
O litergrb = itergrb
O 'GOSUB 1500 ! Actualiza pretensado de las barras....
O a$ = Trim(nom$) + "\" + "resultc." + extension$
O b$ = Trim(nom$) + "\" + "resultv." + extension$
O d$ = Trim(nom$) + "\" + "axil." + extension$
O e$ = Trim(nom$) + "\" + "cort." + extension$
O f$ = Trim(nom$) + "\" + "flec." + extension$
O If (xpos$ = "Si") Then Open a$ For Binary As #1
O If (xvel$ = "Si") Then Open b$ For Binary As #2
O
O Lbitermax.Caption = itermax
O
O nParar = 0
O MaCero AKa
O MaCero vk: MaCero ck: MaCero vka: MaCero cka: MaCero ac: MaCero aca
O MaCero cp: MaCero cpa: MaCero vp: MaCero vpa: MaCero ap: MaCero apa
O
O If (ndim = 3) Then
O   If (xaxi$ = "Si") Then Open d$ For Binary As #9
O Else
O   If (xaxi$ = "Si") Then Open d$ For Binary As #9
O   If (xcor$ = "Si") Then Open e$ For Binary As #7
O   If (xfile$ = "Si") Then Open f$ For Binary As #8
O End If
O
O desplaz = 0 'desplazamiento máximo de cada iteración
O
O #####
O #####RUNGE-KUTTA#####
O #####
O
O Do While (nParar = 0)
O
O ' Rellenamos matrices a usar en Adamas-Moulton
O If (ndim = 3) Then
O   For i = 1 To np
O     For j = 1 To 3
O       cp(i, j, iterac) = C(i, j)
O       vp(i, j, iterac) = v(i, j)
O       ap(i, j, iterac) = ac(i, j)
O     Next j
O   Next i
O Else
O   For i = 1 To np2
O     For j = 1 To 3
O       cp(i, j, iterac) = C(i, j)
O       vp(i, j, iterac) = v(i, j)
O       ap(i, j, iterac) = ac(i, j)
O     Next j
O   Next i
O   For i = 1 To ne
O     For j = 1 To 2
O       cpa(i, j, iterac) = CA(i, j)
O       vpa(i, j, iterac) = va(i, j)
O       apa(i, j, iterac) = aca(i, j)
O     Next j
O   Next i
O End If
O
O 'CALCULO DE K1,K2,K3,K4.....
O For k = 1 To 4
O   Select Case k
O     Case 1: ul = 0
O     Case 2: ul = 0.5
O     Case 3: ul = 0.5
O     Case 4: ul = 1
O   End Select
O   tiempo = dt * iterac + ul * dt
O
O
O
O If (ndim = 3) Then

```



```

O      For j = 1 To 3
O      For i = 1 To np2
O      If k > 1 Then AUXa = ul * AKa(i, j, k - 1): AUXb = ul * AKb(i, j, k - 1)
O      Else AUXa = 0: AUXb = 0
O      ck(i, j) = C(i, j) + AUXa
O      vk(i, j) = v(i, j) + AUXb
O      Next i
O      Next j
O      Call A2n(ck, vk, ac, tiempo, fueraxil) 'calcula acel. sin nudo intermedio
O      For j = 1 To 3
O      For i = 1 To np2
O      AKa(i, j, k) = vk(i, j) * dt
O      AKb(i, j, k) = ac(i, j) * dt
O      Next i
O      Next j
O      Else
O      ' Nudo intermedio
O      For i = 1 To np2
O      For j = 1 To 3
O      If k > 1 Then AUXa = ul * AKa(i, j, k - 1): AUXb = ul * AKb(i, j, k - 1)
O      Else AUXa = 0: AUXb = 0
O      ck(i, j) = C(i, j) + AUXa
O      vk(i, j) = v(i, j) + AUXb
O      Next j
O      Next i
O      For i = 1 To ne
O      For j = 1 To 2
O      If k > 1 Then AUXba = ul * AKba(i, j, k - 1) Else AUXaa = 0: AUXba = 0
O      cka(i, j) = CA(i, j) + AUXaa
O      vka(i, j) = va(i, j) + AUXba
O      Next j
O      Next i
O      Call A3n(ck, cka, vk, vka, ac, aca, tiempo, fueraxil, fuercort, flec) 'n.inter
O      For j = 1 To 3
O      For i = 1 To np2
O      AKa(i, j, k) = vk(i, j) * dt
O      AKb(i, j, k) = ac(i, j) * dt
O      Next i
O      Next j
O      For j = 1 To 2
O      For i = 1 To ne
O      AKaa(i, j, k) = vka(i, j) * dt
O      AKba(i, j, k) = aca(i, j) * dt
O      Next i
O      Next j
O      End If
O      Next k
O
O
O      'CALCULO DE LOS DESPLAZAMIENTOS por Runge-Kutta.....
O      If (ndim = 3) Then 'sin nudo intermedio
O      For j = 1 To 3
O      For i = 1 To np2
O      desp = (AKa(i, j, 1) + 2 * AKa(i, j, 2) + 2 * AKa(i, j, 3) + AKa(i, j, 4)) / 6
O      C(i, j) = C(i, j) + desp
O      v(i, j) = v(i, j) + (AKb(i, j, 1) + 2 * AKb(i, j, 2) + 2 * AKb(i, j, 3) + AKb(i,
O      j, 4)) / 6
O      vabs = Abs(v(i, j))
O      If (vabs > velmax) Then Call Avisovel(i): Stop ' Límite de velocidad..
O      If (Abs(desp) > Abs(desplaz)) Then desplaz = desp 'despl. máx. en esta iteración
O      Next i
O      Next j
O
O      For i = 1 To np 'nudos coaccionados (libres o cero)....
O      If nx(i, 1) <> 1234 Then v(i, 1) = 0
O      If nx(i, 2) <> 1234 Then v(i, 2) = 0
O      If nx(i, 3) <> 1234 Then v(i, 3) = 0
O      Next i
O
O
O      'Grabación de c,v.....
O      If ((iterac Mod litergrb) = 0) Then
O      For j = 1 To 3
O      For i = 1 To np2
O      If (xpos$ = "Si") Then Put #1, , C(i, j) ' Graba desplaz. y velocidades..
O      If (xvel$ = "Si") Then Put #2, , v(i, j)
O      Next i
O      Next j

```

```

O For k = 1 To ne
O   xa = fueraxil(k, 1) / G
O   If (xaxi$ = "Si") Then Put #9, , xa
O Next k
O 'barra de progreso.....
O BarraTiempo Progreso, (iterac), (itermax)
O LbDesplaz.Caption = Str$(desplaz)
O Lbiterac.Caption = iterac
O End If
O Else 'Nudo intermedio
O For j = 1 To 3
O   For i = 1 To np2
O     desp = (AKa(i, j, 1) + 2 * AKa(i, j, 2) + 2 * AKa(i, j, 3) + AKa(i, j, 4)) / 6
O     C(i, j) = C(i, j) + desp
O     v(i, j) = v(i, j) + (AKb(i, j, 1) + 2 * AKb(i, j, 2) + 2 * AKb(i, j, 3) + AKb(i,
O       j, 4)) / 6
O     vabs = Abs(v(i, j))
O     If (vabs > velmax) Then Call Avisovel(i): Stop ' Límite de velocidad...
O     If (Abs(desp) > Abs(desplaz)) Then desplaz = desp
O   Next i
O Next j
O For j = 1 To 2
O   For i = 1 To ne 'nudos intermedios
O     CA(i, j) = CA(i, j) + (AKaa(i, j, 1) + 2 * AKaa(i, j, 2) + 2 * AKaa(i, j, 3) +
O       AKaa(i, j, 4)) / 6
O     va(i, j) = va(i, j) + (AKba(i, j, 1) + 2 * AKba(i, j, 2) + 2 * AKba(i, j, 3) +
O       AKba(i, j, 4)) / 6
O   Next i
O Next j
O
O For i = 1 To np 'coacciones...
O   For j = 1 To 3
O     If nx(i, j) <> 1234 Then C(i, j) = C0(i, j): v(i, j) = 0
O   Next j
O Next i
O
O Call A3n(C, CA, v, va, ac, aca, tiempo, fueraxil, fuercort, flec)
O
O 'grabación de posiciones c() y velocidades v() resultantes...
O If ((iterac Mod litergrb) = 0) Then
O
O   For j = 1 To 3
O     For i = 1 To np2
O       If (xpos$ = "Si") Then Put #1, , C(i, j) ' Graba desplaz. y velocidades..
O       If (xvel$ = "Si") Then Put #2, , v(i, j)
O     Next i
O   Next j
O
O   For j = 1 To 2
O     For i = 1 To ne
O       If (xpos$ = "Si") Then Put #1, , CA(i, j)
O       If (xvel$ = "Si") Then Put #2, , va(i, j)
O     Next i
O   Next j
O
O   For k = 1 To ne
O     For j = 1 To 2
O       If (xaxi$ = "Si") Then xa = fueraxil(k, j) / G: Put #9, , xa
O       If (xcor$ = "Si") Then xa = fuercort(k, j) / G: Put #7, , xa
O       If (xfle$ = "Si") Then: xa = flec(k, j) / G: Put #8, , xa
O     Next j
O   Next k
O
O   'barra de progreso.....
O   BarraTiempo Progreso, (iterac), (itermax)
O   LbDesplaz.Caption = Str$(desplaz)
O   Lbiterac.Caption = iterac
O End If
O
O End If
O
O If (iterac = 4) Then nParar = 1 ' para arrancar con Adams-Moulton
O If (iterac = itermax) Then nParar = 2 ' Finaliza si pasa el n°m x. de iteraciones
O
O iterac = iterac + 1!
O Loop
O
O ReDim AKa(0), AKb(0), AKaa(0), AKba(0)
O

```

```

O '#####Adams-Moulton#####
O
O If nParar = 1 Then nParar = 0 'si viene de Runge-Kutta
O Do While (nParar = 0)
O
O tiempo = dt * iterac
O itete = iterac - 2!
O nci1 = ((itete + 1) Mod 4) + 1
O nCi = (itete Mod 4) + 1
O nci1 = ((itete - 1) Mod 4) + 1
O nci2 = ((itete - 2) Mod 4) + 1
O nci3 = nci1
O
O 'PREDICCION.....
O If (ndim = 3) Then 'Dos nudos
O   For j = 1 To 3
O     For i = 1 To np2
O       ck(i, j) = cp(i, j, nCi) + dt * (55 * vp(i, j, nCi) - 59 * vp(i, j, nci1) + 37 *
O         vp(i, j, nci2) - 9 * vp(i, j, nci3)) / 24#
O       vk(i, j) = vp(i, j, nCi) + dt * (55 * ap(i, j, nCi) - 59 * ap(i, j, nci1) + 37 *
O         ap(i, j, nci2) - 9 * ap(i, j, nci3)) / 24#
O     Next i
O   Next j
O   For i = 1 To np°
O     For j = 1 To 3
O       If nx(i, j) <> 1234 Then vk(i, j) = 0
O     Next j
O   Next i
O   Call A2n(ck, vk, ac, tiempo, fueraxil) 'calcula aceleración...
O
O Else 'Nudo intermedio
O   For j = 1 To 3
O     For i = 1 To np2
O       ck(i, j) = cp(i, j, nCi) + dt * (55 * vp(i, j, nCi) - 59 * vp(i, j, nci1) + 37 *
O         vp(i, j, nci2) - 9 * vp(i, j, nci3)) / 24#
O       vk(i, j) = vp(i, j, nCi) + dt * (55 * ap(i, j, nCi) - 59 * ap(i, j, nci1) + 37 *
O         ap(i, j, nci2) - 9 * ap(i, j, nci3)) / 24#
O     Next i
O   Next j
O   For i = 1 To np
O     For j = 1 To 3
O       If nx(i, j) <> 1234 Then ck(i, j) = C0(i, j): vk(i, j) = 0
O     Next j
O   Next i
O
O   For j = 1 To 2
O     For i = 1 To ne
O       cka(i, j) = cpa(i, j, nCi) + dt * (55 * vpa(i, j, nCi) - 59 * vpa(i, j, nci1) +
O         37 * vpa(i, j, nci2) - 9 * vpa(i, j, nci3)) / 24#
O       vka(i, j) = vpa(i, j, nCi) + dt * (55 * apa(i, j, nCi) - 59 * apa(i, j, nci1) +
O         37 * apa(i, j, nci2) - 9 * apa(i, j, nci3)) / 24#
O     Next i
O   Next j
O
O End If
O
O 'CORRECCION.....
O If (ndim = 3) Then 'Dos nudos...
O   For j = 1 To 3
O     For i = 1 To np2
O       v(i, j) = vp(i, j, nCi) + dt * (9 * ac(i, j) + 19 * ap(i, j, nCi) - 5 * ap(i, j,
O         nci1) + ap(i, j, nci2)) / 24
O       vabs = Abs(v(i, j))
O       If (vabs > velmax) Then Call Avisovel(i): Stop 'Límite de velocidad...
O       desp = dt * (9 * v(i, j) + 19 * vp(i, j, nCi) - 5 * vp(i, j, nci1) + vp(i, j,
O         nci2)) / 24
O       C(i, j) = cp(i, j, nCi) + desp
O       If (Abs(desp) > Abs(desplaz)) Then desplaz = desp
O     Next i
O   Next j
O
O   For i = 1 To np2
O     For j = 1 To 3
O       If nx(i, j) <> 1234 Then v(i, j) = 0
O     Next j
O   Next i
O
O   Call A2n(C, v, ac, tiempo, fueraxil) 'calcula aceleración

```

```

O   For i = 1 To np2
O     For j = 1 To 3
O       cp(i, j, nci1) = C(i, j)
O       vp(i, j, nci1) = v(i, j)
O       ap(i, j, nci1) = ac(i, j)
O     Next j
O   Next i
O
O   Else 'Modelo con nudos intermedios
O     For j = 1 To 3
O       For i = 1 To np2
O         v(i, j) = vp(i, j, nCi) + dt * (9 * ac(i, j) + 19 * ap(i, j, nCi) - 5 * ap(i, j,
O           nci1) + ap(i, j, nci2)) / 24
O         vabs = Abs(v(i, j))
O         If (vabs > velmax) Then Call Avisovel(i): Stop ' Límite de velocidad...
O         desp = dt * (9 * v(i, j) + 19 * vp(i, j, nCi) - 5 * vp(i, j, nci1) + vp(i, j,
O           nci2)) / 24
O         C(i, j) = cp(i, j, nCi) + desp
O         If (Abs(desp) > Abs(desplaz)) Then desplaz = desp
O       Next i
O     Next j
O
O   For i = 1 To np
O     For j = 1 To 3
O       If nx(i, j) <> 1234 Then C(i, j) = C0(i, j): v(i, j) = 0
O     Next j
O   Next i
O
O   For i = 1 To ne
O     For j = 1 To 2
O       va(i, j) = vpa(i, j, nCi) + dt * (9 * aca(i, j) + 19 * apa(i, j, nCi) - 5 *
O         apa(i, j, nci1) + apa(i, j, nci2)) / 24
O       CA(i, j) = cpa(i, j, nCi) + dt * (9 * va(i, j) + 19 * vpa(i, j, nCi) - 5 *
O         vpa(i, j, nci1) + vpa(i, j, nci2)) / 24
O       cpa(i, j, nci1) = CA(i, j)
O       vpa(i, j, nci1) = va(i, j)
O     Next j
O     If (CA(i, 2) > 2# * pi) Then
O       cpa(i, 2, nci1) = cpa(i, 2, nci1) - 2# * pi
O       cpa(i, 2, nCi) = cpa(i, 2, nCi) - 2# * pi
O       cpa(i, 2, nci1) = cpa(i, 2, nci1) - 2# * pi
O       cpa(i, 2, nci2) = cpa(i, 2, nci2) - 2# * pi
O     End If
O     If (CA(i, 2) < 0#) Then
O       cpa(i, 2, nci1) = cpa(i, 2, nci1) + 2# * pi
O       cpa(i, 2, nCi) = cpa(i, 2, nCi) + 2# * pi
O       cpa(i, 2, nci1) = cpa(i, 2, nci1) + 2# * pi
O       cpa(i, 2, nci2) = cpa(i, 2, nci2) + 2# * pi
O     End If
O   Next i
O   Call A3n(C, CA, v, va, ac, aca, tiempo, fueraxil, fuercort, flec) 'aceleración
O   For i = 1 To np2
O     For j = 1 To 3
O       cp(i, j, nci1) = C(i, j)
O       vp(i, j, nci1) = v(i, j)
O       ap(i, j, nci1) = ac(i, j)
O     Next j
O   Next i
O   For i = 1 To ne
O     For j = 1 To 2
O       apa(i, j, nci1) = aca(i, j)
O     Next j
O   Next i
O End If
O
O 'ARCHIVO c,v.....
O
O If (ndim = 3) Then
O
O   If ((iterac Mod litergrb) = 0!) Then
O     For j = 1 To 3
O       For i = 1 To np2
O         If (xpos$ = "Si") Then Put #1, , C(i, j)      'Graba despl. y velocidades..
O         If (xvel$ = "Si") Then Put #2, , v(i, j)
O       Next i
O     Next j
O
O     For k = 1 To ne
O       If (xaxi$ = "Si") Then xa = fueraxil(k, 1) / G: Put #9, , xa

```

```

O     Next k
O     End If
O     'barra proceso...
O     BarraTiempo Progreso, (iterac), (itermax)
O     LbDesplaz.Caption = Str$(desplaz)
O     Lbiterac.Caption = iterac
O     Else
O
O     'nudo intermedio
O     If ((iterac Mod litergrb) = 0) Then
O
O     For j = 1 To 3
O     For i = 1 To np2
O     If (xpos$ = "Si") Then Put #1, , C(i, j) 'Graba despl. y velocidades..
O     If (xvel$ = "Si") Then Put #2, , v(i, j)
O     Next i
O     Next j
O
O     For j = 1 To 2
O     For i = 1 To ne
O     If (xpos$ = "Si") Then Put #1, , CA(i, j) 'Graba despl. y velocidades..
O     If (xvel$ = "Si") Then Put #2, , va(i, j)
O     Next i
O     Next j
O
O     For k = 1 To ne
O     For j = 1 To 2
O     If (xaxi$ = "Si") Then xa = fueraxil(k, j) / G: Put #9, , xa
O     If (xcor$ = "Si") Then xa = fuercort(k, j) / G: Put #7, , xa
O     If (xfle$ = "Si") Then xa = flec(k, j) / G: Put #8, , xa
O     Next j
O     Next k
O     'barra progreso...
O     BarraTiempo Progreso, (iterac), (itermax)
O     LbDesplaz.Caption = Str$(desplaz)
O     Lbiterac.Caption = iterac
O     End If
O
O     End If
O
O     If (iterac = itermax) Then nParar = 2 ' Finaliza si pasa el n°m x. de iteraciones
O
O     iterac = iterac + 1
O     Loop
O
O     'por que paró? (nParar):
O     ' 2- cumplió las iteraciones máximas
O     ' 3- el usuario pulsó el botón cancelar
O
O     Close #1
O     Close #2
O     Close #9
O     Close #7
O     Close #8
O
O
O     'Graba fichero de continuación...
O     a$ = Trim(nom$) + "\" + "contin." + extension$
O     Open a$ For Binary As #1
O
O     If (ndim = 3) Then 'Dos nudos.....
O     For j = 1 To 3 ' Graba desplazamientos y velocidades
O     For i = 1 To np2
O     Put #1, , C(i, j)
O     Put #1, , v(i, j)
O     Next i
O     Next j
O
O     Else 'nudo intermedio.....
O     For j = 1 To 3 ' Graba desplazamientos y velocidades
O     For i = 1 To np2
O     Put #1, , C(i, j)
O     Put #1, , v(i, j)
O     Next i
O     Next j
O     For j = 1 To 2
O     For i = 1 To ne
O     Put #1, , CA(i, j)
O     Put #1, , va(i, j)

```

```

O     Next i
O     Next j
O End If
O
O Close #1
O
O Unload Me 'sale del cálculo...
O
O End Sub
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O ' CALCULO DE ESFUERZOS MODELO DOS NUDOS.....
O Sub A2n(ct(), vt(), a(), tiempo, fuer())
O 'IN - ct(np,3),vt(np,3), tiempo
O 'OUT - a(np,3), fuer(ne,1)
O
O 'Double
O Dim rs()
O ReDim rs(np, 3)
O Dim x1, x2, y1, y2, z1, z2, al, ale, xp, yp, zp, c1, c2, c3, deff, frzamt, zz, velo
O Dim Resu, aMasa
O 'Enteros
O Dim i, j, k, n1, n2, ntipob
O
O 'Fuerzas externas en Nw.....
O For j = 1 To 3
O   For i = 1 To np
O     k = nq(i, j)
O     rs(i, j) = qst(i, j) * G + qDin(tiempo, k) * G
O   Next i
O Next j
O
O For k = 1 To ne 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
O   n1 = n11(k, 1) ' nudo inicial
O   n2 = n11(k, 2) ' nudo final
O   ntipob = n11(k, 3)
O   x1 = ct(n1, 1)
O   y1 = ct(n1, 2)
O   z1 = ct(n1, 3) 'coordenadas nudos
O   x2 = ct(n2, 1)
O   y2 = ct(n2, 2)
O   z2 = ct(n2, 3)
O   al = alon(k) 'long.incial
O   xp = x2 - x1
O   yp = y2 - y1
O   zp = z2 - z1 'Vector 1-2 ....
O   ale = Sqr(xp * xp + yp * yp + zp * zp) 'long.actual
O   c1 = xp / ale
O   c2 = yp / ale
O   c3 = zp / ale 'cosenos directores barra
O   'Cálculo fuerzas axiales.....
O   deff = (ale - al) / al
O   If ((p(ntipob, 2) <= 0) And (deff < 0)) Then 'cables.(ineracia<=0).....
O     zz = 0
O   Else
O     zz = (p(ntipob, 3) * G * p(ntipob, 1)) * deff 'E*g*A*ep
O   End If
O   fuer(k, 1) = zz
O   'Fuerzas desequilibradas (Nw)= cosenos directores * FUER.....
O   rs(n1, 1) = rs(n1, 1) + c1 * zz
O   rs(n1, 2) = rs(n1, 2) + c2 * zz
O   rs(n1, 3) = rs(n1, 3) + c3 * zz
O   rs(n2, 1) = rs(n2, 1) - c1 * zz
O   rs(n2, 2) = rs(n2, 2) - c2 * zz
O   rs(n2, 3) = rs(n2, 3) - c3 * zz
O Next k 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
O
O 'Fuerzas desequilibradas+coacciones
O For i = 1 To np
O   For j = 1 To 3
O     If (ct(i, j) >= bnx(i, j)) Then
O       rs(i, j) = rs(i, j) - kc * G * (ct(i, j) - bnx(i, j))
O     End If
O
O     If (ct(i, j) <= anx(i, j)) Then
O       rs(i, j) = rs(i, j) - kc * G * (ct(i, j) - anx(i, j))
O     End If

```

```

O Next j
O Next i
O
O 'Cálculo aceleraciones.....
O For i = 1 To np
O For j = 1 To 3
O velo = vt(i, j): Resu = rs(i, j): aMasa = amas(i)
O frzamt = Fueroz(amortt$, velo, Resu, aMasa, alambda, vCritica)
O a(i, j) = (Resu - frzamt) / aMasa 'aMasa - masa de los nudos
O Next j
O Next i
O
O 'Coacciones.....
O For i = 1 To np
O For j = 1 To 3
O If nx(i, j) <> 1234 Then a(i, j) = 0
O Next j
O Next i
O
O 'Aceleraciones exteriores.....
O 'k= tipo de fuerza externa...(rampa, seno...)
O For i = 1 To np
O For j = 1 To 3
O k = NA(i, j)
O If (k <> 1234) Then a(i, j) = aDin(tiempo, k)
O Next j
O Next i
O End Sub
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O 'CALCULO DE ESFUERZOS MODELO TRES NUDOS.....
O Sub A3n(ct(), cta(), vt(), vta(), a(), aa(), tiempo, fueraxil(), fuercort(),
O flector())
O 'IN - ct(np2,3),cta(ne,2),vt(np2,3),vta(ne,2),tiempo
O 'OUT - a(np2,3),aa(ne,2), fueraxil(ne,2),fuercort(ne,2),flector(ne,2)
O
O Dim rs(), rsa()
O 'dobles
O ReDim rs(np2, 3), rsa(ne, 2) 'Fuerzas y momentos desequilibrados
O Dim fle(3), bb(3), aaa(3), cc(3)
O Dim dd1(3), dd2(3), vx1(3), vy1(3), vz1(3), vx2(3), vy2(3), vz2(3)
O Dim deff, frzamt, thetaj1, thetaj3, qaxil, qaxi2, qcort1, qcort2, qflec1, qflec2
O Dim al, ale, ale1, ale3, h1, h2, h3, rig, vtt5, vtt4, stt5, stt4, zet, eta
O Dim velo, ym, x1, x2, x3, y1, y2, y3, z1, z2, z3, al1, al3, hMin
O Dim Resu, aMasa, qq, al, a2
O 'enteros
O Dim i, j, k, n1, n2, n3, ntipob ', nea
O 'cadenas
O Dim ab$
O
O hMin = 0.00000001 'flecha mínima a partir de la cual considera cort. y flexión
O MaCero rs
O MaCero rsa
O MaCero a
O MaCero aa
O
O 'Fuerzas dinámicas externas en Nw.....
O For i = 1 To np
O For j = 1 To 3
O k = nq(i, j)
O qq = qDin(tiempo, k) * G
O rs(i, j) = qst(i, j) * G + qq 'cargas estáticas kgf.g + dinámicas Nw
O Next j
O Next i
O
O aaa(3) = 0#
O
O For k = 1 To ne 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
O n1 = n11(k, 1) ' nudo inicial
O n3 = n11(k, 2) ' nudo final
O n2 = np + k ' nudo intermedio
O ntipob = n11(k, 3) ' tipo barra
O al = alon(k) ' long.equili. barra
O al1 = slon1(k) ' long.equili .semibarra 1-2
O al3 = al - al1 ' long.equili.semibarra 2-3
O 'coordenadas globales
O x1 = ct(n1, 1) 'nudo inicial

```

```

O   y1 = ct(n1, 2)
O   z1 = ct(n1, 3)
O   x2 = ct(n2, 1) 'nudo intermedio
O   y2 = ct(n2, 2)
O   z2 = ct(n2, 3)
O   zet = cta(k, 1)
O   eta = cta(k, 2)
O   x3 = ct(n3, 1) 'nudo final
O   y3 = ct(n3, 2)
O   z3 = ct(n3, 3)
O
O   'longitud real semibarra 1-2
O   vx1(1) = (x2 - x1)
O   vx1(2) = (y2 - y1)
O   vx1(3) = (z2 - z1)
O   ale1 = vMod(vx1)
O   vx1(1) = vx1(1) / ale1 'unitario 1-2
O   vx1(2) = vx1(2) / ale1
O   vx1(3) = vx1(3) / ale1
O
O   'longitud real semibarra 2-3
O   vx2(1) = (x2 - x3)
O   vx2(2) = (y2 - y3)
O   vx2(3) = (z2 - z3)
O   ale3 = vMod(vx2)
O   vx2(1) = vx2(1) / ale3 'unitario 3-2
O   vx2(2) = vx2(2) / ale3
O   vx2(3) = vx2(3) / ale3
O
O   'vectores unitarios locales
O   aaa(1) = Cos(eta)
O   aaa(2) = Sin(eta)
O   bb(1) = Sin(zet) * aaa(2)
O   bb(2) = -Sin(zet) * aaa(1)
O   bb(3) = Cos(zet)
O
O   Call ProVect(bb, aaa, cc)
O
O   'SEMIBARRA 1-2 .....
O
O   Call ProVect(vx1, bb, dd1)
O   h1 = vMod(dd1) 'h1=flecha/ale1
O   'h1=sen(theta1) seno del ángulo que está girado el punto medio con respecto a los
O       1-3
O
O   'Cálculo fuerzas axiles 1...
O   deff = (all - ale1) / all 'def. cambiada de signo
O   ym = p(ntipob, 3)
O   qcort1 = 0
O   qflecl = 0
O
O   If p(ntipob, 2) <= 0 Then 'cables...
O       If deff > 0 Then 'acortamientos
O           qaxil = 0
O       Else
O           qaxil = (ym * G * p(ntipob, 1)) * deff ' E*g * A* -dL/L en Nw
O       End If
O   Else
O       qaxil = (ym * G * p(ntipob, 1)) * deff ' E*g * A* -dL/L en Nw
O       fueraxil(k, 1) = qaxil
O       'thetaj1 = 0
O       If (h1 > hMin) Then 'cortante y flector para barras dobladas
O           vz1(1) = dd1(1) / h1
O           vz1(2) = dd1(2) / h1
O           vz1(3) = dd1(3) / h1
O           Call ProVect(dd1, vx1, vy1)
O           h3 = vMod(vy1)
O           vy1(1) = vy1(1) / h3
O           vy1(2) = vy1(2) / h3
O           vy1(3) = vy1(3) / h3
O           'thetaj1 = ASin(h1)
O           rig = (ym * p(ntipob, 2) * 0.0001 * G) / ale1 'E*g * I*1e-4 / L en Nw.m.
O           If (axil$ = "Si") Then rig = Correc(qaxil, rig, ale1) Else rig = 3 * rig
O           qflecl = rig * h1 ' h1=flecha/L
O           qcort1 = qflecl / ale1 ' 3EI/L³·flecha
O       End If
O   End If
O   fuercort(k, 1) = qcort1
O   flector(k, 1) = qflecl

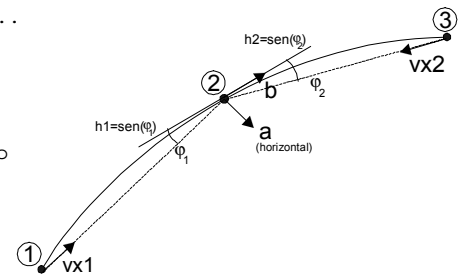
```



```

O
O
O 'SEMIBARRA 2-3 .....
O
O Call ProVect(bb, vx2, dd2)
O h2 = vMod(dd2)
O
O 'Cálculo fuerzas axiales 2...
O deff = (al3 - ale3) / al3 'def. cambiada de signo
O ym = p(ntipob, 3)
O qcort2 = 0
O qflec2 = 0
O
O If p(ntipob, 2) <= 0 Then 'cables...
O   If deff > 0 Then 'acortamientos
O     qaxi2 = 0
O   Else
O     qaxi2 = (ym * G * p(ntipob, 1)) * deff ' E*g * A* -dL/L en Nw
O   End If
O   Else
O     qaxi2 = (ym * G * p(ntipob, 1)) * deff
O     fueraxil(k, 2) = qaxi2
O     'thetaj3 = 0
O     If (h2 > hMin) Then 'cortante y flector para barras dobladas
O       vz2(1) = dd2(1) / h2
O       vz2(2) = dd2(2) / h2
O       vz2(3) = dd2(3) / h2
O       Call ProVect(dd2, vx2, vy2)
O       h3 = vMod(vy2)
O       vy2(1) = vy2(1) / h3
O       vy2(2) = vy2(2) / h3
O       vy2(3) = vy2(3) / h3
O       'thetaj3 = ASin(h2)
O       rig = (ym * p(ntipob, 2) * 0.0001 * G) / ale3
O       If (axil$ = "Si") Then rig = Correc(qaxi2, rig, ale3) Else rig = 3 * rig
O       qflec2 = rig * h2
O       qcort2 = qflec2 / ale3 ' 3EI/L³·flecha
O     End If
O   End If
O   fuercort(k, 2) = qcort2
O   flector(k, 2) = qflec2
O
O 'Cortante y axil
O For j = 1 To 3
O   a1 = -vx1(j) * qaxi1 + vy1(j) * qcort1
O   a2 = -vx2(j) * qaxi2 + vy2(j) * qcort2
O   rs(n1, j) = rs(n1, j) + a1
O   rs(n3, j) = rs(n3, j) + a2
O   rs(n2, j) = rs(n2, j) - a1 - a2
O 'flector
O   fle(j) = vz1(j) * qflec1 + vz2(j) * qflec2
O Next j
O rsa(k, 1) = fle(1) * aaa(1) + fle(2) * aaa(2)
O rsa(k, 2) = fle(1) * cc(1) + fle(2) * cc(2) + fle(3) * cc(3)
O
O Next k 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
O
O ' Fuerzas desequilibradas + coacciones límite.....
O For i = 1 To np
O   For j = 1 To 3
O     qq = 0
O     If (ct(i, j) >= bnx(i, j)) Then qq = -kc * G * (ct(i, j) - bnx(i, j)) 'topes
O     If (ct(i, j) <= anx(i, j)) Then qq = -kc * G * (ct(i, j) - anx(i, j))
O     rs(i, j) = rs(i, j) + qq
O   Next j
O Next i
O
O ' Cálculo aceleraciones.....
O For i = 1 To np2
O   For j = 1 To 3
O     velo = vt(i, j): Resu = rs(i, j): aMasa = amas(i)
O     frzamt = Fueroz(amortt$, velo, Resu, aMasa, alambda, vCritica)
O     a(i, j) = (Resu - frzamt) / aMasa
O   Next j
O Next i
O
O 'Coacciones exteriores: puntos fijos.....
O For i = 1 To np
O   For j = 1 To 3
O     If nx(i, j) <> 1234 Then a(i, j) = 0

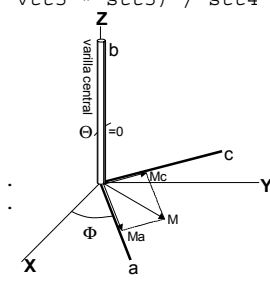
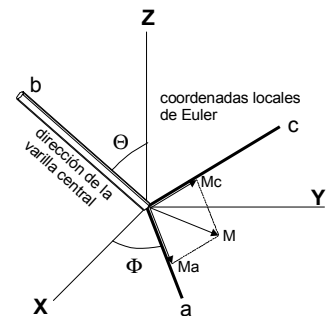
```



```

O Next j
O Next i
O
O 'Aceleraciones exteriores
O For i = 1 To np
O For j = 1 To 3
O k = NA(i, j)
O If k <> 1234 Then a(i, j) = aDin(tiempo, k)
O Next j
O Next i
O
O 'Barras enlazadas: nea = ne2 - ne 'global
O If (nea > 0) Then
O For k = ne + 1 To ne2
O n1 = n11(k, 1) ' nudo inicial
O n2 = n11(k, 2) ' nudo final
O For j = 1 To 3
O a(n1, j) = a(n1, j) * amas(n1)
O a(n2, j) = a(n2, j) * amas(n2)
O a(n1, j) = (a(n1, j) + a(n2, j)) / (amas(n1) + amas(n2))
O a(n2, j) = a(n1, j)
O Next j
O Next k
O End If
O
O 'Cálculo aceleraciones angulares
O For k = 1 To ne
O vtt5 = vta(k, 2)
O vtt4 = vta(k, 1)
O stt4 = cta(k, 1)
O stt5 = Cos(stt4)
O stt4 = Sin(stt4)
O
O frzamt = alambda * vtt4 'amortiguamiento
O aa(k, 1) = (rsa(k, 1) / ainer(np + k)) + vtt5 * vtt5 * stt5 * stt4 - frzamt
O frzamt = alambda * vtt5
O If (Abs(stt4) > 0.00001) Then
O aa(k, 2) = ((rsa(k, 2) / ainer(np + k)) - 2 * vtt4 * vtt5 * stt5) / stt4 - frzamt
O Else
O aa(k, 2) = 0
O End If
O Next k
O End Sub
O
O '.....
O '.....
O ' Factores de corrección debidos al axil
O Function Correc(ByVal p, ByVal rig, ByVal ale)
O 'IN - p-carga, rig-rigidez EI/L, ale-longitud
O 'OUT- Correc-nueva rigidez
O Dim u1, u, w, sss, ttt
O u1 = (p * ale / rig)
O If (u1 < -0.00001) Then
O u = Sqr(-u1)
O w = u * HSin(u) - 2 * HCos(u) + 2
O sss = u * (u * HCos(u) - HSin(u)) * rig / w
O ttt = u * (HSin(u) - u) * rig / w
O Correc = (sss - (ttt * ttt / sss))
O ElseIf (u1 > 0.00001) Then
O u = Sqr(u1)
O w = -u * Sin(u) - 2 * Cos(u) + 2
O sss = -u * (u * Cos(u) - Sin(u)) * rig / w
O ttt = -u * ((Sin(u) - u)) * rig / w
O Correc = (sss - (ttt * ttt / sss))
O Else
O Correc = 3 * rig
O End If
O End Function
O
O '.....
O '.....
O Function qDin(ByVal tiempo, ByVal ntipo) ' Result(b)
O 'Cargas dinámicas en Kgf
O Dim NN, delt, ti, b
O
O Select Case (ntipo)

```



```

O Case (-3) 'Función tabla 1 Interpolación lineal
O If (tiempo < t0(-3)) Then b = f0(-3)
O If (tiempo >= t0(-3) And tiempo <= t1(-3)) Then
O deltax = (t1(-3) - t0(-3)) / (npqd(1) - 1)
O NN = Int(((tiempo - t0(-3)) / deltax))
O ti = tiempo - t0(-3) - NN * deltax
O b = f0(-3) + f1(-3) * (tabqd1(NN + 1) + ti / deltax * (tabqd1(NN + 2) -
O tabqd1(NN + 1)))
O End If
O If (tiempo > t1(-3)) Then b = f0(-3) + f1(-3) * tabqd1(npqd(1))
O Case (-2) 'Función tabla 2 Interpolación lineal
O If (tiempo < t0(-2)) Then b = f0(-2)
O If (tiempo >= t0(-2) And tiempo <= t1(-2)) Then
O deltax = (t1(-2) - t0(-2)) / (npqd(2) - 1)
O NN = Int(((tiempo - t0(-2)) / deltax))
O ti = tiempo - t0(-2) - NN * deltax
O b = f0(-2) + f1(-2) * (tabqd2(NN + 1) + ti / deltax * (tabqd2(NN + 2) -
O tabqd2(NN + 1)))
O End If
O If (tiempo > t1(-2)) Then b = f0(-2) + f1(-2) * tabqd2(npqd(2))
O Case (-1) 'Función tabla 3 Interpolación lineal
O If (tiempo < t0(-1)) Then b = f0(-1)
O If (tiempo >= t0(-1) And tiempo <= t1(-1)) Then
O deltax = (t1(-1) - t0(-1)) / (npqd(3) - 1)
O NN = Int(((tiempo - t0(-1)) / deltax))
O ti = tiempo - t0(-1) - NN * deltax
O b = f0(-1) + f1(-1) * (tabqd3(NN + 1) + ti / deltax * (tabqd3(NN + 2) -
O tabqd3(NN + 1)))
O End If
O If (tiempo > t1(-1)) Then b = f0(-1) + f1(-1) * tabqd3(npqd(3))
O Case (0)
O b = 0#
O Case (1) 'Función rampa 1
O If (tiempo < t0(1)) Then b = f0(1)
O If (tiempo <= t1(1) And tiempo >= t0(1)) Then b = f0(1) + f1(1) * (tiempo -
O t0(1)) / (t1(1) - t0(1))
O If (tiempo > t1(1)) Then b = f1(1) + f0(1)
O Case (2) 'Función rampa 2
O If (tiempo < t0(2)) Then b = f0(2)
O If (tiempo <= t1(2) And tiempo >= t0(2)) Then b = f0(2) + f1(2) * (tiempo -
O t0(2)) / (t1(2) - t0(2))
O If (tiempo > t1(2)) Then b = f1(2) + f0(2)
O Case (3) 'Función rampa 3
O If (tiempo < t0(3)) Then b = f0(3)
O If (tiempo <= t1(3) And tiempo >= t0(3)) Then b = f0(3) + f1(3) * (tiempo -
O t0(3)) / (t1(3) - t0(3))
O If (tiempo > t1(3)) Then b = f1(3) + f0(3)
O Case (4) 'Función seno 1
O If (tiempo < t0(4)) Then b = f0(4)
O If (tiempo >= t0(4)) Then b = f0(4) + f1(4) * Sin(2# * pi * (tiempo - t0(4)) /
O t1(4))
O Case (5) 'Función seno 2
O If (tiempo < t0(5)) Then b = f0(5)
O If (tiempo >= t0(5)) Then b = f0(5) + f1(5) * Sin(2# * pi * (tiempo - t0(5)) /
O t1(5))
O Case (6) 'Función seno 1
O If (tiempo < t0(6)) Then b = f0(6)
O If (tiempo >= t0(6)) Then b = f0(6) + f1(6) * Sin(2# * pi * (tiempo - t0(6)) /
O t1(6))
O Case Default
O b = 0#
O End Select
O qDin = b
O End Function
O
O .....
O .....
O Function aDin(tiempo, ntipo)
O 'Cargas dinámicas en Kgf
O Dim NN, deltax, ti, b
O
O Select Case (ntipo)
O Case (0)
O b = 0#
O Case (1) 'Función rampa 1
O If (tiempo < at0(1)) Then b = af0(1)
O If (tiempo <= at1(1) And tiempo >= at0(1)) Then
O b = af0(1) + afl(1) * (tiempo - at0(1)) / (at1(1) - at0(1))
O End If

```

```

O      If (tiempo > at1(1)) Then b = af1(1) + af0(1)
O      Case (2) 'Función rampa 2
O      If (tiempo < at0(2)) Then b = af0(2)
O      If (tiempo <= at1(2) And tiempo >= at0(2)) Then
O      b = af0(2) + af1(2) * (tiempo - at0(2)) / (at1(2) - at0(2))
O      End If
O      If (tiempo > at1(2)) Then b = af1(2) + af0(2)
O      Case (3) 'Función rampa 3
O      If (tiempo < at0(3)) Then b = af0(3)
O      If (tiempo <= at1(3) And tiempo >= at0(3)) Then
O      b = af0(3) + af1(3) * (tiempo - at0(3)) / (at1(3) - at0(3))
O      End If
O      If (tiempo > at1(3)) Then b = af1(3) + af0(3)
O      Case (4) 'Función seno 1
O      If (tiempo < at0(4)) Then b = af0(4)
O      If (tiempo >= at0(4)) Then
O      b = af0(4) + af1(4) * Sin(2# * pi * (tiempo - at0(4)) / at1(4))
O      End If
O      Case (5) 'Función seno 2
O      If (tiempo < at0(5)) Then b = af0(5)
O      If (tiempo >= at0(5)) Then
O      b = af0(5) + af1(5) * Sin(2# * pi * (tiempo - at0(5)) / at1(5))
O      End If
O      Case (6) 'Función seno 3
O      If (tiempo < at0(6)) Then b = af0(6)
O      If (tiempo >= at0(6)) Then
O      b = af0(6) + af1(6) * Sin(2# * pi * (tiempo - at0(6)) / at1(6))
O      End If
O      Case (-3) 'Función tabla 3 Interpolación lineal
O      If (tiempo < at0(-3)) Then b = af0(-3)
O      If (tiempo >= at0(-3) And tiempo <= at1(-3)) Then
O      delt = (at1(-3) - at0(-3)) / (npad(1) - 1)
O      NN = Int(((tiempo - at0(-3)) / delt))
O      ti = tiempo - at0(-3) - NN * delt
O      b = af0(-3) + af1(-3) * (tabad1(NN + 1) + ti / delt * (tabad1(NN + 2) -
O      tabad1(NN + 1)))
O      End If
O      If (tiempo > at1(-3)) Then b = af0(-3) + af1(-3) * tabad1(npad(1))
O      Case (-2) 'Función tabla 2 Interpolación lineal
O      If (tiempo < at0(-2)) Then b = af0(-2)
O      If (tiempo >= at0(-2) And tiempo <= at1(-2)) Then
O      delt = (at1(-2) - at0(-2)) / (npad(2) - 1)
O      NN = Int(((tiempo - at0(-2)) / delt))
O      ti = tiempo - at0(-2) - NN * delt
O      b = af0(-2) + af1(-2) * (tabad2(NN + 1) + ti / delt * (tabad2(NN + 2) -
O      tabad2(NN + 1)))
O      End If
O      If (tiempo > at1(-2)) Then b = af0(-2) + af1(-2) * tabad2(npad(2))
O      Case (-1) 'Función tabla 1 Interpolación lineal
O      If (tiempo < at0(-1)) Then b = af0(-1)
O      If (tiempo >= at0(-1) And tiempo <= at1(-1)) Then
O      delt = (at1(-1) - at0(-1)) / (npad(3) - 1)
O      NN = Int(((tiempo - at0(-1)) / delt))
O      ti = tiempo - at0(-1) - NN * delt
O      b = af0(-1) + af1(-1) * (tabad3(NN + 1) + ti / delt * (tabad3(NN + 2) -
O      tabad3(NN + 1)))
O      End If
O      If (tiempo > at1(-1)) Then b = af0(-1) + af1(-1) * tabad3(npad(3))
O      Case Default
O      b = 0
O      End Select
O      aDin = b
O
O      End Function
O
O      '.....
O      '.....
O      Function Fueroz(tipo$, ByVal Velocidad As Double, ByVal Resultante As Double, ByVal
O      Masa As Double, ByVal lambda As Double, ByVal vCritica As Double)
O      If (tipo$ = "Rozamiento") Then
O      If (Abs(Velocidad) < vCritica) Then
O      Fueroz = lambda * 1.2 * Masa
O      If (Fueroz >= Abs(Resultante)) Then
O      Fueroz = Resultante
O      Else
O      Fueroz = Sign(Fueroz, Resultante)
O      End If
O      Else
O      Fueroz = Sign(Fueroz, Resultante)
O      End If
O      Else

```

```

O         Fueroz = Sign(lambda * Masa, Velocidad)
O     End If
O     Else
O         Fueroz = lambda * Velocidad * Masa      ' Amortiguamiento viscoso
O     End If
O End Function
O
O '.....
O '.....
O Function ymDin(ByVal def, ByVal ntip)
O 'Módulo de Young variable
O ymDin = p(ntip, 3) 'no variable
O End Function
O
O '.....
O '.....
O Sub Avisovel(i As Long)
O Dim a$
O a$ = "El nudo n°" + Str$(i) + " sobrepasó la velocidad máxima!!!"
O MsgBox a$, vbOKOnly, "Error"
O End Sub
O
O '.....
O '.....
O Sub mTranspose(ByRef t1(), ByRef t2())
O 'IN-t1  OUT-t2
O Dim i, j, i0, j0, i1, j1
O i0 = LBound(t1, 1)
O j0 = LBound(t1, 2)
O i1 = UBound(t1, 1)
O j1 = UBound(t1, 2)
O 'ReDim t2(j0 To j1, i0 To i1)
O For i = i0 To i1
O     For j = j0 To j1
O         t2(j, i) = t1(i, j)
O     Next j
O Next i
O End Sub
O
O '.....
O '.....
O 'pone a cero una matriz, puede tener 1, 2 o 3 dimensiones.
O Sub MaCero(ByRef TT())
O Dim i, j, k, i0, i1, j0, j1, k0, k1, n
O n = NofDim(TT) 'número de dimensiones de tt()
O Select Case n
O     Case 1
O         i0 = LBound(TT, 1)
O         i1 = UBound(TT, 1)
O         For i = i0 To i1
O             TT(i) = 0
O         Next i
O     Case 2
O         i0 = LBound(TT, 1)
O         i1 = UBound(TT, 1)
O         j0 = LBound(TT, 2)
O         j1 = UBound(TT, 2)
O         For i = i0 To i1
O             For j = j0 To j1
O                 TT(i, j) = 0
O             Next j
O         Next i
O     Case 3
O         i0 = LBound(TT, 1)
O         i1 = UBound(TT, 1)
O         j0 = LBound(TT, 2)
O         j1 = UBound(TT, 2)
O         k0 = LBound(TT, 3)
O         k1 = UBound(TT, 3)
O         For i = i0 To i1
O             For j = j0 To j1
O                 For k = k0 To k1
O                     TT(i, j, k) = 0
O                 Next k
O             Next j
O         Next i
O     Case Else: Stop
O End Select
O

```

```

O End Sub
O
O '.....
O '.....
O Sub mIguala(t2(), t1())
O 'hace t2=t1
O 'IN-t1 OUT-t2
O Dim i, j, i0, j0, i1, j1
O i0 = LBound(t1, 1)
O j0 = LBound(t1, 2)
O i1 = UBound(t1, 1)
O j1 = UBound(t1, 2)
O For i = i0 To i1
O For j = j0 To j1
O t2(i, j) = t1(i, j)
O Next j
O Next i
O End Sub
O
O '.....
O '.....
O 'Producto matricial. Se supone que el indice menor es 1?
O 'las matrices pueden tener una o dos dimensiones
O Sub MatMul(ByRef a1(), ByRef a2(), ByRef Resul())
O 'Dim a1(n1, nc1), a2(n2, nc2), a(n1, nc2)
O 'IN- a1,a2 OUT-a
O Dim n1, nc1, n2, nc2 'lineas y columnas de las matrices 1 y 2
O Dim n1R, ncR 'lineas y columnas del resultado
O Dim i, j, n, nd1, nd2, ndr, kal
O
O n1 = UBound(a1, 1): n2 = UBound(a2, 1): n1R = UBound(Resul, 1)
O nd1 = NofDim(a1): nd2 = NofDim(a2): ndr = NofDim(Resul)
O If nd1 > 1 Then nc1 = UBound(a1, 2) Else nc1 = n1: n1 = 1 'pasamos a vect. columna
O If nd2 > 1 Then nc2 = UBound(a2, 2) Else nc2 = 1
O If ndr > 1 Then ncR = UBound(Resul, 2) Else ncR = 1
O
O If nc1 <> nc2 Then MsgBox "Las matrices no se pueden multiplicar", vbOKOnly,
O "Error": Stop
O If n1 <> n1R Then MsgBox "La matriz resultado no tiene las dimensiones correctas",
O vbOKOnly, "Error": Stop
O If nc2 <> ncR Then MsgBox "La matriz resultado no tiene las dimensiones correctas",
O vbOKOnly, "Error": Stop
O
O MaCero Resul ' limpiamos la matriz resultado
O
O 'tipo de cálculo
O kal = 0
O If n1 = 1 Then kal = 1 ' a(0, nc2) ndimRes = 1
O If nc2 = 1 Then kal = 2 ' a(n1, 0) ndimRes = 1
O If n1 = 1 And nc2 = 1 Then kal = 3 ' ???
O
O 'cálculo del producto...
O For i = 1 To n1
O For j = 1 To nc2
O For n = 1 To nc1
O Select Case kal
O Case 0: Resul(i, j) = Resul(i, j) + a1(i, n) * a2(n, j)
O Case 1: Resul(j) = Resul(j) + a1(n) * a2(n, j)
O Case 2: Resul(i) = Resul(i) + (a1(i, n) * a2(n))
O Case 3: Resul(i, j) = Resul(i, j) + a1(n) * a2(n) 'Result es matriz 1x1 ???
O End Select
O Next n
O Next j
O Next i
O End Sub
O
O '.....
O '.....
O Function NofDim(ByRef TT())
O 'Determina el número de dimensiones de una matriz
O On Error GoTo errores
O Dim i, n
O n = 0
O Do
O n = n + 1
O i = LBound(TT, n)
O Loop
O errores:
O NofDim = n - 1

```

```

O Resume fin
O fin:
O End Function
O
O '.....
O '.....
O Sub ProVect(u(), v(), w())
O 'Multiplica vectorialmente las componentes u por las v obteniendo el resultado w
O w(1) = u(2) * v(3) - v(2) * u(3)
O w(2) = v(1) * u(3) - u(1) * v(3)
O w(3) = u(1) * v(2) - u(2) * v(1)
O End Sub
O
O '.....
O '.....
O 'Calcula el módulo del vector u
O Function vMod(u())
O Dim a As Double, b As Double
O vMod = Sqr(u(1) * u(1) + u(2) * u(2) + u(3) * u(3))
O End Function
O
O '.....
O '.....
O Function Sign(ByVal valor, ByVal Signo) As Double
O Sign = Abs(valor) * Sgn(Signo)
O End Function
O
O '.....
O '.....
O ' arcoseno
O ' error si el valor de entrada está fuera del rango [-1,1]
O Function ASin(ByVal value)
O If Abs(value) <> 1 Then
O ASin = Atn(value / Sqr(1 - value * value))
O Else
O ASin = 1.5707963267949 * Sgn(value)
O End If
O End Function
O
O '.....
O '.....
O ' arcocoseno
O ' error si el valor de entrada está fuera del rango [-1,1]
O Function ACos(ByVal value)
O If Abs(value) <> 1 Then
O ACos = 1.5707963267949 - Atn(value / Sqr(1 - value * value))
O ElseIf value = -1 Then
O ACos = 3.14159265358979
O ElseIf value = 1 Then ACos = 0 '(implicit)
O End If
O End Function
O
O '.....
O '.....
O 'arcotangente de Y/X
O Function Atn2(ByVal x As Double, ByVal y As Double) As Double
O 'mide desde la parte positiva del eje X y en sentido antihorario '===
O If x = 0 Then
O Atn2 = Sgn(y) * 1.5707963267949
O ElseIf x > 0 Then
O Atn2 = Atn(y / x)
O Else
O Atn2 = Atn(y / x) + 3.14159265358979
O End If
O 'hasta aquí devuelve un valor entre -0.5pi y 1.5pi
O ''''If Atn2 < 0 Then Atn2 = Atn2 + 6.28318530717959
O ''''ahora devolvería un valor entre 0 y 2pi
O End Function
O
O '.....
O '.....
O Function Atn3(ByVal x As Double, ByVal y As Double) As Double
O 'mide desde la parte negativa del eje Y y en sentido antihorario '===
O Atn3 = Atn2(x, y) + 1.5707963267949
O 'devuelve un valor entre 0 y 2pi '===
O
O End Function
O
O '.....
O '.....

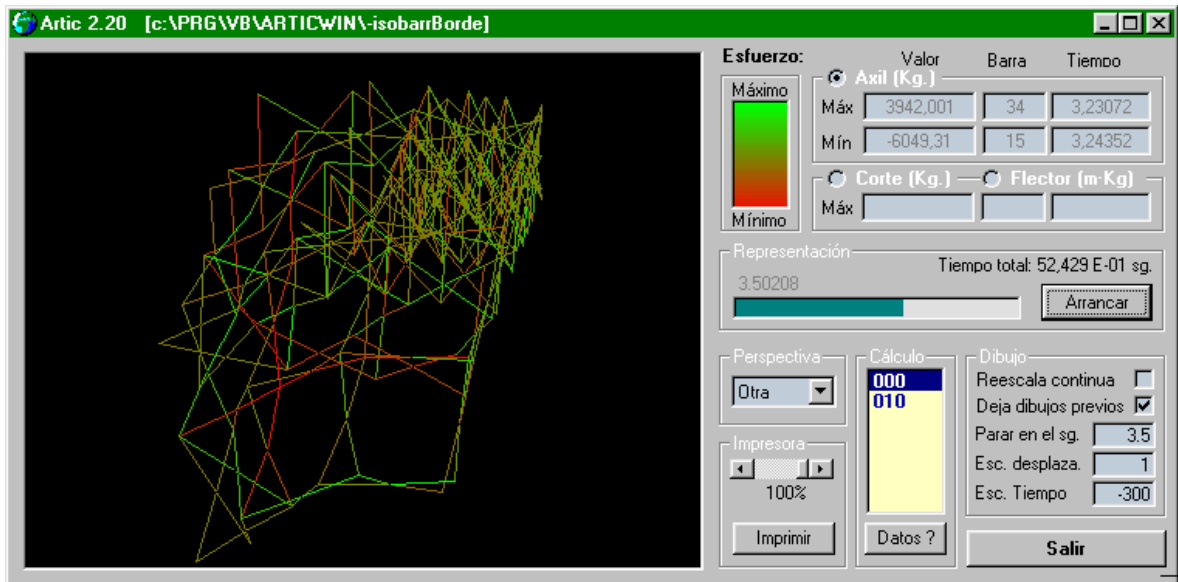
```

```

O ' Seno hiperbólico
O Function HSin(ByVal x) As Double
O   HSin = (Exp(x) - Exp(-x)) / 2
O End Function
O
O .....
O
O ' Coseno hiperbólico
O Function HCos(ByVal x) As Double
O   If x > 709.782712893 Then Stop
O   HCos = (Exp(x) + Exp(-x)) / 2
O End Function
O

```

## 7.3.2 MÓDULO DE ANÁLISIS GLOBAL



```

O
O Option Explicit
O DefSng A-H
O DefInt I-N
O DefSng O-Z
O Dim hx1, hx2, hx3, hx4, hx5, np2 As Long
O Dim hm10, hm1, hm20, hm2, Esc0, Esc
O Dim Parar As Boolean, TiempoAct As Single
O Dim iterac, itergrb As Long, tiempo, dt, grabanu$, iterini%
O 'Dim Creal() 'coordenadas sim amplificar, para cálculo alargamiento barras
O Dim L() As Long, C(), C0(), cp(), CP0(), gr(), lon#()
O Dim Escala, EscalaTemp, nnd$, EmaxD, Extension$
O Dim loneje ' longitud ejes
O 'Dim Wo, iterac, dt, iterini%, itergrb, nnd$, grabanu$, tiempo, comentario$
O 'Dim ik, im, il, itergrb, itt, l
O 'Dim itermax As Long
O 'Dim dt, alambda, velmax
O 'Dim amortt$, ppropio$, nnd$, extension$, axil$, comentario$, cont$
O 'Dim ncont, np2, ne2, np, ne, ndim, npp
O 'Dim ctt(), vtt()
O
O Sub axo3d2d(x, y, Z, a, b)
O 'x,y,z coordenadas 3d
O ' a,b coordenadas 2d
O 'pasa de 3d a 2d las coordenadas de axonometrías.....
O a = hm1 + Esc * (hx4 * y - hx5 * x)
O b = hm2 - Esc * (-hx1 * x - hx2 * y + hx3 * Z)
O
O End Sub
O
O
O

```



```

○ Private Sub ChReescala_Click()
○ 'If Parar = False Then Dib.Cls
○ End Sub
○
○ Private Sub Esfuerz_Click(index As Integer)
○ If index = 3 Then index = 2
○ EscalCol Escal, index
○ End Sub
○
○ Private Sub Form_Load()
○ Parar = False
○ End Sub
○
○ Private Sub Form_Resize()
○ Dim aa As Single, Dx As Single, Dy As Single
○ If WindowState <> 1 Then 'no minimizada
○ 'ajustar tamaño form al tamaño de la pantalla.....
○ aa = Dib.Left
○ FrDer.Left = Me.Width - FrDer.Width - aa
○ Dx = FrDer.Left - aa * 2.1
○ Dy = Me.ScaleHeight - Dib.Top * 1.9
○ If Dx < 100 Then Dx = 100
○ If Dy < 100 Then Dy = 100
○ Dib.Width = Dx
○ Dib.Height = Dy
○ End If
○
○ End Sub
○
○ Private Sub HPrn_Change()
○ LbPorc.Caption = Str$(HPrn.value) + "%"
○ End Sub
○
○ Private Sub Imprimir_Click()
○ Dim Fe As Single 'factor de escala
○ Dim px As Single, py As Single, pxd As Single, pyd As Single, Esc As Single
○ 'Dim taman2 As Single
○ Printer.FontName = "Arial"
○ Printer.FontSize = 8
○ Printer.Print "Estructura: " + nom$
○ Printer.Print "Cálculo: " + extension$
○ Printer.Print "Tiempo: "; TiempoAct; " sg."
○
○ Fe = HPrn.value / 100
○
○ pxd = Dib.ScaleWidth: pyd = Dib.ScaleHeight
○ px = Printer.ScaleWidth: py = Printer.ScaleHeight
○ If (pxd / px) < (pyd / py) Then Esc = py / pyd Else Esc = px / pxd 'escala
○ Esc = Esc * Fe
○
○ 'taman2 = taman: taman = taman * Esc * 0.9
○ DibPan Printer, Esc, False
○ Printer.EndDoc
○ 'taman = taman2
○
○ End Sub
○
○ Private Sub VerDatos_Click()
○ DatoDin.Show
○ End Sub
○
○ Private Sub Form_Activate()
○ Dim a$, aa, j, MyName As String
○ Dim Archiv$
○
○ If Parar = True Then Exit Sub 'si ya está iniciado el programa
○
○ Parar = True 'inicializa variable
○ LeeGral
○ 'LeeDat
○
○ 'inicialización variables.....
○ loneje = 3 'longitud de los ejes en metros
○ Escala = 1: EscEsp.Text = CStr(Escala)
○ EscalaTemp = 1: EscTmp.Text = CStr(EscalaTemp)

```

```

O 'EmaxD = 1:          EsfuMax.Text = CStr(EmaxD)
O
O EscalCol Escal, 1
O
O
O 'Nombre de la estructura en el título del form.....
O AnGlobal.Caption = titulo$ + " [" + nom$ + "]"
O
O 'Lista de cálculos realizados de la estructura actual.....
O Archiv$ = nom$ + "\parametr.*"
O ListExt ListCalc, Archiv$
O
O If ListCalc.Enabled = False Then
O   FrEsc.Enabled = False
O   Arrancar.Enabled = False
O   VerDatos.Enabled = False
O Else
O   'Perspec_Click (7)
O   Perspec.ListIndex = 7
O End If
O
O
O 'List1.Clear
O 'Myname = Dir(Nom$ + "\RESULTD.*") ' Retrieve the first entry.
O ''Myname = Right$(Myname, 3)
O 'Do While Myname <> "" ' Start the loop.
O '   Myname = Dir           ' Get next entry.
O '   If Myname <> "" Then
O '     Myname = UCase$(Right$(Myname, 3))
O '     List1.AddItem Myname ' Agrega la entrada.
O '   End If
O 'Loop
O 'If List1.ListCount > 0 Then 'hay al menos un elemento
O '   List1.ListIndex = 0
O 'End If
O
O End Sub
O Private Sub Dibujo()
O Dim a$, Iter, i, j, poj$, b As Double
O Dim EmaxV, EminV, EmaxB, EminB, EmaxT, EminT 'para guardar los valores máx. y mínimos
O Dim k, Esfu As Integer
O
O EmaxV = 0.0000001: EminV = 0: EmaxB = 0: EminB = 0: EmaxT = 0: EminT = 0
O a$ = nom$ + "\RESULTC." + extension$: Open a$ For Binary As #1 Len = 8
O
O Esfu = 0
O If Esfuerz(1).value Then Esfu = 1
O If Esfuerz(2).value Then Esfu = 2
O If Esfuerz(3).value Then Esfu = 3
O If Esfu = 1 Then a$ = nom$ + "\axil." + extension$: Open a$ For Binary As #2 Len = 8
O If Esfu = 2 Then a$ = nom$ + "\cort." + extension$: Open a$ For Binary As #2 Len = 8
O If Esfu = 3 Then a$ = nom$ + "\flec." + extension$: Open a$ For Binary As #2 Len = 8
O
O Dib.AutoRedraw = False: TimeBar.AutoRedraw = False
O Parar = EOF(1)
O If Parar Then Dib.AutoRedraw = True: TimeBar.AutoRedraw = True
O
O Iter = 0
O Do While Parar = False
O   Iter = Iter + 1
O   TiempoAct = dt * Iter
O   For j = 1 To 3
O     For i = 1 To np2
O       Get #1, , b
O
O If Iter = 50 And j = 3 And i = 38 Then Debug.Print Iter, j, i, b
O '       Select Case grabanu$
O '         Case "Todos": Input #1, b
O '         Case Else: If gr(i) <> 0 Then Input #1, b
O '       End Select
O       C(i, j) = C0(i, j) + ((b - C0(i, j)) * Escala)
O       Creal(i, j) = C0(i, j) + b
O     Next i
O   Next j
O   If (mnd$ = "Si") Then 'Barras con 3 nudos.....
O     For j = 4 To 5: For i = np + 1 To np2
O       Get #1, , b 'giros
O If Iter = 50 And j = 4 And i = np + 3 Then Debug.Print Iter, j, i, b
O     Next i: Next j

```

```

O End If
O If Esfu <> 0 Then
O For k = 1 To ne 'barra.....
O Get #2, , b 'esfuer 1
O If Esfu = 1 Then 'axil
O If b > EmaxV Then EmaxV = b: EmaxB = k: EmaxT = TiempoAct
O If b < EminV Then EminV = b: EminB = k: EminT = TiempoAct
O b = (b / EmaxV) * 127 'exageración del color
O End If
O If Esfu = 2 Or Esfu = 3 Then 'cortante o flector
O b = Abs(b)
O If b > EmaxV Then EmaxV = b: EmaxB = k: EmaxT = TiempoAct
O b = (b / EmaxV) * 254 - 127 'exageración del color
O End If
O If b > 127 Then b = 127
O If b < -127 Then b = -127
O L(k, 3) = b 'color esfuerzo 1'.....
O If nnd$ = "Si" Then 'segundo tramo de la barra
O Get #2, , b 'esfuer 2
O If Esfu = 1 Then 'axil
O If b > EmaxV Then EmaxV = b: EmaxB = k: EmaxT = TiempoAct
O If b < EminV Then EminV = b: EminB = k: EminT = TiempoAct
O b = (b / EmaxV) * 127 'exageración del color
O End If
O If Esfu = 2 Or Esfu = 3 Then 'cortante o flector
O b = Abs(b)
O If b > EmaxV Then EmaxV = b: EmaxB = k: EmaxT = TiempoAct
O b = (b / EmaxV) * 254 - 127 'exageración del color
O End If
O If b > 127 Then b = 127
O If b < -127 Then b = -127
O L(k, 0) = b 'color esfuerzo 2'.....
O End If
O Next k
O Else
O If Iter = 1 Then
O For k = 1 To ne 'barra.....
O L(k, 0) = 0: L(k, 3) = 0
O Next k
O End If
O End If
O
O Parar = EOF(1)
O If Iter = iterac& Then Parar = True
O DoEvents
O
O If ChReescala.value = 1 Then CalcEsc 'Recalcula la escala
O Paso2D 'Calcula las coordenadas de la pantalla CP().
O CalcDef 'Cálculo de las deformaciones de las barras
O retardo 'Realiza retardo
O If Parar Then Dib.AutoRedraw = True: TimeBar.AutoRedraw = True
O DibPan Dib, 1, True 'Realiza el dibujo en pantalla.
O BarraTiempo TimeBar, (Iter), (iterac&)
O
O Loop
O Close #1
O Close #2
O
O If Esfuerz(1).value Then 'axil
O AxilVmax.Text = EmaxV: AxilBmax.Text = EmaxB: AxilTmax.Text = EmaxT
O AxilVmin.Text = EminV: AxilBmin.Text = EminB: AxilTmin.Text = EminT
O End If
O If Esfuerz(2).value Then 'Corte
O CortVmax.Text = EmaxV: CortBmax.Text = EmaxB: CortTmax.Text = EmaxT
O End If
O If Esfuerz(3).value Then 'Flector
O CortVmax.Text = EmaxV: CortBmax.Text = EmaxB: CortTmax.Text = EmaxT
O End If
O
O b = TimeBar.ForeColor
O TimeBar.ForeColor = QBColor(14)
O TimeBar.FontName = "Arial"
O TimeBar.FontSize = 8
O TimeBar.CurrentX = 1100
O TimeBar.CurrentY = 10
O TimeBar.Print TiempoAct
O TimeBar.ForeColor = b
O
O End Sub

```

```

O
O
O
O Private Sub Paso2D()
O Dim x, y, Z, i, a, b
O 'pasa de 3d a 2d las coordenadas de axonometrías.....
O For i = 1 To np2
O     x = C(i, 1): y = C(i, 2): Z = C(i, 3)
O     axo3d2d x, y, Z, a, b
O     cp(i, 1) = a
O     cp(i, 2) = b
O Next i
O End Sub
O
O
O
O Private Sub retardo()
O Dim a As Single
O ' Realiza retardo.....
O Dim i, j
O For i = 1 To EscalaTemp - 1
O     For j = 1 To 10000
O         a = 1
O     Next j
O Next i
O End Sub
O
O
O
O Private Sub DibPan(Pct As Object, Scal As Single, Borrarr As Boolean)
O 'si el objeto es un piture box, debe ponerse al llamar: Scal=1
O 'si el objeto es la impresora, debe ponerse al llamar: Borrarr=false
O Dim n1, n2, x1, x2, y1, y2
O Dim x10, y10, x20, y20
O Dim i, j, px, py, col
O Dim n, Ntramos '1,0 2 si hay nudo intermedio
O
O px = Pct.ScaleWidth: py = Pct.ScaleHeight
O If Borrarr Then
O     x1 = -loneje * hx5: y1 = -loneje * hx1: Dib.Line (hm10, hm20)-(hm10 + Esc0 * x1,
O         hm20 - Esc0 * y1), QBColor(0)
O     x1 = loneje * hx4: y1 = -loneje * hx2: Dib.Line (hm10, hm20)-(hm10 + Esc0 * x1,
O         hm20 - Esc0 * y1), QBColor(0)
O     x1 = 0: y1 = loneje: Dib.Line (hm10, hm20)-(hm10 + Esc0 * x1,
O         hm20 - Esc0 * y1), QBColor(0)
O End If
O
O If nnd$ = "Si" Then Ntramos = 2 Else Ntramos = 1
O For i = 1 To ne
O For n = 1 To Ntramos
O     If Ntramos = 1 Then
O         n1 = L(i, 1): n2 = L(i, 2)
O         j = L(i, 3)
O     Else
O         n1 = L(i, n) 'nudo extremo
O         n2 = np + i 'nudo intermedio
O         j = L(i, -3 * (n = 1)) 'color
O     End If
O
O     If Esfuerz(1).value Then
O         col = RGB(127 - j, 127 + j, 0)
O     Else
O         col = RGB(0, 127 + j, 127 - j)
O     End If
O
O If Borrarr Then
O     x10 = CP0(n1, 1): x20 = CP0(n2, 1) 'borrado .....
O     If Not ((x10 < 0 And x20 < 0) Or (x10 > px And x20 > px)) Then
O         y10 = CP0(n1, 2): y20 = CP0(n2, 2)
O         If Not ((y10 < 0 And y20 < 0) Or (y10 > py And y20 > py)) Then
O             Dib.Line (x10, y10)-(x20, y20), 0
O         End If
O     End If
O End If
O
O     x1 = cp(n1, 1) * Scal: x2 = cp(n2, 1) * Scal 'nuevo dibujo .....
O     If Not ((x1 < 0 And x2 < 0) Or (x1 > px And x2 > px)) Then
O         y1 = cp(n1, 2) * Scal: y2 = cp(n2, 2) * Scal
O         If Not ((y1 < 0 And y2 < 0) Or (y1 > py And y2 > py)) Then
O             Pct.Line (x1, y1)-(x2, y2), col

```

```

O         End If
O     End If
O Next n
O Next i
O
O If Not (Pct Is Printer) Then
O     For i = 1 To np2
O         CP0(i, 1) = cp(i, 1) ' Guarda coordenadas pantalla en CP0(,)...
O         CP0(i, 2) = cp(i, 2)
O     Next i
O     hm10 = hm1: hm20 = hm2: Esc0 = Esc 'si se recalculó la escala.
O End If
O
O End Sub
O
O
O
O Private Sub Arrancar_Click()
O
O If Parar = True Then
O     Esfuerz(1).Enabled = False
O     Esfuerz(2).Enabled = False
O     Esfuerz(3).Enabled = False
O     TimeBar.Cls
O
O     Parar = False
O     'Arrancar.Enabled = False
O     Arrancar.Caption = "Parar"
O     Salir.Enabled = False
O     VerDatos.Enabled = False
O     ListCalc.Enabled = False
O
O     Dib.Cls
O     Dibujo '.....
O
O     'Arrancar.Enabled = True
O     Arrancar.Caption = "Arrancar"
O     Salir.Enabled = True
O     VerDatos.Enabled = True
O     ListCalc.Enabled = True
O     Esfuerz(1).Enabled = True
O     Esfuerz(2).Enabled = True
O     Esfuerz(3).Enabled = True
O Else
O     Parar = True
O End If
O
O End Sub
O Private Sub EscEsp_Change()
O Escala = Cdbl("0" + EscEsp.Text)
O End Sub
O
O Private Sub EscTmp_Change()
O EscalaTemp = Cdbl("0" + EscTmp.Text)
O End Sub
O
O
O
O Private Sub Form_Unload(Cancel As Integer)
O Menu.Show
O End Sub
O
O
O
O Private Sub ListCalc_Click()
O Dim a$
O extension$ = ListCalc.List(ListCalc.ListIndex)
O a$ = nom$ + "\PARAMETR." + extension$ 'lee los nuevos parámetros
O datosdin a$
O TiempoAct = 0
O time.Caption = "Tiempo total: " + Format$(tiempo, "#0.000 E+00") + " sg."
O If nnd$ = "Si" Then np2 = np + ne Else np2 = np
O
O 'para dibujo en pantalla posición inicial...
O LeeDat 'lee coordenadas 1ª iteración...
O TimeBar.Cls
O Paso2D 'Calcula las coordenadas de la pantalla CP().
O Dib.Cls: DoEvents: DibPan Dib, 1, False 'Realiza el dibujo en pantalla.
O

```

```

○ End Sub
○
○
○
○ Private Sub Salir_Click()
○ 'Para_Click
○ Unload Me
○ End Sub
○
○
○
○ Sub datosdin(a$)
○ Dim b$
○
○ Open a$ For Input As #1
○ Line Input #1, b$ 'Descripción datos (texto)
○ Input #1, dt 'Delta tiempo (sg)
○ Input #1, itergrb 'Relación de iteración/grabación (itergrb)
○ Input #1, b$ 'Tipo de amortiguamiento (texto)
○ Input #1, b$ 'Constante de amortiguamiento (sg-1)
○ Input #1, iterac& 'N° máximo de iteraciones (itermax)
○ Input #1, b$ 'Velocidad máxima admisible (m/s)
○ Input #1, b$ 'Si se considera el peso propio (texto)
○ Input #1, nnd$ 'Si de usa nudo intermedio (texto)
○ Close #1
○
○ tiempo = dt * iterac&
○ dt = dt * itergrb
○ iterac& = Int(iterac& / itergrb)
○ If nnd$ = "Si" Then np2 = np + ne Else np2 = np
○ End Sub
○
○ Private Sub CalcEsc()
○ Dim px, py, x, y, Z, x1, x2, y1, y2, z1, z2, xx, yy
○ Dim i
○ 'calculo de la escala para representaciøn axonometrja.....
○ '::::::::::::::::::::::::::::::::::::::::::::::::::
○ hm1 = 0: hm2 = 0: Esc = 1
○ px = Dib.ScaleWidth: py = Dib.ScaleHeight
○ x = C(1, 1): y = C(1, 2): Z = C(1, 3)
○ axo3d2d x, y, Z, x1, y1: y2 = y1: x2 = x1
○
○ For i = 1 To np2
○ x = C(i, 1): y = C(i, 2): Z = C(i, 3)
○ axo3d2d x, y, Z, xx, yy
○ If xx < x1 Then x1 = xx
○ If xx > x2 Then x2 = xx
○ If yy < y1 Then y1 = yy
○ If yy > y2 Then y2 = yy
○ Next i
○
○ xx = x2 - x1: yy = y2 - y1
○ If xx = 0 Then xx = 0.0001
○ If yy = 0 Then yy = 0.0001
○ If (xx / px) < (yy / py) Then Esc = 0.95 * py / yy Else Esc = 0.95 * px / xx 'escala
○ hm1 = (px - Esc * xx) / 2 - x1 * Esc 'centrado imagen
○ hm2 = (py - Esc * yy) / 2 - y1 * Esc
○ End Sub
○
○ Private Sub Perspec_Click()
○ 'Selección del tipo de perspectiva a realizar
○ Dim index As Integer
○ index = Perspec.ListIndex
○ Select Case index
○ Case 0: hx1 = 0.395: hx2 = 0.124: hx3 = 0.55: hx4 = 0.496: hx5 = 0.225 'DIN-5
○ Case 1: hx1 = 0.5: hx2 = 0: hx3 = 1: hx4 = 1: hx5 = 0.5 'Caballera
○ Case 2: hx1 = 0: hx2 = -1: hx3 = 0: hx4 = 0: hx5 = -1 'Planta
○ Case 3: hx1 = 0: hx2 = 0: hx3 = 1: hx4 = 1: hx5 = 0 'Alzado YZ
○ Case 4: hx1 = 0: hx2 = 0: hx3 = 1: hx4 = 0: hx5 = -1 'Alzado XZ
○ Case 5: hx1 = 0.5: hx2 = 0.5: hx3 = 1: hx4 = 0.866: hx5 = 0.866 'Isometria
○ Case 6: hx1 = 0.866: hx2 = 0.5: hx3 = 0.5: hx4 = 0.866: hx5 = 0.5 'Militar
○ Case 7: hx1 = 0.7: hx2 = 0.22: hx3 = 0.75: hx4 = 1: hx5 = 0.3 'Otra
○ End Select
○
○ If Parar = True Then 'si no está en proceso de animación
○ CalcEsc
○ Paso2D 'Calcula las coordenadas de la pantalla CP().
○ Dib.Cls: DibPan Dib, 1, False 'Realiza el dibujo en pantalla.
○ Else

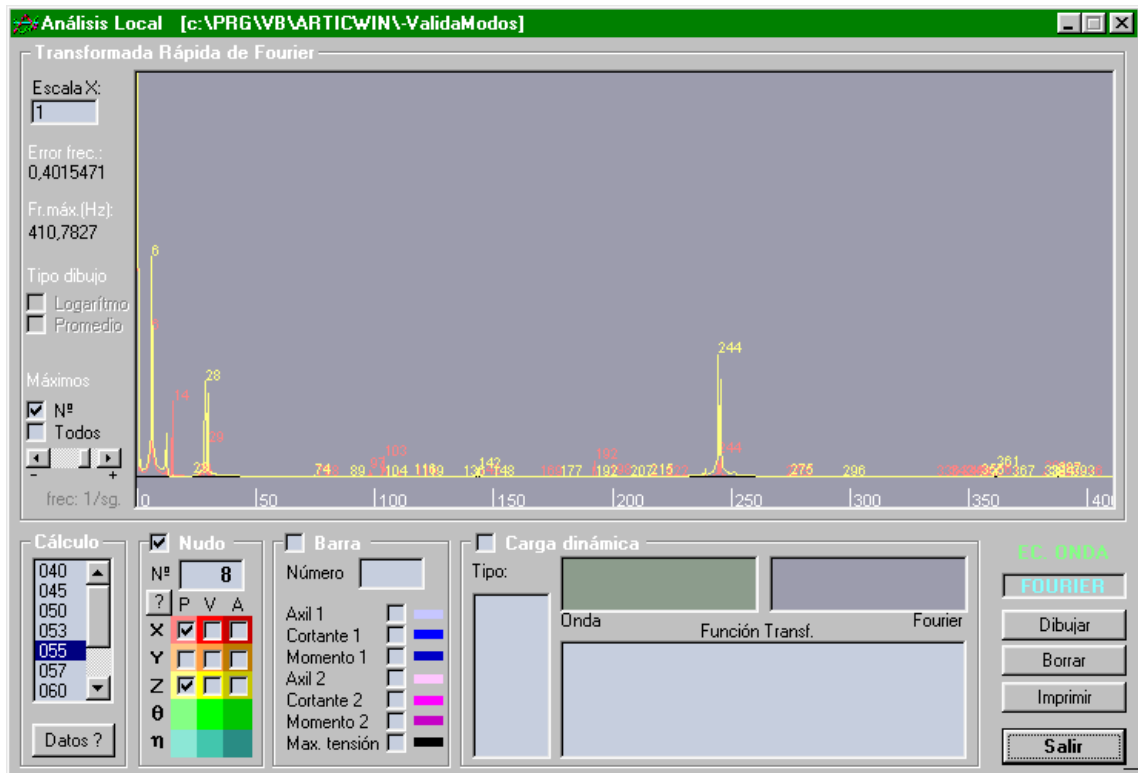
```

```

O Dib.Cls: CalcEsc
O End If
O
O End Sub
O
O Private Sub LeeDat()
O Dim i As Long, j As Long, b As Double, Existe As Boolean
O
O 'Leemos como coordenadas de referencia las del arranque del cálculo...
O If Dir(nom$ + "\RESULTC." + extension$) <> "" Then Existe = True Else Existe = False
O 'existe fichero
O If Existe Then
O If FileLen(nom$ + "\RESULTC." + extension$) > 0 Then Existe = True 'longitud >0
O Else
O Existe = False
O End If
O
O If Existe Then
O Open (nom$ + "\RESULTC." + extension$) For Binary As #1 'Len = 8
O For j = 1 To 3
O For i = 1 To np2
O Get #1, , b
O C(i, j) = b: CO(i, j) = b
O Next i
O Next j
O Close #1
O Arrancar.Enabled = True
O Arrancar.Caption = "Arrancar"
O Else 'si no hay cálculo efectuado
O nnd$ = "No": np2 = np
O DiscoX "COORDENA.DAT", "LnoRedim", "S3A", iii, C(), ddd, xxx, Str$(ne)
O DiscoX "COORDENA.DAT", "LnoRedim", "S3A", iii, CO(), ddd, xxx, Str$(ne)
O Arrancar.Enabled = False
O End If
O End Sub
O
O
O
O Private Sub LeeGral()
O 'leemos los parámetros generales de la estructura y dimensionamos las matricesl11
O DiscoX "datos.dat", "L", "-", iii, sss, ddd, xxx, ttt$
O ReDim L(ne, 3), C(np + ne, 3), CO(np + ne, 3), cp(np + ne, 2), CP0(np + ne, 2)
O ' Dimensionamos al valor máximo, por si algún cálculo incluya nudo intermedio
O ' barras 'puntos 'pts.grabad'coord.inicio'crd.pant.'crd.pa.itera.anterior
O ' 0-long.inic.; 1-nudo inic.; 2-nudo final; 3-tipo
O DiscoX "BARRAS.DAT", "L", "I", L(), sss, ddd, xxx, ttt$
O End Sub
O
O
O
O Sub EscalCol(obj As Object, num As Integer)
O 'Escala gráfica de colores.....
O 'obj - donde dibujamos
O 'num - número de escala a usar
O 'FillStyle = 0 'continuo
O Dim col, Dx, Dy, n, i
O
O n = 25 'n° tramos escala colores
O col = 255 / n
O Dx = Escal.Width: Dy = Escal.Height / n
O For i = 0 To n - 1
O Select Case num
O Case 1:obj.Line (0, Dy * i)-(Dx, Dy * (i + 1)), RGB(0 + col * i, 255 - col * i, 0),BF
O Case 2:obj.Line (0, Dy * i)-(Dx, Dy * (i + 1)), RGB(0, 255 - col * i, 0 + col * i),BF
O End Select
O Next i
O
O End Sub
O

```

## 7.3.3 MÓDULO DE ANÁLISIS LOCAL



```

○ Option Explicit
○ DefDbl A-H
○ DefInt I-N
○ DefDbl O-Z
○
○ Dim Matriz() As Double, np As Long, ne As Long, npa As Long
○ Dim nnd$, dt As Single, TiempoFF As Single, xxml As Long, iterac As Long
○ Dim factmatr As Single 'factor para ver cuantos máximos se dibujan
○ Dim Relx As Single
○ Dim maxOndas(15) As Single, dimensN As Integer, dimensB As Integer
○ Dim frxFFT(15) As Single
○ 'carga...
○ Dim Carga As Integer, DatCar() As Single
○ 'nudo....
○ Dim Nudo As Long, DatNud() As Single
○ 'barra...
○ Dim barra As Long, DatBar() As Single
○ Dim CalcOld As String, CargaOld, NudoOld, BarraOld 'dibujados en llamada anterior
○
○ Private Sub Borrar_Click()
○ Dim i As Integer
○ PictO.Cls
○ PictF.Cls
○
○ For i = 1 To 5
○ MxN(i) = "0"
○ Next i
○ For i = 1 To 6
○ MxB(i) = "0"
○ Next i
○
○ PictOC.Cls
○ PictFC.Cls
○ PictFT.Cls
○ End Sub
○
○ Private Sub CheckBarra_Click()
○ DatosBarra.Enabled = (CheckBarra.value = 1)
○ End Sub

```



```

○
○
○ Private Sub CheckCarga_Click()
○ DatosCarga.Enabled = (CheckCarga.value = 1)
○ End Sub
○
○
○ Private Sub CheckNudo_Click()
○ DatosNudo.Enabled = (CheckNudo.value = 1)
○ End Sub
○
○
○ Private Sub Command1_Click()
○ MsgBox "Los desplazamientos y ángulos se dan relativos a la posición de inicio." +
○ vbCrLf + "Las velocidades y aceleraciones son valores absolutos.",
○ vbOKOnly, "Nota:"
○ End Sub
○
○
○ Private Sub Dibujar_Click()
○ Dibu 1
○ End Sub
○
○
○ Private Sub Form_Unload(Cancel As Integer)
○ Menu.Show
○ End Sub
○
○
○
○ Private Sub HSmax_Change()
○ PictF.Cls
○ Dibu 1
○ End Sub
○
○
○ Private Sub Imprimir_Click()
○ Printer.FontName = "Arial"
○ Printer.FontSize = 8
○ Printer.Print "Estructura: " + nom$
○ Printer.Print "Cálculo: " + extension$
○ Dibu 2
○ End Sub
○
○
○ Private Sub LbFourier_Click()
○ LbOnda.BorderStyle = 0
○ LbFourier.BorderStyle = 1
○ FrOnda.Visible = False
○ FrFourier.Visible = True
○ End Sub
○
○
○ Private Sub LbOnda_Click()
○ LbOnda.BorderStyle = 1
○ LbFourier.BorderStyle = 0
○ FrOnda.Visible = True
○ FrFourier.Visible = False
○ End Sub
○
○
○ Private Sub ListCalc_Click()
○ extension$ = ListCalc.List(ListCalc.ListIndex)
○ extensionDat$ = extension$
○ LeeGral
○ End Sub
○
○
○ Private Sub Salir_Click()
○ Unload Me
○ End Sub
○
○
○ Private Sub Form_Load()
○ Dim archiv$
○ CalcOld = "": CargaOld = 0: NudoOld = 0: BarraOld = 0 'dibujados en llamada anterior
○ archiv$ = nom$ + "\parametr.*"
○ ListExt ListCalc, archiv$
○ AnLocal.Caption = "Análisis Local" + " [" + nom$ + "]"
○
○
○ LbOnda_Click
○ If ListCalc.Enabled = False Then

```

```

O   CheckNudo.Enabled = False
O   CheckBarra.Enabled = False
O   CheckCarga.Enabled = False
O   VerDatos.Enabled = False
O   Dibujar.Enabled = False
O   Borrar.Enabled = False
O   Imprimir.Enabled = False
O End If
O End Sub
O
O
O Private Sub Numnud_Change()
O Dim NPP As Long, i As Integer
O NPP = Val(Numnud.Text)
O If NPP > (np + npa) Or NPP <= 0 Then
O   Numnud.Text = " "
O   NPP = 0
O Else
O   If NPP > np Then
O     For i = 10 To 15
O       ChNud(i).value = 0
O       ChNud(i).Visible = True
O     Next i
O   Else
O     For i = 10 To 15
O       ChNud(i).Visible = False
O     Next i
O   End If
O End If
O End Sub
O
O
O Sub FFTp(Matrx() As Single, fpmatrx() As Single, fpmatrx2() As Single, isign%, result%,
O Res%)
O 'realiza FFT de Matr() sin cambiarla, y da el resultado en fpmatrx (módulos)
O '(o parte real) y en fpmatrx2 (fase)(o parte imaginaria)
O 'el tamaño de fpmatrx() y fpmatrx2() será la mitad de matr()
O 'isign%= .. Transformada normal (0) o inversa (1)
O 'result%=.. Si se desea obtener módulo y fase (0) o parte real e imaginaria(1)
O 'si res%=0 no llena la matriz fpmatrx2, por lo que se puede dar una matriz mínima
O 'fpmatrx2(1)
O Dim npts As Long, i As Long
O Dim basura1() As Single
O Dim basura2() As Single
O npts = UBound(Matrx)
O ReDim basura1(npts)
O ReDim basura2(npts)
O For i = 1 To npts
O   basura1(i) = Matrx(i)
O Next i
O Call FFT(basura1(), basura2(), npts, isign%, result%) 'FAST FOURIER
O For i = 1 To npts / 2
O   fpmatrx(i) = basura1(i)
O   If Res <> 0 Then fpmatrx2(i) = basura2(i)
O Next i
O End Sub
O
O
O Sub FFT(x() As Single, y() As Single, nxm As Long, isign%, result%)
O Dim nexp, i, j, k, L, jh, ii, nxml, lx, nblock, lblock, lbhalf, iblock, istart
O Dim intgr()
O Dim flx, fsign, fk, v, wkr, wki, qr, qi, holdr, holdi, x0, y0
O 'npts= .. Número de puntos disponibles
O 'isign%= .. Transformada normal (0) o inversa (1)
O 'result%=.. Si se desea obtener módulo y fase (0) o parte real e imagin.(1)
O 'nexp= .. Valor mínimo para que se cumpla 2^NEXP >= NPTS
O
O nxm = UBound(x) 'límite superior de la matriz de datos, debe ser potencia de 2
O nexp = 0
O Do
O   nexp = nexp + 1
O   nxml = 2 ^ nexp
O Loop Until nxml >= nxm
O If UBound(y) <> nxm Then Stop 'si las matrices X,Y son de distinto tamaño
O
O ReDim intgr(nexp + 2)
O For i = 1 To nexp
O   intgr(i) = 2 ^ (nexp - i)
O Next i
O
O

```

```

O ' -----
O '
O '          FAST FOURIER TRANSFORM PROGRAM
O '          ISIGN%=0 ;FORWARD TRANSFORM
O '          ISIGN%=1 ;REVERSE TRANSFORM
O '          NEXP=POWER OF 2 EQUAL TO NUMBER OF POINTS 2^10 = 1024 POINTS
O '          INPUT ARRAY X(I)  IN THE TIME DOMAIN
O ' -----
O '
O lx = 2 ^ nexp
O flx = lx
O fsign = -1#
O If isign% > 0 Then fsign = 1#
O
O For L = 1 To nexp
O   nblock = 2 ^ (L - 1)
O   lblock = lx / nblock
O   lbhalf = lblock / 2
O   k = 0
O
O   For iblock = 1 To nblock
O     fk = k
O     v = fsign * 2# * pi * fk / flx
O     wkr = Cos(v)
O     wki = Sin(v)
O     istart = lblock * (iblock - 1)
O
O     For i = 1 To lbhalf
O       j = istart + i
O       jh = j + lbhalf
O       qr = x(jh) * wkr - y(jh) * wki
O       qi = x(jh) * wki + y(jh) * wkr
O       x(jh) = x(j) - qr
O       y(jh) = y(j) - qi
O       x(j) = x(j) + qr
O       y(j) = y(j) + qi
O     Next i
O
O     For i = 2 To nexp
O       ii = i
O       If k < intgr(i) Then Exit For
O       k = k - intgr(i)
O     Next i
O
O     k = k + intgr(i)
O   Next iblock
O Next L
O
O k = 0
O For j = 1 To lx
O   If k >= j Then
O     holdr = x(j)
O     holdi = y(j)
O     x(j) = x(k + 1)
O     y(j) = y(k + 1)
O     x(k + 1) = holdr
O     y(k + 1) = holdi
O   End If
O
O   For i = 1 To nexp
O     ii = i '5520
O     If k < intgr(i) Then Exit For
O     k = k - intgr(i)
O   Next i
O
O   k = k + intgr(ii)
O Next j
O
O If fsign > 0 Then 'si es la transformada inversa..
O   For i = 1 To lx
O     x(i) = x(i) / flx
O     y(i) = y(i) / flx
O   Next i
O End If
O
O 'calcula módulos y fases.....
O If result% = 0 Then
O   For i = 1 To nxml / 2
O     x0 = x(i): y0 = y(i)

```

```

O      x(i) = Sqr(x0 * x0 + y0 * y0)           'módulo
O      If x0 <> 0 Then                          'ángulo de fase
O          y(i) = Atn(y0 / x0)
O          If x0 < 0 Then y(i) = y(i) + pi * Sgn(y0)
O          Else
O              y(i) = pi / 2 * Sgn(y0)
O          End If
O      Next i
O  End If
O
O  End Sub
O
O
O  Private Sub numbar_Change()
O  Dim NPP As Long, i As Integer
O  NPP = Val(Numbar.Text)
O  If NPP > ne Or NPP <= 0 Then
O      Numbar.Text = " "
O      NPP = 0
O  End If
O  End Sub
O
O
O
O  Sub LeeNud(Nudo As Long, DatNud() As Single)
O  Dim a$, b As Double, n As Long, i As Long, j As Long
O  Dim np2 As Long, Tot As Single
O  Dim x, y, Z, Te, Et
O  If nnd$ = "Si" Then np2 = np + ne Else np2 = np
O  'np2 = np + npa
O  Tot = np2 * 3: If (nnd$ = "Si") Then Tot = np2 * 3 + (np2 - np) * 2
O  'lee posiciones del nudo en el arranque (solo 1ª posición)
O  'para valores de desplazamientos y ángulos relativos a los de partida
O  a$ = nom$ + "\resultc." + extension$
O  Open a$ For Binary As #1 Len = 8
O      For j = 1 To 3 'X Y Z
O          For i = 1 To np2
O              Get #1, , b
O              If i = Nudo Then
O                  Select Case j
O                      Case 1: x = b 'X.....
O                      Case 2: y = b 'Y.....
O                      Case 3: Z = b 'Z.....
O                  End Select
O              End If
O          Next i
O      Next j
O      If (nnd$ = "Si") Then 'Barras con 3 nudos.....
O          'Lee valores angulares de los nudos ampliados
O          For j = 4 To 5 'T E
O              For i = np + 1 To np2
O                  Get #1, , b
O                  If i = Nudo Then
O                      Select Case j
O                          Case 4: Te = b 'theta....
O                          Case 5: Et = b 'eta.....
O                      End Select
O                  End If
O              Next i
O          Next j
O      End If
O  Close #1
O
O  'Valores de posición.....
O  a$ = nom$ + "\resultc." + extension$
O  Open a$ For Binary As #1 Len = 8
O  For n = 1 To iterac 'para el número total de iteraciones
O  BarraTiempo PrgNudo, (n), (iterac): DoEvents
O      For j = 1 To 3 'X Y Z
O          For i = 1 To np2
O              Get #1, , b
O              If i = Nudo Then
O                  Select Case j
O                      Case 1: DatNud(1, n) = b - x 'despl.X.....
O                      Case 2: DatNud(2, n) = b - y 'despl.Y.....
O                      Case 3: DatNud(3, n) = b - Z 'despl.Z.....
O                  End Select
O              End If
O          Next i
O      Next j

```

```

O     Next j
O     If (nnd$ = "Si") Then 'Barras con 3 nudos.....
O     'Lee valores angulares de los nudos ampliados
O     For j = 4 To 5 'T E
O     For i = np + 1 To np2
O     Get #1, , b
O     If i = Nudo Then
O     Select Case j
O     Case 4: DatNud(4, n) = b - Te 'despl.theta....
O     Case 5: DatNud(5, n) = b - Et 'despl.eta.....
O     End Select
O     End If
O     Next i
O     Next j
O     End If
O Next n
O Close #1
O End Sub
O
O
O
O Sub LeeBar(barra As Long, DatBar() As Single)
O Dim a$, n As Long, i As Long, k As Long
O Dim b1 As Double, b2 As Double, b3 As Double, b4 As Double, b5 As Double, b6 As Double
O If nnd$ = "Si" Then ' CÁLCULO CON NUDOS INTERMEDIOS.....
O a$ = nom$ + "\axil." + extension$: Open a$ For Binary As #1 Len = 8
O a$ = nom$ + "\cort." + extension$: Open a$ For Binary As #2 Len = 8
O a$ = nom$ + "\flec." + extension$: Open a$ For Binary As #3 Len = 8
O For i = 1 To iterac 'iteración...
O BarraTiempo PrgBarra, (i), (iterac): DoEvents
O For k = 1 To ne 'punto....
O Get #1, , b1 'axil 1
O Get #2, , b2 'cortante 1
O Get #3, , b3 'flector 1
O Get #1, , b4 'axil 2
O Get #2, , b5 'cortante 2
O Get #3, , b6 'flector 2
O If k = barra Then
O DatBar(1, i) = b1
O DatBar(2, i) = b2
O DatBar(3, i) = b3
O DatBar(4, i) = b4
O DatBar(5, i) = b5
O DatBar(6, i) = b6
O End If
O Next k
O Next i
O Close
O Else ' Lectura de los datos de las barras...(2 nudos).....
O a$ = nom$ + "\axil." + extension$: Open a$ For Binary As #1 Len = 8
O For i = 1 To iterac
O For k = 1 To ne
O Get #1, , b1 'axil
O If k = barra Then
O DatBar(1, i) = b1
O End If
O Next k
O Next i
O Close
O End If
O End Sub
O
O
O Function frec(i As Long, TiempoFF As Single) As Single
O 'Da la frecuencia angular (f*2*pi) para el punto i de la matriz de la fft
O 'tiempoff= tiempo de la matriz original
O frec = (i - 1) / TiempoFF ' * 2 * pi 'frecuencia angular
O End Function
O
O
O Sub dibmatrmx(ff2() As Single, Pict As Object, Scl2 As Single, Colo As Single, Eje As
O Single, Fr As Single, tiempo As Single, Posix As Integer, valor As
O Integer)
O 'SHARED factmatr
O '-----
O ' ff2() - Matriz a dibujar, numerando los máximos (debe ser la mitad si es de
O FFT, si no sale simétrica)
O ' Pict - Objeto en el que se va a dibujar
O ' Scl2 - corrección de escala (<=1)

```

```

O ' Colo      - color de dibujo
O ' Eje       - si es >0 dibuja el eje X con el color Eje
O ' Fr        - dato a devolver:frecuencia del punto de valor máximo
O ' Tiempo    - tiempo de duración de la matriz original total (no la mitad)
O '           - si tiempo=0 entonces en el máx. pone valor de coordenada Y (no la X)
O ' Posix     - posición eje x (1=parte inferior ventana, 2=centrado)
O '           - Si el eje se desea en la parte inferior, se debe garantizar
O '           - que la matriz solo contiene datos positivos
O ' Valor     - si es <>0 calcula y numera los picos máximos.
O '-----
O Dim Scalx As Single, Scaly As Single
O Dim Ox As Single, Oy As Single, Dx As Single, Dy As Single
O Dim x As Single, y As Single, pto As Single, C As Single
O Dim Max() As Single, Maxi As Single, a As Single, b As Single
O Dim DibMax As Boolean
O
O Dim i As Long, j As Long, i0 As Long, i2 As Long, Tot As Long
O Dim om As Long, cmx As Integer, fMax As Single
O
O
O Tot = UBound(ff2)
O Ox = Pict.ScaleLeft: Oy = Pict.ScaleTop
O Dx = (Pict.ScaleWidth - Ox) * Relx: Dy = (Pict.ScaleHeight * 0.93) - Oy
O Oy = Oy + Dy / Posix 'origen de coordenadas
O
O Tot = UBound(ff2)
O ReDim Max(Tot / 2) 'máximos relativos del dibujo de la fft
O cmx = 0 'número de máximos relativos del análisis de pos.
O Scalx = 0 'escalas de representación del análisis
O Scaly = 0 'escalas de representación del análisis
O Fr = 0 'si se devuelve cero es que no hay máximos
O fMax = 1.5 - (HSmax.value / HSmax.Max * 0.5) 'de 1 a 1.5
O
O 'para calcular la escala.....
O Maxi = ff2(1)
O For i = 1 To Tot
O If Abs(ff2(i)) > Abs(Maxi) Then Maxi = ff2(i): Fr = i
O Next i
O If Maxi = 0 Then Maxi = 1 'para que no casque
O Scalx = Dx / (Tot - 1) 'escalas de representación
O Scaly = Scl2 * Dy / (Posix * Abs(Maxi))
O
O If Eje >= 0 Then Pict.Line (Ox, Oy)-(Ox + Dx, Oy), Eje
O 'dibuja los puntos.....
O cmx = 0
O For i = 1 To Tot
O pto = ff2(i)
O x = Ox + (i - 1) * Scalx - 1: y = Oy - pto * Scaly
O If i = 1 Then Pict.PSet (x, y), Colo Else Pict.Line -(x, y), Colo 'dibuja
O líneas
O 'máximos relativos
O If valor <> 0 Then
O C = Sgn(pto)
O If i > 1 Then a = (pto - ff2(i - 1)) * C Else a = 1
O If i < Tot Then b = ((pto - ff2(i + 1)) * C) Else b = 1
O If (a > 0) And (b > 0) Then cmx = cmx + 1: Max(cmx) = i
O End If
O Next i
O
O 'texto máximos.....
O 'PictF.FontName = "Courier New"
O PictF.FontName = "Arial"
O PictF.FontSize = 7
O PictF.ForeColor = Colo
O
O If valor <> 0 Then
O 'numeración de máximos.....
O Debug.Print "-----"
O For j = 1 To cmx
O i = Max(j): pto = ff2(i) 'coord.x,y
O If ChTodos.value = 1 Then 'si se marca que se dibujen todos
O DibMax = True
O Else
O DibMax = False
O Select Case cmx
O Case Is < 25: DibMax = True ' dibuja todos
O Case 26 To 200
O i0 = 0: i2 = 0
O 'If j > 1 Then i0 = Max(j - 1)

```

```

O      'If j < cmx Then i2 = Max(j + 1)
O      'If pto > ff2(i0) And pto > ff2(i2) Then: DibMax = True 'dibuja si es mayor
      que los 2 maximos adyacentes
O      If j > 1 Then i0 = i - 1
O      If j < cmx Then i2 = i + 1
O      If pto > ff2(i0) * fMax And pto > ff2(i2) * fMax Then: DibMax = True 'dibuja
      si sobreesale sobre los 2 valores adyacentes
O      Case Is > 200
O      i0 = 0: i2 = 0
O      If j > 1 Then i0 = Max(j - 1)
O      If j < cmx Then i2 = Max(j + 1)
O      If pto > ff2(i0) * fMax And pto > ff2(i2) * fMax Then DibMax = True
O      End Select
O      End If
O
O      If DibMax = True Then 'dibuja el texto del máximo
O      x = Ox + (i - 1) * Scalx - 1: y = Oy - pto * Scaly
O      om = Int(frec(i, tiempo)) 'frecuencia angular
O      Pict.CurrentX = x - 30
O      Pict.CurrentY = y - 140
O      Pict.Print (om)
O      Debug.Print "n°mx="; j, i, "w="; om
O      End If
O      Next j
O      End If
O      End Sub
O
O
O      Sub dibmatr(mat() As Single, Pict As Object, Scl As Single, Colo As Single, Eje As
      Single, Max As Single)
O      '-----
O      ' Dibuja las matriz mat(), en el objeto Pict, con el color Colo
O      ' devuelve el valor del máximo absoluto en Max
O      ' el dibujo queda centrado con el eje X a la mitad de la altura.
O      ' Scl - factor escala Y. El valor 1 ajusta verticalmente al máximo.
O      ' Eje - si es >0 dibuja el eje X con el color Eje
O      '-----
O      Dim Iter As Long, Scalx As Single, Scaly As Single
O      Dim Ox As Single, Oy As Single, Dx As Single, Dy As Single
O      Dim i As Long
O      Dim aa As Single, bb As Single
O
O      Iter = UBound(mat)
O      Ox = Pict.ScaleLeft: Oy = Pict.ScaleTop
O      Dx = Pict.ScaleWidth - Ox: Dy = Pict.ScaleHeight - Oy
O      Oy = Oy + Dy / 2 'eje centrado
O
O      aa = CSng(EscalaH.Text)
O      bb = CSng(MaxV.Text)
O
O      'calculamos los valores máximos para escalarlos.....
O      Max = mat(1)
O      For i = 1 To Iter
O      If Abs(mat(i)) > Abs(Max) Then Max = mat(i)
O      Next i
O      If Max = 0 Then Max = 1 'para que no casque
O      Scalx = Dx / (Iter - 1) * aa 'escalas de representación
O      Scaly = Scl * Dy / (2 * Abs(Max))
O      If bb <> 0 Then Scaly = Dy / (2 * bb)
O
O      'dibujo.....
O      If Eje >= 0 Then Pict.Line (Ox, Oy)-(Ox + Dx, Oy), Eje
O      For i = 1 To Iter
O      Pict.PSet (Ox + (i - 1) * Scalx, Oy - mat(i) * Scaly), Colo
O      Next i
O      End Sub
O      Sub dibescalaV(Pict As Object, Colo As Long, MxV As Single)
O      Dim a1 As Long, wmin As Single, wmax As Single, spac As Single, spac2 As Single, i As
      Long, aa$, Ax As Single
O      Dim numdiv As Single, py As Single, Oy, Scaly As Single, pox As Single, poy As Single,
      poy2 As Single
O
O      a1 = Pict.ScaleWidth
O      wmin = 0
O      wmax = MxV
O      'dibujo escala.....
O      spac = 1000 ' m,kg,rad
O      Pict.FontName = "Arial"

```

```

O Pict.FontSize = 7
O numdiv = 8
O spac = wmax / numdiv
O Pict.ForeColor = vbBlack
O py = Pict.ScaleHeight / 2 'x2% -x1%
O Oy = py 'x1%
O Scaly = py / numdiv
O
O For i = 1 To numdiv
O   poy = Int(Oy - i * Scaly - 1)
O   poy2 = Int(Oy + i * Scaly - 1)
O   pox = (0) ' * Scl
O   Pict.Line (pox, poy)-(350, poy), Colo
O   Pict.Line (pox, poy2)-(350, poy2), Colo
O   'If poy > 0 Then
O     Ax = i / numdiv * (wmax - wmin)
O     Select Case Ax
O       Case Is > 99999: aa$ = LTrim$(Format$(Ax, "0.0e+00"))
O       Case Is > 999.9: aa$ = LTrim$(Format$(Ax, "00000"))
O       Case Is > 99.99: aa$ = LTrim$(Format$(Ax, "000.0"))
O       Case Is > 9.999: aa$ = LTrim$(Format$(Ax, "00.00"))
O       Case Is > 0.01: aa$ = LTrim$(Format$(Ax, "0.000"))
O       Case Is <= 0.01: aa$ = LTrim$(Format$(Ax, "0.0e+00"))
O     End Select
O     Pict.CurrentX = pox + 20
O     Pict.CurrentY = poy
O     Pict.Print aa$
O     Pict.CurrentX = pox + 20
O     Pict.CurrentY = poy2
O     Pict.Print "-" + aa$
O   'End If
O   '.....
O Next i
O
O End Sub
O
O Sub dibescalaT(Pict As Object, Colo As Long)
O Dim a1 As Long, Ax As Single, wmin As Single, wmax As Single, spac As Single, spac2 As
O   Single, i As Long, aa$
O Dim numdiv As Single, px As Single, Ox, Scalx As Single, pox As Single, poy As Single
O
O a1 = Pict.ScaleWidth
O wmin = 0
O wmax = TiempoFF / CSng(EscalaH.Text)
O 'dibujo escala.....
O spac = 64 'sg.
O Pict.FontName = "Arial"
O Pict.FontSize = 7
O numdiv = 15
O spac = wmax / numdiv
O Pict.ForeColor = vbBlack
O px = Pict.ScaleWidth 'x2% -x1%
O Ox = 0 'x1%
O Scalx = px / numdiv
O
O For i = 1 To numdiv
O   pox = Int(Ox + i * Scalx - 1)
O   poy = (Pict.Height - 240) ' * SclY
O   Pict.Line (pox, poy + 10)-(pox, poy + 180), Colo
O   Ax = i / numdiv * (wmax - wmin)
O   Select Case Ax
O     Case Is > 9999: aa$ = LTrim$(Format$(Ax, "0.0e+00"))
O     Case Is > 100: aa$ = LTrim$(Format$(Ax, "###0.0"))
O     Case Is > 0.01: aa$ = LTrim$(Format$(Ax, "#0.000"))
O     Case Is <= 0.01: aa$ = LTrim$(Format$(Ax, "0.0e+00"))
O   End Select
O   Pict.CurrentX = pox + 50
O   Pict.CurrentY = poy + 20
O   Pict.Print aa$
O Next i
O End Sub
O Sub dibescala(Pict As Object, Colo As Long, SclY As Single)
O 'SHARED iterac, tiempoFF
O Dim a1 As Long, wmin As Single, wmax As Single, spac As Single, i As Long, aa$
O Dim numdiv As Single, px As Single, Ox, Scalx As Single, pox As Single, poy As Single
O
O a1 = xxml / 2
O wmin = frec(1, TiempoFF)
O wmax = frec(a1, TiempoFF) / Relx

```



```

O
O 'dibujo escala.....
O spac = 0
O Pict.FontName = "Arial"
O Pict.FontSize = 8
O Do
O   'IF LEFT$(LTRIM$(STR$(spac)), 1) = "1" THEN a = 5 ELSE a = 2
O   'a = 2
O   'spac = INT(spac * a)
O   spac = spac + 50
O   numdiv = wmax / spac
O Loop Until numdiv < 19
O Pict.ForeColor = Colo
O px = Pict.ScaleWidth 'x2% -x1%
O Ox = 0 'x1%
O Scalx = px / numdiv
O
O For i = 0 To numdiv
O   pox = Int(Ox + i * Scalx - 1)
O   poy = (Pict.Height - 240) * SclY
O   Pict.Line (pox, poy + 10)-(pox, poy + 180), Colo
O   If pox < (px) Then
O     aa$ = LTrim$(Str$(Int(i * (wmax - wmin) / numdiv + 0.5)))
O     Pict.CurrentX = pox + 50
O     Pict.CurrentY = poy + 20
O     Pict.Print aa$
O   End If
O Next i
O '.....
O End Sub
O
O
O Sub CalcOndas(Psc() As Single, Vel() As Single, Ace() As Single, iterac As Long, dt As
O   Single)
O '-----
O ' Calcula las curvas de velocidad (y aceleración).....
O '-----
O Dim i As Long, vell As Single, vel2 As Single, velo As Single, Acel As Single
O For i = 2 To iterac - 1
O   vell = (Psc(i) - Psc(i - 1)) / dt
O   vel2 = (Psc(i + 1) - Psc(i)) / dt
O   velo = 0.5 * (vell + vel2): Vel(i) = velo
O   Acel = (vel2 - vell) / dt: Ace(i) = Acel
O Next i
O Vel(1) = Vel(2): Vel(iterac) = Vel(iterac - 1)
O Ace(1) = Ace(2): Ace(iterac) = Ace(iterac - 1)
O End Sub
O
O
O Sub promediar(ff(), dd(), navg)
O Dim xml As Long, NHALF As Long, i As Long, ILO As Long, IHI As Long, Sum, j As Long,
O   Count As Long
O '-----
O '           Suavizado mediante un promedio de valores
O '-----
O 'la matriz ff() debe ser de doble de tamaño que la ff2()
O 'navg= n° de puntos entre los que se promedia (1,3,5,7 ó 9)
O navg = 3
O xml = UBound(dd)
O NHALF = Int(navg / 2)
O
O If navg <> 1 Then
O   For i = 1 To xml
O     ILO = i - NHALF
O     IHI = i + NHALF 'nota: tenía un 1 en lugar de i
O     Sum = 0!
O     Count = 0!
O     For j = ILO To IHI
O       If (j > 0) And (j <= xml) Then: Sum = Sum + ff(j): Count = Count + 1!
O     Next j
O     If Count <> 0 Then dd(i) = Sum / Count
O   Next i
O Else ' si no promedia.....
O   For i = 1 To xml
O     dd(i) = ff(i)
O   Next i
O End If
O

```

```

O End Sub
O
O
O
O Sub suavlog(ff2())
O Dim xml As Long, i As Long
O '-----
O ' Suavizado logarítmico
O '-----
O xml = UBound(ff2)
O
O For i = 1 To xml
O Select Case ff2(i)
O Case Is > 0: ff2(i) = Log(ff2(i) + 1)
O Case Is < 0: ff2(i) = -Log(-ff2(i) + 1)
O End Select
O Next i
O End Sub
O
O
O
O Private Sub LeeGral()
O 'lectura de los datos generales
O Dim a$, nexp As Integer
O Dim axil$, ppropio$, itermax As Long, itergrb As Integer, amortt$
O Dim alambda As Single, cont$, extension2$, comentario$
O Dim GrPos$, GrVel$, GrAxi$, GrCor$, GrFle$, tiempo As Single
O Dim velmax As Single
O If (Dir(nom$ + "\parametr." + extension$) = "") Then
O CheckNudo.Enabled = False
O CheckBarra.Enabled = False
O CheckCarga.Enabled = False
O VerDatos.Enabled = False
O Dibujar.Enabled = False
O Borrar.Enabled = False
O Imprimir.Enabled = False
O Exit Sub
O End If
O
O If (Dir(nom$ + "\resultc." + extension$) = "") Then
O Dibujar.Enabled = False: Imprimir.Enabled = False
O Else
O CheckNudo.Enabled = True
O CheckBarra.Enabled = True
O CheckCarga.Enabled = True
O VerDatos.Enabled = True
O Dibujar.Enabled = True
O Borrar.Enabled = True
O Imprimir.Enabled = True
O End If
O
O Open nom$ + "\parametr." + extension$ For Input As #1
O
O Line Input #1, a$ 'Descripción datos (texto)
O Input #1, dt 'Delta tiempo (sg)
O Input #1, itergrb 'Relación de iteración/grabación (itergrb)
O Input #1, amortt$ 'Tipo de amortiguamiento (texto)
O Input #1, alambda 'Constante de amortiguamiento (sg-1)
O Input #1, itermax 'N° máximo de iteraciones (itermax)
O Input #1, a$ 'Velocidad máxima admisible (m/s)
O Input #1, ppropio$ 'Si se considera el peso propio (texto)
O Input #1, nnd$ 'Si de usa nudo intermedio (texto)
O Input #1, extension2$ 'Extensión del cálculo anterior (texto)
O Input #1, cont$ 'Si es continuación cálculo anterior (texto)
O Input #1, axil$ 'Si Calcula corrección por axil (texto)
O Line Input #1, a$ 'Breve texto descriptivo calculo (texto)
O
O Input #1, GrPos$ 'Si se graban posiciones (texto)
O Input #1, GrVel$ 'Si se graban velocidades (texto)
O Input #1, GrAxi$ 'Si se graban axiles (texto)
O Input #1, GrCor$ 'Si se graban cortantes (texto)
O Input #1, GrFle$ 'Si se graban flectores (texto)
O Close #1
O
O tiempo = dt * itermax
O dt = dt * itergrb
O iterac = itermax / itergrb
O
O nexp = 0

```

```

O Do
O   nexp = nexp + 1
O   xxml = 2 ^ nexp
O Loop Until xxml >= iterac      'n° de iteraciones puede ser cualquiera
O TiempoFF = tiempo / iterac * xxml 'tiempo de la fft
O
O ReDim Matriz(xxml)
O
O 'Datos pantalla.....
O LabDT.Caption = CStr(dt)
O LabTT.Caption = CStr(TiempoFF)
O LabFM.Caption = CStr(frec(xxml / 2, TiempoFF))
O LabEF.Caption = CStr(1 / TiempoFF)
O TxtEscX.Text = "1": Relx = 1#
O If nnd$ = "No" Then
O   ChBar(2).Visible = False 'cort1
O   ChBar(3).Visible = False 'mome1
O   ChBar(4).Visible = False 'axil2
O   ChBar(5).Visible = False 'cort2
O   ChBar(6).Visible = False 'mome2
O   ChNud(10).Visible = False '
O   ChNud(11).Visible = False 'Theta
O   ChNud(12).Visible = False '
O   ChNud(13).Visible = False '
O   ChNud(14).Visible = False 'Eta
O   ChNud(15).Visible = False '
O End If
O
O a$ = nom$ + "\datos.dat"
O Open a$ For Input As #1
O   Input #1, np
O   Input #1, ne
O   'Input #1, nea
O Close #1
O If nnd$ = "Si" Then npa = ne Else npa = 0
O
O 'carga.....
O ReDim DatCar(xxml)
O
O 'nudo.....
O If nnd$ = "Si" Then
O   dimensN = 5 'x,y,z,t,e
O   Else
O   dimensN = 3 'x,y,z
O End If
O ReDim DatNud(dimensN, xxml)
O
O 'barra.....
O If nnd$ = "Si" Then
O   dimensB = 6 'ax1,cor1,mom1,ax2,cor2,mom2
O   Else
O   dimensB = 1 'axil
O End If
O ReDim DatBar(dimensB, xxml)
O
O End Sub
O
O Private Sub TxtEscX_LostFocus()
O If TxtEscX.Text = "" Then TxtEscX.Text = "1"
O Relx = TxtEscX.Text
O If Relx < 1 Then TxtEscX.Text = 1
O End Sub
O
O Private Sub VerDatos_Click()
O 'Load DatoDin
O DatoDin.Show
O End Sub
O
O
O Private Sub Dibujado_Click()
O Private Sub Dibujado_Click()
O Dim n As Integer, i As Integer, j As Long, k As Integer, kkk As Integer
O Dim Mfft() As Single
O Dim kk() As Single, Maxi As Single, Frmax As Single
O Dim Pos() As Single, Vel() As Single, Ace() As Single
O Dim FFTCar() As Single, bb As Single
O 'Static CalcOld As String, CargaOld, NudoOld, BarraOld 'dibujados en llamada anterior
O ReDim Mfft(xxml / 2), kk(1), Pos(xxml), FFTCar(xxml / 2)
O ReDim Vel(xxml), Ace(xxml)

```

```

O
O Dibujar.Enabled = False
O Borrar.Enabled = False
O Imprimir.Enabled = False
O Salir.Enabled = False
O Me.MousePointer = 11
O PrgNudo.Cls
O PrgBarra.Cls
O
O If CheckCarga.value = 1 Then 'si está seleccionado cargas.....
O   Carga = ListCarga.ListIndex + 1
O   If Carga > 0 Then
O     dibmatr DatCar(), PictOC, 1, PctBar(n).BackColor, 0, Maxi 'onda...
O     kkk = Valores.value
O     dibmatrmx FFTCar(), PictFC, 1, PctBar(n).BackColor, 0, Frmax, TiempoFF, 1, kkk
O       'FFT...
O   End If
O End If
O
O If CheckNudo.value = 1 Then 'si está seleccionado nudos.....
O   Nudo = CLng(Numnud.Text)
O   If Nudo <> NudoOld Or CalcOld <> extension$ Then FrProgr.Visible = True: LeeNud
O     Nudo, DatNud()
O
O For n = 1 To dimensN
O   i = 3 * (n - 1) + 1
O   If ChNud(i).value = 1 Or ChNud(i + 1).value = 1 Or ChNud(i + 2).value = 1 Then
O     For j = 1 To xxml: Pos(j) = DatNud(n, j): Next j
O     If ChNud(i + 1).value = 1 Or ChNud(i + 2).value = 1 Then
O       CalcOndas Pos(), Vel(), Ace(), iterac, dt
O     End If
O     For k = 0 To 2 'Pos, vel, ace.....
O       'Dibuja las curvas de ondas y FFT si el CheckBox está marcado...
O       'Para el color de las curvas usa el BackColor del Chekbox.
O       If ChNud(i + k) = 1 Then
O         If LbOnda.BorderStyle = 1 Then 'se está viendo ONDA
O           If Out = 1 Then
O             If k = 0 Then dibmatr Pos(), PictO, 0.95, ChNud(i + k).BackColor, 0,
O               Maxi: MxN(n) = Format$(Maxi, "#.##E+##") 'onda...
O             'If k = 0 Then dibmatrmx Pos(), PictO, 1, ChNud(i + k).BackColor, 0,
O               Maxi, 0, 2, 1 'onda...
O             If k = 1 Then dibmatr Vel(), PictO, 0.85, ChNud(i + k).BackColor, 0,
O               Maxi 'onda...
O             If k = 2 Then dibmatr Ace(), PictO, 0.75, ChNud(i + k).BackColor, 0,
O               Maxi 'onda...
O           Else
O             If k = 0 Then dibmatr Pos(), Printer, 0.33, ChNud(i + k).BackColor, 0,
O               Maxi 'onda...
O             If k = 1 Then dibmatr Vel(), Printer, 0.3, ChNud(i + k).BackColor, 0,
O               Maxi 'onda...
O             If k = 2 Then dibmatr Ace(), Printer, 0.27, ChNud(i + k).BackColor, 0,
O               Maxi 'onda...
O           End If
O         Else
O           If Out = 1 Then
O             If k = 0 Then FFTp Pos(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               PictF, 1, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O             If k = 1 Then FFTp Vel(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               PictF, 1, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O             If k = 2 Then FFTp Ace(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               PictF, 1, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O           Else
O             If k = 0 Then FFTp Pos(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               Printer, 0.33, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O             If k = 1 Then FFTp Vel(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               Printer, 0.33, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O             If k = 2 Then FFTp Ace(), Mfft(), kk(), 0, 0, 0: dibmatrmx Mfft(),
O               Printer, 0.33, ChNud(i + k).BackColor, 0, Frmax, TiempoFF, 1,
O               Valores.value 'FFT...
O           End If
O         End If
O       End If
O     maxOndas(i + k) = Maxi: frxFFT(i + k) = Frmax
O     If CheckCarga.value = 1 And Carga > 0 Then
O       FT FFTCar(), Mfft() 'función transferencia

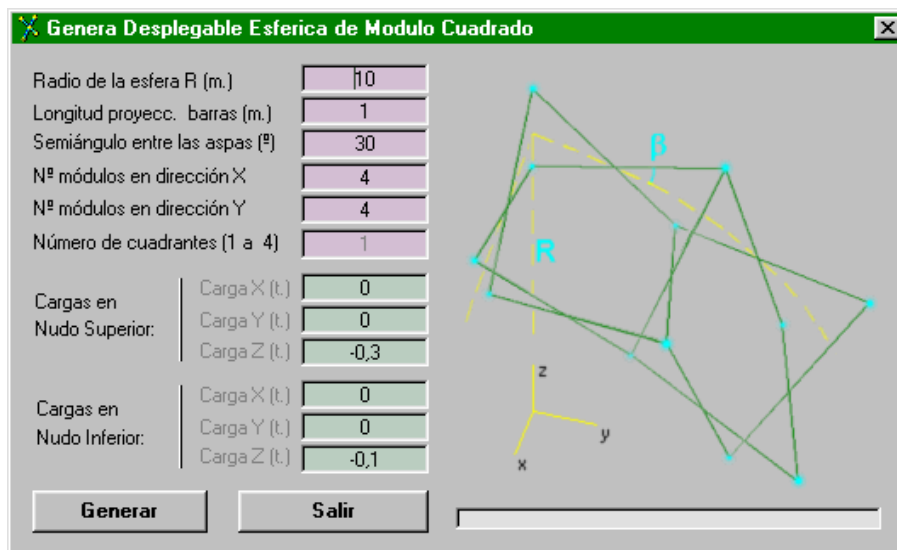
```

```

O         End If
O         End If
O     Next k
O     End If
O Next n
O End If
O
O If CheckBarra.value = 1 Then 'si está seleccionado barras.....
O     barra = CLng(Numbar.Text)
O     If barra <> BarraOld Or CalcOld <> extension$ Then FrProgr.Visible = True: LeeBar
O         barra, DatBar()
O
O     For n = 1 To dimensB
O         If ChBar(n).value = 1 Then
O             For j = 1 To iterac: Pos(j) = DatBar(n, j): Next j
O                 If LbOnda.BorderStyle = 1 Then 'se está viendo ONDA
O                     If Out = 1 Then
O                         dibmatr Pos(), PictO, 1, PctBar(n).BackColor, 0, Maxi: MxB(n) =
O                             CStr(Maxi) 'onda...
O                     Else
O                         dibmatr Pos(), Printer, 0.33, PctBar(n).BackColor, 0, Maxi 'onda...
O                     End If
O                 Else
O                     If Out = 1 Then
O                         FFTp Pos(), Mfft(), kk(), 0, 0, 0
O                         dibmatrmx Mfft(), PictF, 1, PctBar(n).BackColor, 0, Frmax, TiempoFF, 1,
O                             Valores.value 'FFT...
O                     Else
O                         FFTp Pos(), Mfft(), kk(), 0, 0, 0
O                         dibmatrmx Mfft(), Printer, 0.33, PctBar(n).BackColor, 0, Frmax, TiempoFF,
O                             1, Valores.value 'FFT...
O                     End If
O                 End If
O             End If
O
O             maxOndas(n) = Maxi: frxFFT(n) = Frmax
O             If CheckCarga.value = 1 And Carga > 0 Then
O                 End If
O         End If
O     Next n
O End If
O
O If LbOnda.BorderStyle <> 1 Then 'se está viendo Fourier
O     If Out = 1 Then
O         dibescala PictF, QBColor(15), 1
O     Else
O         dibescala Printer, QBColor(13), 0.92
O     End If
O Else 'se está viendo Ondas
O     bb = CSng(MaxV)
O     If Out = 1 Then
O         dibescalaT PictO, QBColor(15)
O         If bb <> 0 Then dibescalaV PictO, QBColor(15), bb
O     Else
O         dibescalaT Printer, QBColor(1)
O         If bb <> 0 Then dibescalaV Printer, QBColor(1), bb
O     End If
O End If
O
O CalcOld = extension$: CargaOld = Carga: NudoOld = Nudo: BarraOld = barra
O
O FrProgr.Visible = False
O Dibujar.Enabled = True
O Borrar.Enabled = True
O Imprimir.Enabled = True
O Salir.Enabled = True
O Me.MousePointer = 0
O
O End Sub
O
O

```

## 7.3.4 GENERACIÓN MALLA DESPLEGABLE CUADRADOS.



```

○ DefDbl A-Z
○ Option Explicit
○ 'malla 1 capa
○ Dim C() 'coordenadas nudos malla 1 capa
○ Dim L() 'barras: nudo inicio, medio y fin en malla 1 capa
○ 'malla 2 capas (final)
○ Dim Cd() 'coordenadas nudos
○ Dim Ld() 'nudos origen y fin
○ Dim Enlace() 'barras enlazadas
○ Dim Ra, R, nx, npd, ned
○
○
○
○ Private Sub Form_Activate()
○ NSQZ.Text = CStr(-0.3)
○ NIQZ.Text = CStr(-0.1)
○ End Sub
○
○
○ Private Sub Form_Unload(Cancel As Integer)
○ Menu.Show
○
○ End Sub
○
○ Private Sub Generar_Click()
○ Progreso.Cls
○ Progreso.AutoRedraw = False
○ Generar.Enabled = False
○ Dim d, G, P1, P2, M, w, MI, v, a, Be, L1, L2, ORX, ORY, ORZ, n1, n2
○ Dim p, num, j, i, Bx, By, Bz
○ Dim x1, y1, z1, x2, y2, z2, x3, y3, z3, Mx, My, Mz, rp, cow, tgw, la1, la2
○ Dim dis, e, r0, Dx, Dy, Dz
○ ' Variables usadas.....
○ ' Ra - radio de la esfera en m.
○ ' d - diámetro en m.
○ ' r - lado de cada una de las barras de la cuadrícula base en m.
○ ' G - ángulo, en radianes, que se corresponde con la longitud r.
○ ' n1 - n° de divisiones de la semimalla en dirección X
○ ' n2 - n° de divisiones de la semimalla en dirección Y
○ ' P1,P2 - puntos a partir de los cuales se hallan los equidistantes.
○ ' M - punto medio de P1-P2
○ ' w - semi ángulo de los vectores OP1-OP2
○ ' MI - distancia M-I
○ ' V - vector asociado a MI
○ ' A - punto de intersección buscado
○ ' Be - semi ángulo entre las aspas
○ ' .....
○ Ra = ValP(Radio.Text)
○ R = ValP(Longitud.Text)
○ n1 = ValP(Divisi1.Text)
○ n2 = ValP(Divisi2.Text)

```

```

O Be = ValP(Semiangulo.Text)
O Be = Be / 180 * pi
O
O 'n1 = N: n2 = N
O d = 2 * Ra
O p = Sqr((Ra * Ra) - (R * R / 4))
O G = 2 * Atn(R / (p * 2))
O
O 'CREACION DE LA MALLA BASE DE 1 CAPA (1/4).....
O np = (n1 + 1) * (n2 + 1)
O ReDim C(np, 3)
O ne = n1 * (n2 + 1) + n2 * (n1 + 1)
O ReDim L(ne, 3)
O
O 'Generación de nudos.....
O num = 1
O For j = 0 To n2: For i = 0 To n1
O BarraTiempo Progreso, (num), (np + ne)
O If i = 0 Or j = 0 Then
O C(num, 1) = Ra * Sin(G * i)
O C(num, 2) = Ra * Sin(G * j)
O C(num, 3) = Ra * Cos(G * (i + j))
O Else
O P1 = num - n1 - 1 '.....
O P2 = num - 1
O Intersec P1, P2, Bx, By, Bz
O C(num, 1) = Bx
O C(num, 2) = By
O C(num, 3) = Bz
O End If
O num = num + 1
O Next i: Next j
O
O 'Generación de barras.....
O num = 1
O For j = 1 To n2 + 1: For i = 1 To n1
O BarraTiempo Progreso, (np + num), (np + ne)
O L(num, 1) = i + ((j - 1) * (n1 + 1))
O L(num, 2) = i + ((j - 1) * (n1 + 1)) + 1
O num = num + 1
O Next i: Next j
O For j = 1 To n2: For i = 1 To n1 + 1
O BarraTiempo Progreso, np + num, np + ne
O L(num, 1) = i + ((j - 1) * (n1 + 1))
O L(num, 2) = i + ((j - 1) * (n1 + 1)) + n1 + 1
O num = num + 1
O Next i: Next j
O
O ' Y vamos a pasar de la malla de una capa ya calculada a la generación
O ' de las aspas y a almacenar los datos de barras en una matriz compatible
O ' para su cálculo LD(ned,4) así como los nudos en CD(npd,3)
O
O nx = ne ' nudos en el cruce de las aspas (se calculan en barras)'.....
O npd = np * 2 + nx
O ReDim Cd(npd, 3)
O ned = ne * 2
O ReDim Ld(ned, 4)
O ReDim Enlace(ne, 5) 'barral,barra2'.....
O
O ' Cálculo de la1 y la2
O P1 = 1: P2 = 2
O x1 = C(P1, 1): y1 = C(P1, 2): z1 = C(P1, 3)
O x2 = C(P2, 1): y2 = C(P2, 2): z2 = C(P2, 3)
O Dx = Abs(x2 - x1)
O Dy = Abs(y2 - y1)
O Dz = Abs(z2 - z1)
O dis = Sqr(Dx * Dx + Dy * Dy + Dz * Dz)
O Mx = (x2 + x1) / 2 'punto medio.
O My = (y2 + y1) / 2
O Mz = (z2 + z1) / 2
O M = Sqr(Mx * Mx + My * My + Mz * Mz)
O 'rp = Sqr(R * R - (dis * dis / 4)) ' obtenemos r'
O cow = (Ra * Ra + M * M - R * R / 4) / (2 * Ra * M) 'teorema del coseno
O tgw = Sqr(1 / (cow * cow) - 1)
O w = Atn(tgw)
O la1 = Ra * Sin(pi / 2 + Be) / Sin(pi / 2 - Be - w) 'teorema del seno
O la2 = Ra * Sin(pi / 2 - Be) / Sin(pi / 2 + Be - w) 'teorema del seno
O
O ' Generación nudos.....

```

```

O For i = 1 To np
O   e = i * 2
O   Cd(e - 1, 1) = la1 * C(i, 1) / Ra 'nudo inicial
O   Cd(e - 1, 2) = la1 * C(i, 2) / Ra
O   Cd(e - 1, 3) = la1 * C(i, 3) / Ra
O   Cd(e, 1) = la2 * C(i, 1) / Ra 'nudo final
O   Cd(e, 2) = la2 * C(i, 2) / Ra
O   Cd(e, 3) = la2 * C(i, 3) / Ra
O Next i
O
O ' Generación de barras.....
O For i = 1 To ne
O   P1 = L(i, 1)
O   P2 = L(i, 2)
O   ' punto de cruce...
O   x1 = C(P1, 1): y1 = C(P1, 2): z1 = C(P1, 3)
O   x2 = C(P2, 1): y2 = C(P2, 2): z2 = C(P2, 3)
O   Mx = (x2 + x1) / 2
O   My = (y2 + y1) / 2
O   Mz = (z2 + z1) / 2
O   r0 = Sqr(Mx * Mx + My * My + Mz * Mz)
O   Mx = Ra * Mx / r0
O   My = Ra * My / r0
O   Mz = Ra * Mz / r0
O   Cd((np * 2) + i, 1) = Mx 'nudo intermedio
O   Cd((np * 2) + i, 2) = My
O   Cd((np * 2) + i, 3) = Mz
O
O   ' barra 1...
O   Ld(i, 1) = P1 * 2 - 1 'nudo inicial
O   Ld(i, 2) = npd - nx + i 'nudo medio
O   Ld(i, 3) = P2 * 2 'nudo final
O   ' barra 2...
O   Ld(i + ne, 1) = P1 * 2 'nudo inicial
O   Ld(i + ne, 2) = npd - nx + i 'nudo medio
O   Ld(i + ne, 3) = P2 * 2 - 1 'nudo final
O
O   Enlace(i, 1) = i 'barras conectadas
O   Enlace(i, 2) = i + ne
O Next i
O
O Progreso.AutoRedraw = True
O BarraTiempo Progreso, 10, 10
O
O 'NB = 0: ntp = 4: ND = 3: AG = 0: CB = 0: MY = 0
O 'GOSUB 7000: ' Calculo ancho de banda .....??? consultar
O GuardaDat
O Generar.Enabled = True
O End Sub
O
O Sub Intersec(P1, P2, Bx, By, Bz)
O Dim x1, y1, z1, x2, y2, z2, x3, y3, z3, Dx, Dy, Dz, Mx, My, Mz, M, rp, cow, MI, dis
O Dim v, Vx, Vy, Vz, Ux, Uy, Uz, Mlx, MIy, MIz, Iax, Iay, Iaz, Ibx, Iby, Ibz, r0, kk
O x1 = C(P1, 1)
O y1 = C(P1, 2)
O z1 = C(P1, 3)
O x2 = C(P2, 1)
O y2 = C(P2, 2)
O z2 = C(P2, 3)
O Dx = (x2 - x1)
O Dy = (y2 - y1)
O Dz = (z2 - z1)
O dis = Sqr(Dx * Dx + Dy * Dy + Dz * Dz)
O Mx = (x2 + x1) / 2 'punto medio.
O My = (y2 + y1) / 2
O Mz = (z2 + z1) / 2
O M = Sqr(Mx * Mx + My * My + Mz * Mz)
O kk = R * R - (dis * dis / 4)
O If kk < 0 Then MsgBox "Los dos puntos se hallan demasiado alejados", vbOKOnly:
O   Stop
O rp = Sqr(kk) ' obtenemos r'
O If (Ra - M) > rp Then MsgBox "No hay solución sobre la esfera", vbOKOnly: Stop
O cow = (Ra * Ra + M * M - rp * rp) / (2 * Ra * M)
O MI = M * Sqr(1 / (cow * cow) - 1)
O Vx = My * Dz - Mz * Dy ' V ortogonal al plano OM,P1,P2
O Vy = Mz * Dx - Mx * Dz
O Vz = Mx * Dy - My * Dx
O v = Sqr(Vx * Vx + Vy * Vy + Vz * Vz)
O Ux = Vx / v

```



```

O   Uy = Vy / v
O   Uz = Vz / v
O   M1x = M1 * Ux
O   M1y = M1 * Uy
O   M1z = M1 * Uz
O   Iax = Mx + M1x ' solución 1...
O   Iay = My + M1y
O   Iaz = Mz + M1z
O   Ibx = Mx - M1x ' solución 2...
O   Iby = My - M1y
O   Ibz = Mz - M1z
O   r0 = Sqr(Ibx * Ibx + Iby * Iby + Ibz * Ibz)
O   Bx = Ra * Ibx / r0
O   By = Ra * Iby / r0
O   Bz = Ra * Ibz / r0
O
O   End Sub
O
O   Sub GuardaDat()
O   Dim aa$, a, b, kk, i, j, k, sQX, sQY, sQZ, iQX, iQY, iQZ
O   Dim x1, y1, z1, x2, y2, z2, Le#, Salir As Boolean
O   sQX = NSQX
O   sQY = NSQY
O   sQZ = NSQZ
O   iQX = NIQX
O   iQY = NIQY
O   iQZ = NIQZ
O
O   Rem *****GRABACION DE DATOS*****
O   Close
O
O   Salir = False
O   aa$ = nom$ + "\*.DAT"
O   If Dir(aa$) <> "" Then
O     If MsgBox("Existen ficheros de datos .DAT de una estructura previa que serán
                sobreescritos por la nueva. ¿Desea proseguir?", vbYesNo, "Existen
                datos previos") = vbYes Then
O       Kill aa$ 'elimina datos previos...
O     Else
O       Salir = True
O     End If
O   End If
O   If Salir Then Exit Sub
O
O   aa$ = nom$ + "\*.*)"
O   If Dir(aa$) <> "" Then
O     If MsgBox("Existen ficheros, aunque no de datos de una estructura previa, que no
                serán sobreescritos por la nueva, pero pueden entrar en conflicto a
                la hora de analizar resultados. ¿Los borramos todos?", vbYesNo,
                "Existen resultados previos") = vbYes Then
O       Kill aa$ 'borra resultados anteriores
O     End If
O   End If
O
O   aa$ = nom$ + "\DATOS.DAT"
O   Open aa$ For Output As #1
O   Write #1, np * 2
O   Write #1, ne * 2
O   Write #1, ne
O   Close #1
O
O   aa$ = nom$ + "\COORDENA.DAT"
O   Open aa$ For Output As #1
O   For j = 1 To 3: For i = 1 To np * 2
O     Print #1, FormatP(Cd(i, j), " 0.0000000E+00")
O   Next i: Next j
O   Close #1
O
O   aa$ = nom$ + "\BARRAS.DAT"
O   Open aa$ For Output As #1
O   For i = 1 To ne * 2
O     a = Ld(i, 1): kk = Ld(i, 2): b = Ld(i, 3)
O     Write #1, a
O     Write #1, b
O     Write #1, 1: ' En principio pone todas las barras del tipo 1
O   Next i
O   Close #1
O
O   'aa$ = Nom$ + "\COACCION.DAT"

```

```

○
○ aa$ = nom$ + "\CARGAS.DAT"
○ Open aa$ For Output As #1
○ For i = 1 To np * 2
○   If i Mod 2 = 0 Then 'nudo par= inferior
○     Print #1, FormatP(iQX, " 0.0000000E+00")
○     Print #1, FormatP(iQY, " 0.0000000E+00")
○     Print #1, FormatP(iQZ, " 0.0000000E+00")
○   Else
○     Print #1, FormatP(sQX, " 0.0000000E+00")
○     Print #1, FormatP(sQY, " 0.0000000E+00")
○     Print #1, FormatP(sQZ, " 0.0000000E+00")
○   End If
○ Next i
○ Close #1
○
○ aa$ = nom$ + "\BARRASA.DAT" '.....
○ Open aa$ For Output As #1
○ For i = 1 To ne
○   a = Enlace(i, 1): b = Enlace(i, 2)
○   Write #1, a
○   Write #1, b
○ Next i
○ Close #1
○
○ aa$ = nom$ + "\COORDENL.DAT"
○ Open aa$ For Output As #1
○ For k = 1 To 2
○   For i = 1 To ne: For j = 1 To 3
○     Print #1, FormatP(Cd((np * 2) + i, j), " 0.0000000E+00")
○   Next j: Next i
○ Next k
○ Close #1
○
○ Crea_Resto_Archivos 'para crear SLONGIT.DAT y otros...
○
○ End Sub
○
○ Private Sub Salir_Click()
○   Unload Me
○ End Sub
○
○
○

```

## 7.3.5 GENERACIÓN MALLA DESPLEGABLE TRIANGULOS.

**Genera Desplegable Esferica de Modulo Triangular**

Radio de la esfera R (m.)

Canto de la malla (m.)

Canto variable, máximo en el polo

Distancia Foco al Polo Sur (m.)

Angulo de generación  $\gamma^\circ$

Angulo de despliegue  $\rho^\circ$

Orden (nº de módulos)

Número de sextantes (1 a 6)

Cargas en Nudo Superior:

Carga X (t.)	<input type="text" value="0"/>
Carga Y (t.)	<input type="text" value="0"/>
Carga Z (t.)	<input type="text" value="-0,3"/>

Cargas en Nudo Inferior:

Carga X (t.)	<input type="text" value="0"/>
Carga Y (t.)	<input type="text" value="0"/>
Carga Z (t.)	<input type="text" value="-0,1"/>

Alinear nudo superior e inferior:

Con el centro de la esfera

Con el foco de proyección

Nota: La forma de generación tiende a garantizar el despliegue, por lo que es conveniente generarla en posición próxima al plegado, para garantizar también esta fase.

**Generar** **Salir**

```

O DefDb1 A-Z
O Option Explicit
O '-----
O 'Dim NP As Single 'nº puntos
O 'Dim NE As Single 'nº barras
O 'Dim NEA As Single 'nº barras enlace NEA=NE/2
O ''''Dim NPA As Single 'Nº puntos enlace NPA=NEA*2
O '-----
O Dim C() As Single 'NP coordenadas nudos
O Dim L() As Long 'NE barras: nudo inicio y fin
O Dim LA() As Long 'barras de enlace: barra inicio, barra fin
O Dim CA() As Single 'coordenadas nudos
O '-----
O Dim Ra As Single 'radio esfera
O 'Dim Prc As Single 'porcentaje
O Dim Gamma As Single 'angulo de generación
O Dim e As Single 'excetricidad foco
O Dim H As Single 'canto malla
O Dim Hvar As Boolean 'si es canto variable
O Dim Ro As Single 'angulo total de despliegue deseado
O Dim Ord As Single 'orden malla (nº módulos)
O Dim Sext As Single 'nº de sextantes a crear: 1 a 6
O '-----
O Dim num As Long, Nbarr As Long 'contadores de nudos y barras
O Dim numA As Long, nbarrasA As Long 'contadores de nudos y barras de enlace
O
O Private Sub Form_Activate()
O 'cargas por defecto en los nudos superior e inferior
O NSQZ.Text = CStr(-0.3)
O NIQZ.Text = CStr(-0.1)
O End Sub
O
O
O Private Sub Form_Unload(Cancel As Integer)
O Menu.Show
O
O End Sub
O
O Private Sub Generar_Click()
O Dim x As Single, y As Single, R As Single
O Dim x0 As Single, y0 As Single
O Dim x1 As Single, y1 As Single, z1 As Single, x2 As Single, y2 As Single, z2 As Single
O Dim lado As Single, NSX As Single, NSX1 As Single, NSX6 As Single
O Dim S As Long, Fila As Long, Nudo As Long
O Dim Fi As Single, a As Long, b As Long

```

```

O Dim NPsext(6) As Long
O Dim NEsext(6) As Long, n As Long, n2 As Long
O 'Rutinas usadas.....
O 'Proyectora(Ra, E, H, X0, Y0, x1, y1, z1, x2, y2, z2) Single
O
O Progreso.Cls
O Progreso.AutoRedraw = False
O Generar.Enabled = False
O
O Ra = ValP(Radio.Text)
O ' Prc = ValP(Porcentaje.Text)
O Gamma = ValP(AngGamma.Text)
O Gamma = Gamma / 180 * pi 'paso a radianes
O e = ValP(Excentricidad.Text)
O H = ValP(Canto.Text)
O If CantoVariable.value = 1 Then Hvar = True Else Hvar = False
O Ord = ValP(Orden.Text)
O Sext = ValP(Sextantes.Text)
O Ro = ValP(AngDespl.Text)
O Ro = Ro / 180 * pi 'paso a radianes
O
O 'lado = Ra * Prc / 100 / Ord 'lado triángulo en plano XY
O lado = (Ra * Sin(Gamma)) / (Ra * Cos(Gamma) + Ra + e) * (Ra + e) / Ord 'lado triángulo
O en plano XY
O
O NSX1 = FactS(Ord + 1) 'n° puntos 1° sextante
O NSX = FactS(Ord + 0) 'n° pto sextante tipo
O NSX6 = FactS(Ord - 1) 'n° puntos 6° sextante
O
O 'n° puntos y barras en función del n° de sextantes
O NPsext(1) = (NSX1) * 2 '2 capas
O NEsext(1) = ((NSX * 3)) * 2 '2 barras por aspa
O NPsext(2) = (NSX1 + NSX) * 2 '2 capas
O NEsext(2) = ((NSX * 3 - Ord) * 2 + Ord) * 2 '2 barras por aspa
O NPsext(3) = (NSX1 + NSX * 2) * 2 '2 capas
O NEsext(3) = ((NSX * 3 - Ord) * 3 + Ord) * 2 '2 barras por aspa
O NPsext(4) = (NSX1 + NSX * 3) * 2 '2 capas
O NEsext(4) = ((NSX * 3 - Ord) * 4 + Ord) * 2 '2 barras por aspa
O NPsext(5) = (NSX1 + NSX * 4) * 2 '2 capas
O NEsext(5) = ((NSX * 3 - Ord) * 5 + Ord) * 2 '2 barras por aspa
O NPsext(6) = (NSX1 + NSX * 4 + NSX6) * 2 '2 capas
O NEsext(6) = ((NSX * 3 - Ord) * 6) * 2 '2 barras por aspa
O
O np = NPsext(Sext)
O ne = NEsext(Sext)
O nea = ne / 2
O
O ReDim C(np, 3) 'NP coordenadas nudos
O ReDim L(ne, 2) 'NE barras: nudo inicio y fin
O ReDim LA(nea, 2) 'barras de enlace: barra inicio, barra fin
O ReDim CA(nea * 2, 3) 'coordenadas nudos
O
O '-----
O 'arrancamos
O
O num = 0: Nbarr = 0
O numA = 0: nbarrasA = 0
O
O 'nudos centrales
O x1 = 0: y1 = 0: z1 = Ra + H / 2
O x2 = 0: y2 = 0: z2 = Ra - H / 2
O otronudo x1, y1, z1: otronudo x1, y2, z2
O
O For S = 1 To Sext: For Fila = 1 To Ord: For Nudo = 1 To Fila + 1
O
O 'Generación de nudos.....
O If Not (S > 1 And Nudo = 1) Then
O If Not (S = 6 And Nudo = Fila + 1) Then
O x = lado * Fila - lado * (Nudo - 1) * Cos(pi / 3)
O y = lado * (Nudo - 1) * Sin(pi / 3)
O R = Sqr(x * x + y * y)
O Fi = Atn(y / x)
O Fi = Fi + pi / 3 * (S - 1)
O x0 = R * Cos(Fi)
O y0 = R * Sin(Fi)
O Proyectora Ra, e, H, Hvar, Ro, Fila, Ord, x0, y0, x1, y1, z1, x2, y2, z2
O otronudo x1, y1, z1: otronudo x2, y2, z2
O End If
O End If

```

```

O
O 'Generación de barras.....
O 'If 1 = 2 Then
O 'Barras A.....
O If Nudo > 1 Then
O If Nudo = 2 And S > 1 Then 'las primeras de los sucesivos sextantes
O If S = 6 Then
O If Fila = 1 Then 'la 1ª del último sextante
O '-----
O n = NPsext(4)
O a = 4: b = n + 1: OtraBarra a, b
O a = 3: b = n + 2: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la
O barra de enlace y sus dos nodos.
O Else 'el resto de 1º nudo del último sextante
O n = NPsext(4) + (FactS(Fila)) * 2
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la
O barra de enlace y sus dos nodos.
O End If
O Else
O n = NPsext(S - 2)
O If S = 2 Then n = n + FactS(Fila + 1) * 2 Else n = n + FactS(Fila) * 2
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n - 0: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
O de enlace y sus dos nodos.
O End If
O Else
O If Nudo = Fila + 1 And S = 6 Then 'las últimas A varia su nodo
O '-----
O n = NPsext(5) + FactS(Fila - 1) * 2
O a = FactS(Fila) * 2 + 2: b = n - 1: OtraBarra a, b
O a = FactS(Fila) * 2 + 1: b = n - 0: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
O de enlace y sus dos nodos.
O Else 'caso general
O a = num: b = num - 3: OtraBarra a, b
O a = num - 1: b = num - 2: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
O de enlace y sus dos nodos.
O End If
O End If
O End If
O 'Barras B.....
O If Nudo > 1 Then
O If Nudo = 2 And S > 1 Then 'las primeras de los sucesivos sextantes
O If Fila = 1 Then
O If S = 6 Then
O '-----
O Else
O a = num: b = 1: OtraBarra a, b
O a = num - 1: b = 2: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la
O barra de enlace y sus dos nodos.
O End If
O Else
O '-----
O n = NPsext(S - 2)
O If S = 2 Then n = n + FactS(Fila) * 2 Else n = n + FactS(Fila - 1) * 2
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n - 0: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
O de enlace y sus dos nodos.
O End If
O Else
O If Nudo = Fila + 1 And S = 6 Then 'las últimas B ya existen
O 'no hace nada
O Else 'caso general
O Select Case S
O Case 1: n = num - Fila * 2 - 2
O Case 2 To 5: n = num - (Fila - 1) * 2 - 2
O Case 6: n = num - (Fila - 1) * 2
O End Select
O a = num: b = n - 1: OtraBarra a, b
O a = num - 1: b = n - 0: OtraBarra a, b
O CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
O de enlace y sus dos nodos.

```

```

○      End If
○      End If
○      End If
○      'Barras C.....
○      If Nudo < (Fila + 1) Then
○          If Nudo = 1 And S > 1 Then 'las primeras de los sucesivos sextantes ya están
○              creadas
○          'no hace nada
○          Else 'caso general
○              Select Case S
○                  Case 1:      n = num - Fila * 2
○                  Case 2 To 5: n = num - (Fila - 1) * 2
○                  Case 6: If Nudo < Fila Then n = num - (Fila - 2) * 2 Else n = (FactS(Fila -
○                      1) + 1) * 2
○              End Select
○          a = num:      b = n - 1: OtraBarra a, b
○          a = num - 1: b = n - 0: OtraBarra a, b
○          CruceBarr 'calcula el punto de cruce de las dos últimas barras y crea la barra
○              de enlace y sus dos nodos.
○      End If
○      End If
○
○      BarraTiempo Progreso, (num), (np)
○      Next Nudo: Next Fila: Next S
○
○      If num <> np Then Stop
○      If Nbarr <> ne Then Stop
○      If nbarrasA <> nea Then Stop
○
○      Progreso.AutoRedraw = True
○      BarraTiempo Progreso, 10, 10
○
○      GuardaDat
○      Generar.Enabled = True
○      End Sub
○
○
○      Private Sub Proyecta(Ra As Single, e As Single, H As Single, Hvar As Boolean, Ro As
○          Single, Fila As Long, Orden As Single, x0 As Single, y0 As Single, x1
○          As Single, y1 As Single, z1 As Single, x2 As Single, y2 As Single, z2
○          As Single)
○      'Para proyectar un punto del plano XY en una esfera de radio Ra desde
○      'el polo sur+una excentricidad E hacia abajo, y obtener las coordenadas proyectadas
○          X,Y,Z
○      Dim R As Single, rp As Single, alfa As Single
○      Dim x As Single, y As Single, Z As Single 'coordenadas punto proyectado
○      Dim k As Single
○      Dim Roi As Single, Hi As Single
○      Dim Hv() As Single, Hm As Single, Roiv As Single
○      Dim n As Integer
○
○      ReDim Hv(Orden)
○      R = Sqr(x0 * x0 + y0 * y0) 'pasamos a un esquema plano
○      If R > Ra Then MsgBox "El punto a proyectar queda fuera de la esfera", vbCritical,
○          "Error": Exit Sub
○
○      If e = 0 Then 'proyección desde el polo sur - rutina simplificada
○          alfa = Atn(R / Ra)
○          'rp = R * (1 + Cos(2 * alfa))
○          rp = Ra * Sin(2 * alfa)
○          Z = Ra * Cos(2 * alfa)
○          'z = Sqr(Ra * Ra - rp * rp)
○      Else 'proyección desde puntos próximos al polo sur - rutina más compleja
○          ProyectaE Ra, R, e, rp, Z 'Datos:Ra,R,e      Devuelve:rp,z
○      End If
○
○      'coordenadas nudo proyectado
○      x = x0 * rp / R
○      y = y0 * rp / R
○
○      '-----
○      If Hvar Then
○          For n = 1 To Fila
○              Roiv = Ro / Orden * n
○              Hv(n) = (pi / (3 * Roiv)) * Sin(Roiv) * H
○          Next n
○          Hm = H
○          For n = 1 To Fila
○              Hm = Hm + Hv(n)

```

```

O     Next n
O     Hm = Hm / (Fila + 1)
O     Roi = Ro / Orden * Fila
O     Hi = (pi / (3 * Roi)) * Sin(Roi) * Hm
O Else
O     Hi = H
O End If
O 'Coordenadas nudo superior e inferior
O If OptionAlinea(0).value = True Then 'nudo sup. en inf. alineados al centro de la
O     esfera.
O     k = (Ra + Hi / 2) / Ra 'h-canto de la malla
O     x1 = x * k
O     y1 = y * k
O     z1 = Z * k
O     k = (Ra - Hi / 2) / Ra
O     x2 = x * k
O     y2 = y * k
O     z2 = Z * k
O Else 'nudo sup. en inf. alineados con el foco de proyección
O     R = Sqr((e + Ra + Z) * (e + Ra + Z) + rp * rp) 'nuevo radio de proyección
O     Z = Z + Ra + e 'desplazamos el origen de coordenadas al foco
O     k = (R + Hi / 2) / R 'h-canto de la malla
O     x1 = x * k
O     y1 = y * k
O     z1 = Z * k
O     k = (R - Hi / 2) / R
O     x2 = x * k
O     y2 = y * k
O     z2 = Z * k
O     z1 = z1 - Ra - e 'restauramos el origen de coordenadas
O     z2 = z2 - Ra - e
O End If
O '-----
O
O End Sub
O
O Sub ProyectaE(Ra As Single, R As Single, e As Single, rp As Single, Z As Single)
O Dim a As Single, b As Single, C As Single, d As Single, G As Single
O
O If R = Ra Then rp = Ra: Z = 0: Exit Sub
O
O G = (Ra + e) * (Ra + e)
O a = G + R * R
O b = -(2 * R * G)
O C = R * R * G - R * R * Ra * Ra
O d = b * b - 4 * a * C
O If d < 0 Then MsgBox "La proyección solo tiene soluciones imaginarias", vbCritical,
O     "Error": Exit Sub
O rp = (-b + Sqr(d)) / (2 * a)
O 'rp = (-b - Sqr(D)) / (2 * A)
O Z = Sqr(Ra * Ra - rp * rp)
O End Sub
O
O
O Private Function FactS(n As Long) As Long
O Dim j As Long, i As Long
O j = 0
O For i = 1 To n
O     j = j + i
O Next i
O FactS = j
O End Function
O Private Sub otronudo(x As Single, y As Single, Z As Single)
O num = num + 1: C(num, 1) = x: C(num, 2) = y: C(num, 3) = Z
O End Sub
O
O Private Sub OtraBarra(a As Long, b As Long)
O Nbarr = Nbarr + 1: L(Nbarr, 1) = a: L(Nbarr, 2) = b
O End Sub
O
O Private Sub CruceBarr()
O Dim n1 As Long, n2 As Long, n3 As Long, n4 As Long
O Dim x1#, y1#, z1#, x2#, y2#, z2#, x3#, y3#, z3#, x4#, y4#, z4#
O 'calcula el punto de cruce de dos últimas barras y crea barra enlace y sus dos nodos.
O
O nbarrasA = nbarrasA + 1
O LA(nbarrasA, 1) = Nbarr: LA(nbarrasA, 2) = Nbarr - 1
O
O n1 = L(Nbarr, 1)

```

```

O   x1 = C(n1, 1)
O   y1 = C(n1, 2)
O   z1 = C(n1, 3)
O   n2 = L(Nbarr, 2)
O   x2 = C(n2, 1)
O   y2 = C(n2, 2)
O   z2 = C(n2, 3)
O   n3 = L(Nbarr - 1, 1)
O   x3 = C(n3, 1)
O   y3 = C(n3, 2)
O   z3 = C(n3, 3)
O   n4 = L(Nbarr - 1, 2)
O   x4 = C(n4, 1)
O   y4 = C(n4, 2)
O   z4 = C(n4, 3)
O
O   CruceBarras x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4
O   numA = numA + 1
O   CA(numA, 1) = x1
O   CA(numA, 2) = y1
O   CA(numA, 3) = z1
O   numA = numA + 1
O   CA(numA, 1) = x2
O   CA(numA, 2) = y2
O   CA(numA, 3) = z2
O   End Sub
O
O   Sub GuardaDat()
O   Dim aa$, a, b, kk, i, j, k, sQX, sQY, sQZ, iQX, iQY, iQZ
O   Dim x1, y1, z1, x2, y2, z2, Le#, Salir As Boolean
O   sQX = NSQX
O   sQY = NSQY
O   sQZ = NSQZ
O   iQX = NIQX
O   iQY = NIQY
O   iQZ = NIQZ
O
O   Rem *****GRABACION DE DATOS*****
O   Close
O   Salir = False
O   aa$ = nom$ + "\*.DAT"
O   If Dir(aa$) <> "" Then
O     If MsgBox("Existen ficheros de datos .DAT de una estructura previa que serán
                sobreescritos por la nueva. ¿Desea proseguir?", vbYesNo, "Existen
                datos previos") = vbYes Then
O       Kill aa$ 'elimina datos previos....
O     Else
O       Salir = True
O     End If
O   End If
O   If Salir Then Exit Sub
O
O   aa$ = nom$ + "\*.*"
O   If Dir(aa$) <> "" Then
O     If MsgBox("Existen ficheros, aunque no de datos de una estructura previa, que no
                serán sobreescritos por la nueva, pero pueden entrar en conflicto a
                la hora de analizar resultados. ¿Los borramos todos?", vbYesNo,
                "Existen datos previos") = vbYes Then
O       Kill aa$ 'borra resultados anteriores
O     End If
O   End If
O
O
O
O   aa$ = nom$ + "\DATOS.DAT"
O   Open aa$ For Output As #1
O   Write #1, num 'np
O   Write #1, Nbarr 'ne
O   Write #1, nbarrasA 'nea
O   Close #1
O
O   aa$ = nom$ + "\COORDENA.DAT"
O   Open aa$ For Output As #1
O   For j = 1 To 3: For i = 1 To num 'np
O     Print #1, FormatP(C(i, j), " 0.0000000E+00")
O   Next i: Next j
O   Close #1
O   'DiscoX "COORDENA.DAT", "G", "S3", iii, C(), ddd, xxx, ttt$
O

```

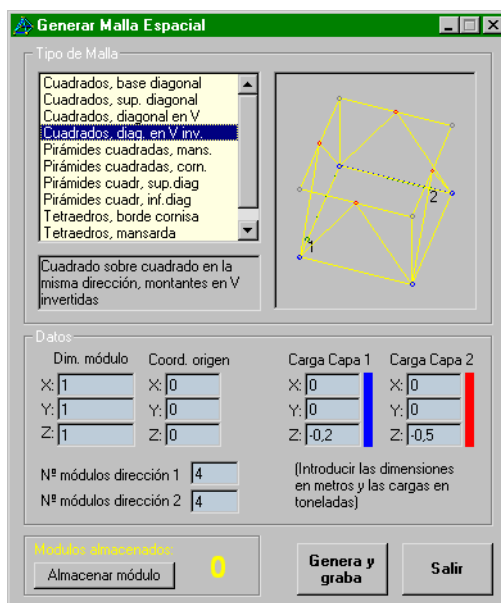


```

O aa$ = nom$ + "\BARRAS.DAT"
O Open aa$ For Output As #1
O For i = 1 To Nbarr 'ne
O   a = L(i, 1)
O   b = L(i, 2)
O   If a > b Then Swap a, b
O   Write #1, a
O   Write #1, b
O   Write #1, 1: ' En principio pone todas las barras del tipo 1
O Next i
O Close #1
O
O aa$ = nom$ + "\CARGAS.DAT"
O Open aa$ For Output As #1
O For i = 1 To num * 2
O   If i Mod 2 = 0 Then 'nudo par= inferior
O     Print #1, FormatP(iQX, " 0.0000000E+00")
O     Print #1, FormatP(iQY, " 0.0000000E+00")
O     Print #1, FormatP(iQZ, " 0.0000000E+00")
O   Else
O     Print #1, FormatP(sQX, " 0.0000000E+00")
O     Print #1, FormatP(sQY, " 0.0000000E+00")
O     Print #1, FormatP(sQZ, " 0.0000000E+00")
O   End If
O Next i
O Close #1
O
O aa$ = nom$ + "\BARRASA.DAT" '.....
O Open aa$ For Output As #1
O For i = 1 To nbarrasA
O   a = LA(i, 1): b = LA(i, 2)
O   Write #1, a
O   Write #1, b
O Next i
O Close #1
O 'DiscoX "BARRASA.DAT", "G", "L", LA(), sss, ddd, xxx, ttt$
O
O aa$ = nom$ + "\COORDENL.DAT" '.....
O Open aa$ For Output As #1
O For i = 1 To nbarrasA * 2: For j = 1 To 3
O   Print #1, FormatP(CA(i, j), " 0.0000000E+00")
O Next j: Next i
O Close #1
O 'DiscoX "COORDENL.DAT", "G", "S", iii, CA(), ddd, xxx, ttt$
O
O Crea_Resto_Archivos 'para crear SLONGIT.DAT y otros...
O
O
O End Sub
O
O Private Sub CmdSalir_Click()
O   Unload Me
O End Sub
O
O
O

```

## 7.3.6 GENERACIÓN MALLAS ESPACIALES.



```

O Option Explicit
O 'TIPOXX Comentario$ Comienza la definición de un nuevo tipo
O 'De$ Descripción larga de la malla
O 'NCP número de capas
O 'DIRE(2,3) direcciones principales
O 'B número de barras
O 'P número de puntos
O 'BA1OR n° barras a añadir en dir. 1 sobre borde origen
O 'BA1EX n° barras a añadir en dir. 1 sobre borde extremo
O 'BA2OR n° barras a añadir en dir. 2 sobre borde origen
O 'BA2EX n° barras a añadir en dir. 2 sobre borde extremo
O 'BEOO n° barras en esquina origen, origen
O 'BEEO n° " " " extremo, origen
O 'BEOE n° " " " origen, extremo
O 'BEEE n° " " " extremo, extremo
O 'MB(B,2) matriz de datos de barras
O 'MP(P,4) matriz de datos de puntos
O 'BA1ORm(BA1OR,2) " " " barras añadir 1, origen
O 'BA1EXm(BA1EX,2) " " " " " 1, extremo
O 'BA2ORm(BA2OR,2) " " " " " 2, origen
O 'BA2EXm(BA2EX,2) " " " " " 2, extremo
O 'BEOOm(BEOO,2) " " " " " esquina or.,or.
O 'BEEOm(BEEO,2) " " " " " " ex.,or.
O 'BEOEm(BEOE,2) " " " " " " or.,ex.
O 'BEEEm(BEEE,2) " " " " " " ex.,ex.
O 'ESCX, ESCY, ESCZ escalas recomendables para ejes x, y, z
O Dim tipo As Integer, comentario$, De$, NCP As Integer, Dire() As Single
O Dim b As Integer, p As Integer, BA1OR As Integer, BA1EX As Integer, BA2OR As Integer,
  BA2EX As Integer
O Dim BEOO As Integer, BEEO As Integer, BEOE As Integer, BEEE As Integer
O Dim MB() As Integer, MP() As Single
O Dim BA1ORm() As Integer, BA1EXm() As Integer, BA2ORm() As Integer, BA2EXm() As Integer
O Dim BEOOm() As Integer, BEEOm() As Integer, BEOEm() As Integer, BEEEm() As Integer
O Dim ESCX As Single, ESCY As Single, ESCZ As Single, carNCP() As Single
O
O Dim cp() As Single 'cordenadas de pantalla
O Dim fin As Integer, MODULO As Integer
O
O Private Sub Almacena_Click()
O fin = 0
O Adisco
O End Sub
O
O Private Sub Form_Load()
O Dim a$
O ReDim Dire(2, 3)
O
O fin = 0

```

```

O MODULO = 0 'Numero del modulo que se graba en disco.
O
O List1.Clear
O Open App.Path + "\mallas.dat" For Input As 1
O Do
O   Line Input #1, a$
O   If Left$(a$, 4) = "Tipo" Then
O     a$ = Right$(a$, Len(a$) - 7)
O     List1.AddItem a$
O   End If
O Loop Until EOF(1)
O Close #1
O
O If List1.ListCount > 0 Then
O   List1.ListIndex = 0
O End If
O
O
O
O End Sub
O Private Sub leetipo(Tip As Integer)
O Dim i As Integer, j As Integer, a$
O Dim encontrado As Boolean, dato As Boolean
O
O Open App.Path + "\mallas.dat" For Input As 1
O Do
O   Line Input #1, a$
O   If (Left$(a$, 4) = "Tipo") And (Val(Mid$(a$, 5, 2)) = Tip) Then encontrado = True
O Loop Until encontrado Or EOF(1)
O
O If encontrado Then
O   Line Input #1, De$
O   Input #1, NCP
O   For i = 1 To 2: For j = 1 To 3: Input #1, Dire(i, j): Next j: Next i
O   Input #1, b, p, BA1OR, BA1EX, BA2OR, BA2EX, BEOO, BEEO, BEOE, BEEE
O   ReDim MB(b, 2), MP(p, 4), cp(p, 2), BA1ORm(BA1OR, 2), BA1EXm(BA1EX, 2)
O   ReDim BA2ORm(BA2OR, 2), BA2EXm(BA2EX, 2)
O   ReDim BEOOm(BEOO, 2), BEEOm(BEEO, 2), BEOEm(BEOE, 2), BEEEm(BEEE, 2)
O   ReDim carNCP(NCP, 3)
O   For i = 1 To b:   For j = 1 To 2: Input #1, MB(i, j):   Next j: Next i
O   For i = 1 To p: For j = 1 To 4:   Input #1, MP(i, j):   Next j: Next i
O   For i = 1 To BA1OR: For j = 1 To 2: Input #1, BA1ORm(i, j): Next j: Next i
O   For i = 1 To BA1EX: For j = 1 To 2: Input #1, BA1EXm(i, j): Next j: Next i
O   For i = 1 To BA2OR: For j = 1 To 2: Input #1, BA2ORm(i, j): Next j: Next i
O   For i = 1 To BA2EX: For j = 1 To 2: Input #1, BA2EXm(i, j): Next j: Next i
O   For i = 1 To BEOO:  For j = 1 To 2: Input #1, BEOOm(i, j): Next j: Next i
O   For i = 1 To BEEO:  For j = 1 To 2: Input #1, BEEOm(i, j): Next j: Next i
O   For i = 1 To BEOE:  For j = 1 To 2: Input #1, BEOEm(i, j): Next j: Next i
O   For i = 1 To BEEE:  For j = 1 To 2: Input #1, BEEEm(i, j): Next j: Next i
O   Input #1, ESCX, ESCY, ESCZ
O End If
O
O Close #1
O
O If encontrado Then
O   dibujatipo
O   Descr.Caption = De$
O   Dx.Text = ESCX
O   Dy.Text = ESCY
O   Dz.Text = ESCZ
O
O   dato = Not (NCP = 1)
O   CR2X.Visible = dato
O   CR2Y.Visible = dato
O   CR2Z.Visible = dato
O End If
O
O End Sub
O
O Private Sub dibujatipo()
O 'Cálculo de las coordenadas de pantalla CP() para las axonometrías.....
O Dim cr As Single, cs As Single, ct As Single, x As Single, y As Single
O Dim x1 As Single, x2 As Single, y1 As Single, y2 As Single, i As Integer
O Dim hx1 As Single, hx2 As Single, hx3 As Single, hx4 As Single, hx5 As Single
O Dim hml As Single, hm2 As Single, Esc As Single
O Dim a$, coll As Single, n1 As Integer, n2 As Integer
O Dim px As Single, py As Single
O

```

```

O hx1 = 0.4 * 2: hx2 = 0.12 * 2: hx3 = 0.6: hx4 = 0.99: hx5 = 0.35
O '*****
O MP(0, 1) = 0: MP(0, 2) = 0: MP(0, 3) = 0: ' para dibujo de los ejes X Y Z.
O cr = MP(1, 1): cs = MP(1, 2): ct = MP(1, 3)
O y1 = -hx1 * cr - hx2 * cs + hx3 * ct: x1 = hx4 * cs - hx5 * cr: y2 = y1: x2 = x1
O For i = 1 To p
O   cr = MP(i, 1): cs = MP(i, 2): ct = MP(i, 3)
O   If MP(0, 1) < cr Then MP(0, 1) = cr: ' para dibujo de los ejes X Y Z.
O   If MP(0, 2) < cs Then MP(0, 2) = cs: ' para dibujo de los ejes X Y Z.
O   If MP(0, 3) < ct Then MP(0, 3) = ct: ' para dibujo de los ejes X Y Z.
O   cp(i, 2) = -hx1 * cr - hx2 * cs + hx3 * ct: cp(i, 1) = hx4 * cs - hx5 * cr
O   If cp(i, 1) < x1 Then x1 = cp(i, 1)
O   If cp(i, 1) > x2 Then x2 = cp(i, 1)
O   If cp(i, 2) < y1 Then y1 = cp(i, 2)
O   If cp(i, 2) > y2 Then y2 = cp(i, 2)
O Next i
O x = x2 - x1: y = y2 - y1
O
O px = Pict1.ScaleWidth 'n° de pixels del recuadro en la dirección X
O py = Pict1.ScaleHeight 'n° de pixels del recuadro en la dirección Y
O If x / px < y / py Then Esc = 0.8 * py / y Else Esc = 0.8 * px / x
O hm1 = (px - Esc * x) / 2: hm1 = -x1 * Esc + hm1
O hm2 = (py - Esc * y) / 2: hm2 = y2 * Esc + hm2
O For i = 1 To p
O   cp(i, 1) = hm1 + Esc * cp(i, 1)
O   cp(i, 2) = hm2 - Esc * cp(i, 2)
O Next i
O
O '*****
O Pict1.Cls
O
O '*****
O 'Dibujo ejes de coordenadas
O cr = MP(0, 1) * 1.02: cs = 0: ct = 0: GoSub 5010
O cr = 0: cs = MP(0, 2) * 1.02: ct = 0: GoSub 5010
O cr = 0: cs = 0: ct = MP(0, 3) * 1.02: GoSub 5010
O GoTo 5050
O
O 5010
O   y = -(-hx1 * cr - hx2 * cs + hx3 * ct) * Esc + hm2
O   x = (hx4 * cs - hx5 * cr) * Esc + hm1
O   Pict1.Line (hm1, hm2)-(x, y), QBColor(9)
O Return
O
O 5050
O For i = 1 To p
O   x = cp(i, 1)
O   y = cp(i, 2)
O   Pict1.Circle (x, y), 20, QBColor(11)
O Next i
O
O For i = 1 To 2
O   cr = Dire(i, 1): cs = Dire(i, 2): ct = Dire(i, 3)
O   y = -(-hx1 * cr - hx2 * cs + hx3 * ct) * Esc * 0.8 + hm2
O   x = (hx4 * cs - hx5 * cr) * Esc * 0.8 + hm1
O   Pict1.Line (hm1, hm2)-(x, y), QBColor(2)
O   Pict1.Circle (x, y), 20, QBColor(2)
O   a$ = Str$(i): a$ = Right$(a$, 1)
O   Pict1.Print a$
O Next i
O
O coll = QBColor(14)
O For i = 1 To BEOO
O   n1 = BEOOm(i, 1): n2 = BEOOm(i, 2): GoSub 5555
O Next i
O
O coll = QBColor(14)
O For i = 1 To BEEE
O   n1 = BEEEm(i, 1): n2 = BEEEm(i, 2): GoSub 5555
O Next i
O coll = QBColor(14)
O For i = 1 To b
O   n1 = MB(i, 1): n2 = MB(i, 2)
O   GoSub 5555
O Next i
O
O Exit Sub
O '*****
O

```

```

○ 5555
○ x1 = cp(n1, 1): x2 = cp(n2, 1)
○ y1 = cp(n1, 2): y2 = cp(n2, 2)
○ Pict1.Line (x1, y1)-(x2, y2), coll
○ Return
○
○ End Sub
○
○
○ Private Sub Adisco()
○ '*****
○ '***** ALMACENAMIENTO DE DATOS EN DISCO *****
○ '*****
○ Dim a$, i As Integer, j As Integer
○ Dim cr As Single, cs As Single, ct As Single
○
○ 'Actualiza el tamaño de las direcciones y coordenadas de puntos.
○ For i = 1 To 2
○ cr = Dire(i, 1): cs = Dire(i, 2): ct = Dire(i, 3)
○ Dire(i, 1) = cr * CSng(Dx.Text): Dire(i, 2) = cs * CSng(Dy.Text): Dire(i, 3) = ct *
○ CSng(Dz.Text)
○ Next i
○ For i = 1 To p
○ cr = MP(i, 1): cs = MP(i, 2): ct = MP(i, 3)
○ MP(i, 1) = cr * CSng(Dx.Text): MP(i, 2) = cs * CSng(Dy.Text): MP(i, 3) = ct *
○ CSng(Dz.Text)
○ Next i
○
○ carNCP(1, 1) = CSng(CR1X.Text)
○ carNCP(1, 2) = CSng(CR1Y.Text)
○ carNCP(1, 3) = CSng(CR1Z.Text)
○
○ If NCP > 1 Then
○ carNCP(2, 1) = CSng(CR2X.Text)
○ carNCP(2, 2) = CSng(CR2Y.Text)
○ carNCP(2, 3) = CSng(CR2Z.Text)
○ End If
○
○ 'Orden de
○ grabacion:FIN,ORX,ORY,ORZ,N,M,DI(2,3),B,P,BA1OR,BA1EX,BA2OR,BA2EX,BEO
○ O,BEEO,BEOE,BEEE
○ 'MB(B,2),MP(P,4),BA1OR(BA1OR,2),BA1EX(BA1EX,2),BA2OR(BA2OR,2),BA2EX(BA2EX,2),BEO(BEEO
○ ,2),BEEO(BEEO,2),BEOE(BEOE,2),BEEE(BEEE,2),NCP,NCP(NCP,3)
○ a$ = App.Path + "\MODUL-" + Right$(Str$(MODULO), 1) + ".DAT"
○ Open a$ For Output As #1
○ Write #1, fin, CSng(ORX.Text), CSng(ORY.Text), CSng(ORZ.Text), CSng(M1.Text),
○ CSng(M2.Text) ', tipo
○ For i = 1 To 2: For j = 1 To 3: Write #1, Dire(i, j),: Next j: Next i
○ Write #1, b, p, BA1OR, BA1EX, BA2OR, BA2EX, BEOO, BEEO, BEOE, BEEE
○ For i = 1 To b: For j = 1 To 2: Write #1, MB(i, j),: Next j: Next i
○ For i = 1 To p: For j = 1 To 4: Write #1, MP(i, j),: Next j: Next i
○ For i = 1 To BA1OR: For j = 1 To 2: Write #1, BA1ORm(i, j),: Next j: Next i
○ For i = 1 To BA1EX: For j = 1 To 2: Write #1, BA1EXm(i, j),: Next j: Next i
○ For i = 1 To BA2OR: For j = 1 To 2: Write #1, BA2ORM(i, j),: Next j: Next i
○ For i = 1 To BA2EX: For j = 1 To 2: Write #1, BA2EXm(i, j),: Next j: Next i
○ For i = 1 To BEOO: For j = 1 To 2: Write #1, BEOOm(i, j),: Next j: Next i
○ For i = 1 To BEEO: For j = 1 To 2: Write #1, BEEOm(i, j),: Next j: Next i
○ For i = 1 To BEOE: For j = 1 To 2: Write #1, BEOEm(i, j),: Next j: Next i
○ For i = 1 To BEEE: For j = 1 To 2: Write #1, BEEEm(i, j),: Next j: Next i
○ Write #1, NCP,
○ For i = 1 To NCP: For j = 1 To 3
○ Write #1, carNCP(i, j),: Next j: Next i
○ Close #1
○ '*****
○ MODULO = MODULO + 1
○ Modu.Caption = Str$(MODULO)
○
○ End Sub
○
○ Private Sub Form_Unload(Cancel As Integer)
○ Menu.Show
○ End Sub
○
○ Private Sub Generar_Click()
○ Dim a$, aa$, Salir As Boolean
○
○ Salir = False
○ aa$ = nom$ + "\*.DAT"
○ If Dir(aa$) <> "" Then

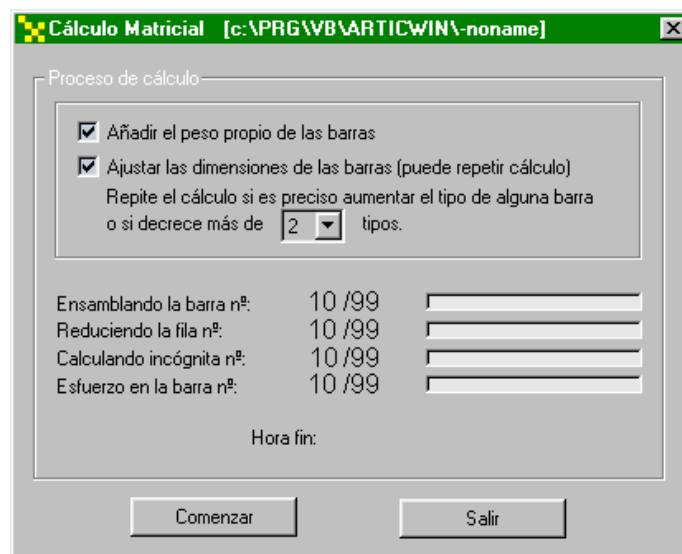
```

```

○ If MsgBox("Existen ficheros de datos .DAT de una estructura previa que serán
sobreescritos por la nueva. ¿Desea proseguir?", vbYesNo, "Existen
datos previos") = vbYes Then
○ Kill aa$ 'elimina datos previos...
○ Else
○ Salir = True
○ End If
○ End If
○ If Salir Then Exit Sub
○
○ aa$ = nom$ + "\*.*)"
○ If Dir(aa$) <> "" Then
○ If MsgBox("Existen ficheros, aunque no de datos de una estructura previa, que no
serán sobreescritos por la nueva, pero pueden entrar en conflicto a
la hora de analizar resultados. ¿Los borramos todos?", vbYesNo,
"Existen resultados previos") = vbYes Then
○ Kill aa$ 'borra resultados anteriores
○ End If
○ End If
○ fin = 1
○ Adisco
○ Genera.Show 1
○
○ a$ = App.Path + "\MODUL-*.DAT": Kill a$
○
○ fin = 0
○ MODULO = 0
○ Modu.Caption = Str$(MODULO)
○ End Sub
○
○ Private Sub List1_Click()
○ If List1.ListIndex >= 0 Then 'hay algún elemento seleccionado
○ tipo = List1.ListIndex
○ leetipo (tipo)
○ End If
○ End Sub
○
○ Private Sub Salir_Click()
○ Unload Me
○ End Sub
○
○
○

```

## 7.3.7 MÓDULO DE CÁLCULO MATRICIAL.



```

○ Option Explicit
○ '.....: CALCUL.BAS .....:
○ '
○ ' ESTE PROGRAMA CALCULA LA MATRIZ DE RIGIDEZ EN COORDENADAS GLOBALES DE
○ ' CADA BARRA, LAS ENSAMBLA Y RESUELVE EL SISTEMA DE ECUACIONES

```

```

O ' Los desplazamientos totales se guardar n en INCOG.DAT
O ' El programa contempla los casos de que la matriz general de rigidez no
O ' quepa totalmente en memoria, esto quedar reflejado en la variable
O ' MATR , que puede tomar los valores:
O '           1 - Matriz A() totalmente en memoria
O '           2 - En memoria solo A(NT,NT)
O '           3 - A() totalmente en disco.
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O 'Datos generales:
O '   np   =   n° puntos
O '   ne   =   n° barras
O 'Condiciones parada en cálculo matricial. Las que son <=0 no se tienen en cuenta.
O 'Cuando se cumpla una condición de parada se detiene el proceso
O '   Itot =   n° iteraciones que se desean efectuar
O '   AdmAbs= desplazamiento absoluto admisible. Cuando en todos
O '           'los puntos el desplazamiento de la última iteración
O '           'es menor que este valor, se detiene el proceso
O '   Adm   =   cuando en todos los puntos, el tanto por uno del
O '           'desplazamiento de esta iteración respecto al
O '           'acumulado es menor de este valor, se para (convergencia).
O 'Si todas son cero, solo se efectuar una iteraciñ.
O 'Datos cálculo dinámico
O '   ncargd=   n° de cargas dinámicas
O '   naceld=   n° aceleraciones
O 'Otras características cálculo matricial:
O '   ajustabar= 0-no ajusta las dimensiones de las barras; 1-sí lo hace
O '   ppropio=  0-no añade el peso propio de las barras; 1-sí lo hace
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O Dim C() As Double      'coordenadas
O Dim L() As Long        'barras
O Dim R() As Double      'cargas
O Dim x() As Double      'incógnitas (desplazamientos)
O Dim pp$( ), p() As Double 'tipos de barras
O Dim despl() As Double  'desplazamiento total acumulado en todas las iteraciones
O Dim lon0#()            'longitudes iniciales de las barras
O Dim a() As Double      'matriz global de rigidez
O Dim Matr As Integer    'tipo de cálculo
O 'Dim np As Long, ne As Long , Ntp As Integer
O 'Dim NS As Long, ND As Integer, NT As Long
O Dim Itot As Integer, Adm As Double, AdmAbs As Double, ppropio As Integer
O Dim Parar As Integer
O
O Private Sub Comenzar_Click()
O
O Dim iteracion As Integer
O Dim aa$, i As Long, ti As Integer, Lng As Double
O Dim masabarra As Double, ix As Double, iy As Double, iz As Double
O Dim n1 As Long, n2 As Long, n4 As Long, n6 As Long
O Dim x1 As Double, y1 As Double, z1 As Double
O Dim x2 As Double, y2 As Double, z2 As Double
O Dim xp As Double, yp As Double, zp As Double
O Dim L0#, Le#, fuer As Double, FM As Double
O Dim c1 As Double, c2 As Double, c3 As Double
O Dim LP As Integer, AR As Double, My As Double
O Dim repetir As Integer, CambiosBar As Integer
O Dim b$, j As Long, a1 As Integer, a2 As Integer 'precisas solo para parte de chequeo
O On Error GoTo errores
O Dim chequear As String
O 'Dim version$
O Dim version$, NumSerie$, program$, versPrg$, frase$, archivo$ 'parámetros para la
O                               rutina cripto
O
O If Not (VerificoArranque) Then Exit Sub
O
O Comenzar.Enabled = False
O Salir.Enabled = False
O ppropio = PesoPropio.value
O
O
O LeeDat
O
O Itot = 0: Adm = 0: AdmAbs = 0
O If ChkItot.value = 1 Then Itot = Cint(TxtItot.Text)
O If ChkAdm.value = 1 Then Adm = CSng(TxtAdm.Text)
O If ChkAdmAbs.value = 1 Then AdmAbs = CSng(TxtAdmAbs.Text)
O If Itot = 0 And Adm = 0 And AdmAbs = 0 Then Itot = 1
O
O
O 'Selección del tipo de matriz A().....

```

```

O If OptMatri(1).value = True Then Matr = 1
O If OptMatri(2).value = True Then Matr = 2
O If OptMatri(3).value = True Then Matr = 3
O
O Param.Enabled = False: Param.Visible = False
O DoEvents
O
O Inicio:
O PctBa.Cls
O PctFi.Cls
O PctIg.Cls
O PctEs.Cls
O
O ' Matriz de rigidez A().....
O Select Case Matr ' *****
O   Case 1: ReDim a(NS, NT)
O   Case 2: ReDim a(NT, NT)
O   Case 3: ReDim a(1, NT)
O End Select
O
O
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::Bucle principal::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O '::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O Parar = 0 ' =1 si se cumple alguna condici3n para detener las iteraciones
O iteracion = 0 ' N3mero de la iteraci3n actual que est efectuando.
O baT.Caption = "/" + CStr(ne)
O fiT.Caption = "/" + CStr(NS)
O igT.Caption = "/" + CStr(NS)
O esT.Caption = "/" + CStr(ne)
O
O Do
O   iteracion = iteracion + 1
O   LabIter.Caption = CStr(iteracion)
O   DiscoX "COACCION.DAT", "L", "D", iii, sss, x(), xxx, ttt$
O   For i = 1 To NS
O   If x(i) <> 0 And x(i) <> 1234 And iteracion > 1 Then x(i) = 0
O   'los desplazamientos impuestos pasan a cero en las sucesivas iteraciones
O   Next i
O   DiscoX "CARGAS.DAT", "L", "D", iii(), sss(), R(), xxx(), ttt$
O   If ppropio = 1 Then 'si se a3ade el peso propio de las barras
O   For i = 1 To ne 'hacemos carga Z en cada nudo = carga Z + p.p. barras
O     n1 = L(i, 1): n2 = L(i, 2): ti = L(i, 3): Lng = lon0#(i)
O     masabarra = p(ti, 1) * Lng * 100 * p(ti, 5) / 1000000 ' en Kg...
O     R(n1 * 3) = R(n1 * 3) - masabarra * 0.0005 ' tn.. (carga hacia abajo)
O     R(n2 * 3) = R(n2 * 3) - masabarra * 0.0005 ' tn.. (carga hacia abajo)
O   Next i
O   End If
O
O   Select Case Matr: ' *****
O     Case Is = 1: Rem
O     Case Is > 1
O       Open App.Path + "\MATRIZ.DAT" For Random As #1 Len = Len(a(1, 1))
O       For i = 1 To NS * NT
O         Put #1, , 0
O       Next i
O     End Select
O
O   Call Ensambla
O
O chequear = "No" 'no chequeamos matrices..
O '.....
O If chequear = "Si" Then
O   Open nom$ + "\matriz.S" For Output As #1 'para chequear,....
O   Open nom$ + "\matriz.L" For Output As #2 'cargas
O   Open nom$ + "\matriz.C" For Output As #3 'coacciones
O   For i = 1 To np * 3
O     For j = 1 To NS
O       Select Case j 'V3lido solamente..
O       Case Is <= (i - NT): Print #1, 0, 'si matriz cabe....
O       Case (i - NT + 1) To (i - 1): Write #1, a(j, i - j + 1), 'enteramente en....
O       Case (i) To (i + NT - 1): Write #1, a(i, j - i + 1), 'memoria.....
O       Case Is >= (i + NT): Write #1, 0,
O     End Select
O   Next j 'ton.metros.
O   Write #1,
O   Write #2, R(i)

```





```

O Case Is = 1: Rem
O Case Is > 1: Kill App.Path + "\matriz.dat"
O End Select
O
O DiscoX "INCOG.RES", "G", "D", iii, sss, x(), xxx, ttt$
O
O Esfuerzos CambiosBar ' #####
O If CambiosBar = 1 Then GoTo Inicio
O
O GoTo fin3:
O
O errores:
O ' Tratamiento de errores.....
O Select Case Err
O Case Is = 53: MsgBox " Debe generar la estructura antes de intentar calcularla ",
vbOKOnly: Resume fin2
O Case Is = 100: MsgBox " Pivote excesivamente pequeño, la estructura es un
mecanismo", vbOKOnly: Resume fin2
O Case Is = 101: MsgBox " Está usando un tipo de barra que no está definido ",
vbOKOnly: Resume fin2
O Case Is = 102: MsgBox " Esta versión reducida no admite más de 45 nudos ",
vbOKOnly: Resume fin2
O Case Else: MsgBox " Error sin tratamiento previsto ",
vbOKOnly: On Error GoTo 0
O End Select
O
O
O fin2:
O Close
O Select Case Matr: ' *****
O Case Is = 1: Rem
O Case Is > 1: Kill App.Path + "\matriz.dat"
O End Select
O
O fin3:
O
O gl.Visible = True
O CreaListado
O gl.Caption = "FIN DEL PROCESO"
O
O Comenzar.Enabled = True
O Salir.Enabled = True
O 'SitMatr.Enabled = True: SitMatr.Visible = True
O Param.Enabled = True: Param.Visible = True
O 'ProCalc.Enabled = False: ProCalc.Visible = False
O DoEvents
O End Sub
O
O Sub aux1(iq)
O ' FORMACION DE LA MATRIZ AUXILIAR 1.....
O Dim ii As Long, jj As Long
O
O For ii = iq To NT + iq - 1
O For jj = 1 To NT
O Get #1, (jj + (ii - 1) * NT), a(ii + 1 - iq, jj)
O Next jj: Next ii
O End Sub
O
O Sub aux2(iq)
O ' FORMACION DE LA MATRIZ AUXILIAR 2.....
O Dim jj As Long
O
O For jj = 1 To NT
O Get #1, (jj + (iq - 1) * NT), a(1, jj)
O Next jj
O End Sub
O
O 'DEFDBL A-Z
O Sub Ensambla()
O ' ENSAMBLAJE DE LA MATRIZ GLOBAL DE RIGIDEZ.....
O ' Realiza el ensamblaje de la matriz global de rigidez.
O ' Tiene en cuenta el efecto de pretensado de barras, iteraciones o temperatura
O ' Los datos precisos del programa principal son:
O ' * NE..... n° de barras
O ' * NS..... n° incognitas
O ' * A()..... matriz global de rigidez
O ' * L(NE,C)..... características de las barras
O ' * P(NTP,5)..... tipos de barras
O ' * C(NS)..... coordenadas de los nudos

```

```

O ' * R(NP*ND)..... matriz de cargas externas
O Dim LP As Integer, AR As Double, My As Double, FM As Double
O Dim ie As Long, i As Long, j As Long, iq As Long, jq As Long, k As Long
O Dim n4 As Long, n5 As Long, n7 As Long
O Dim n1 As Long, n2 As Long
O Dim x1 As Double, y1 As Double, z1 As Double
O Dim x2 As Double, y2 As Double, z2 As Double
O Dim xp As Double, yp As Double, zp As Double, Le#
O Dim At As Double
O Dim c1 As Double, c2 As Double, c3 As Double
O Dim t() As Double, SF() As Double, COEF As Double
O
O
O ReDim t(6), SF(6, 6)
O
O For ie = 1 To ne
O   bal.Caption = CStr(ie) '###
O   BarraTiempo PctBa, (ie), (ne) '###
O   DoEvents '###
O   LP = Int(L(ie, 3)): AR = p(LP, 1): My = p(LP, 3)
O   If AR = 0 Then MsgBox "Ojo: area nula en la barra " + CStr(ie), vbOKOnly: Stop
O   If My = 0 Then MsgBox "Ojo: mod. Young nulo en barra " + CStr(ie), vbOKOnly: Stop
O   If My < 0 Then MsgBox "Ojo: Mod. Young variable en barra " + CStr(ie), vbOKOnly:
O     Stop
O   n1 = L(ie, 1): n2 = L(ie, 2)
O   If n1 > n2 Then Swap n1, n2 'numeraci#n de barras de menor a mayor....
O   x1 = C(n1): y1 = C(n1 + np): z1 = C(n1 + 2 * np)
O   x2 = C(n2): y2 = C(n2 + np): z2 = C(n2 + 2 * np)
O   xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
O   Le# = Sqr(xp * xp + yp * yp + zp * zp)
O   ' cosenos directores de los tres ejes locales de la barra.....
O   c1 = xp / Le#: c2 = yp / Le#: c3 = zp / Le#
O
O   ' Matriz de rigidez.....
O   t(1) = -c1: t(2) = -c2: t(3) = -c3: t(4) = c1: t(5) = c2: t(6) = c3
O   ''FM = MY * AR / 1000 / lon0#(ie) 'unidades: ton. y metros....
O   FM = My * AR / 1000 / Le# 'unidades: ton. y metros....
O   For i = 1 To 6: For j = 1 To 6: SF(i, j) = t(i) * FM * t(j): Next j: Next i
O
O   n4 = (n1 - 1) * ND: n5 = (n2 - 1) * ND: n7 = n5 - n4
O
O   ' *** Ensamblaje de matrices Sii.....
O   For i = 1 To 3: For j = i To 3
O     k = j - i + 1
O     Select Case Matr
O       Case Is = 1: a(n4 + i, k) = a(n4 + i, k) + SF(i, j)
O       Case Is > 1
O         Get #1, ((n4 + i - 1) * NT + k), COEF
O         COEF = COEF + SF(i, j)
O         Put #1, ((n4 + i - 1) * NT + k), COEF
O     End Select
O   Next j
O Next i
O
O   ' *** Ensamblaje de matrices Sjj.....
O   For i = 1 To 3: For j = i To 3
O     k = j - i + 1
O     Select Case Matr
O       Case Is = 1: a(n5 + i, k) = a(n5 + i, k) + SF(i + 3, j + 3)
O       Case Is > 1
O         Get #1, ((n5 + i - 1) * NT + k), COEF
O         COEF = COEF + SF(i + 3, j + 3)
O         Put #1, ((n5 + i - 1) * NT + k), COEF
O     End Select
O   Next j
O Next i
O
O   ' *** Ensamblaje de matrices Sij.....
O   For i = 1 To 3: For j = 1 To 3
O     k = j - i + 1
O     Select Case Matr
O       Case Is = 1: a(n4 + i, n7 + k) = a(n4 + i, n7 + k) + SF(i, j + 3)
O       Case Is > 1
O         Get #1, ((n4 + i - 1) * NT + n7 + k), COEF
O         COEF = COEF + SF(i, j + 3)
O         Put #1, ((n4 + i - 1) * NT + n7 + k), COEF
O     End Select
O   Next j: Next i
O Next ie

```

```

O
O
O 'Para tener en cuenta los desplazamientos impuestos (resolución directa)
O For iq = 1 To NS
O   If x(iq) <> 1234 And x(iq) <> 0 Then 'si es cero no es preciso hacer cambios
O     For jq = NT To 1 Step -1
O       If (iq - jq) > 0 Then 'para no salir de la matriz
O         Select Case Matr
O           Case Is = 1: At = a(iq - jq, jq): a(iq - jq, jq) = 0
O           Case Is > 1: Put #1, ((iq - jq - 1) * NT + jq), 0
O         End Select
O       R(iq - jq) = R(iq - jq) - At * x(iq)
O     End If
O   Next jq
O   For jq = 1 To NT
O     If (iq + jq) <= NS Then 'para no salir de la matriz
O       Select Case Matr
O         Case Is = 1: At = a(iq, jq): a(iq, jq) = 0
O         Case Is > 1: Put #1, ((iq - 1) * NT + jq), 0
O       End Select
O     R(iq + jq) = R(iq + jq) - At * x(iq)
O   End If
O Next jq
O End If
O Next iq
O
O End Sub
O
O 'DEFDBL A-Z
O Sub Gauss()
O '   lin,col.....son la línea y columna donde deseamos que nos
O '   imprima el n$ de la fila que se est reduciendo
O '   Si no queremos que se imprima, basta con que
O '   cualquiera de ellas valga 0.
O ' FUNCION: Resuelve un sistema de ecuaciones por el m, todo de
O ' GAUSS-JORDAN los datos precisos del prog. general ser n:
O '   * NS.....1| dimensión matriz a()
O '   * NT.....2| dimensión matriz a()
O '   * a(NS,NT).....matriz de coeficientes
O '   * r(>=NS).....matriz de los t,rminos independientes
O '   * x(NS).....matriz de incógnitas
O '   La matriz a() resulta modificada -para obtener una matriz
O '   triangular- y tambi,n se modifica la matriz r() y la x(), en
O '   esta fltima se almacenar n los valores de las incógnitas.
O '   Si se obtiene un pivote excesivamente pequeño se produce el
O '   error 100
O '   LLAMADA: Gauss (lin, col)
O Dim a1 As Double, a2 As Double, a3 As Double, f As Double
O Dim ir As Long, it As Long, jq As Long, iq As Long
O Dim COEF As Double
O
O Times (0)
O For iq = 1 To NS
O   fil.Caption = CStr(iq) '###
O   BarraTiempo PctFi, (iq), (NS) '###
O   DoEvents '###
O
O   If x(iq) = 1234 Then
O     Select Case Matr
O       Case Is = 1: a1 = a(iq, 1)
O       Case Is = 2: aux1 iq: a1 = a(1, 1)
O       Case Is = 3: aux2 iq: a1 = a(1, 1)
O     End Select
O     If Abs(a1) < 0.000000001 Then Error 100
O     For jq = iq + 1 To iq + NT - 1
O       If (jq <= NS) Then
O         If x(jq) = 1234 Then
O           Select Case Matr
O             Case Is = 1: a2 = a(iq, jq - iq + 1)
O             Case Is = 2: a2 = a(1, jq - iq + 1)
O             Case Is = 3: a2 = a(1, jq - iq + 1)
O           End Select
O           If Abs(a2) > 0.000000001 Then
O             f = -(a2 / a1) ' pivote
O             For ir = 1 To NT
O               If (ir + jq - iq <= NT) Then
O                 it = ir + jq - iq
O                 Select Case Matr

```

```

O           Case Is = 1: a(jq, ir) = a(jq, ir) + f * a(iq, it)
O           Case Is = 2
O             a(jq + 1 - iq, ir) = a(jq + 1 - iq, ir) + f * a(1, it)
O             Put #1, (ir + (jq - 1) * NT), a(jq + 1 - iq, ir)
O           Case Is = 3
O             Get #1, (ir + (jq - 1) * NT), COEF
O             COEF = COEF + f * a(1, it)
O             Put #1, (ir + (jq - 1) * NT), COEF
O           End Select
O         End If
O       Next ir
O       R(jq) = R(jq) + f * R(iq)
O     End If
O   End If
O Next jq
O End If
O   If iq Mod 5 = 0 Then Times (iq) 'actualiza cada 5 iteraciones
O Next iq
O
O ' Cálculo de las incógnitas.....
O For iq = NS To 1 Step -1
O   ig1.Caption = CStr(NS - iq + 1)          '###
O   BarraTiempo PctIq, (NS - iq + 1), (NS) '###
O   DoEvents                                '###
O
O   Select Case Matr
O     Case Is = 1: Rem
O     Case Is > 1: aux2 iq
O   End Select
O   If x(iq) = 1234 Then
O     For jq = 2 To NT
O       it = iq + jq - 1
O       If it <= NS Then
O         Select Case Matr
O           Case Is = 1: a2 = a(iq, jq)
O           Case Is > 1: a2 = a(1, jq)
O         End Select
O         R(iq) = R(iq) - x(it) * a2
O       End If
O     Next jq
O     Select Case Matr
O       Case Is = 1: a3 = a(iq, 1)
O       Case Is > 1: a3 = a(1, 1)
O     End Select
O     x(iq) = R(iq) / a3
O   End If
O Next iq
O
O End Sub
O
O Sub Times(iq)
O 'para estimar la duración de la reducción de filas.
O 'se debe llamar justo antes de empezar las reducciones con iq=0, y despues
O 'se puede llamar varias veces, para afinar la estimación.
O Static Sgini As Single 'segundos llamada inicial
O Dim Sgtot As Single, segFin As Long, minFin As Long, horFin As Long
O
O If iq = 0 Then
O   Sgini = Timer
O Else
O   Sgtot = ((Timer - Sgini) / iq) * NS * 1.3 'duración total de las iteraciones
O   segFin = (Sgini + Sgtot) 'segundo en que acabará el proceso
O   horFin = Int(segFin / 3600): segFin = segFin - horFin * 3600
O   minFin = Int(segFin / 60)
O   HoraFin.Caption = (CStr(horFin) + ":" + CStr(minFin))
O   HoraFin.Caption = (CStr(horFin) + ":" + Right$("00" + CStr(minFin), 2))
O End If
O End Sub
O
O Private Sub LeeDat()
O Dim aa$, i As Long
O
O 'Lectura de datos del disco.....
O DiscoX "datos.dat", "L", "-", iii, sss, ddd, xxx, ttt$
O

```

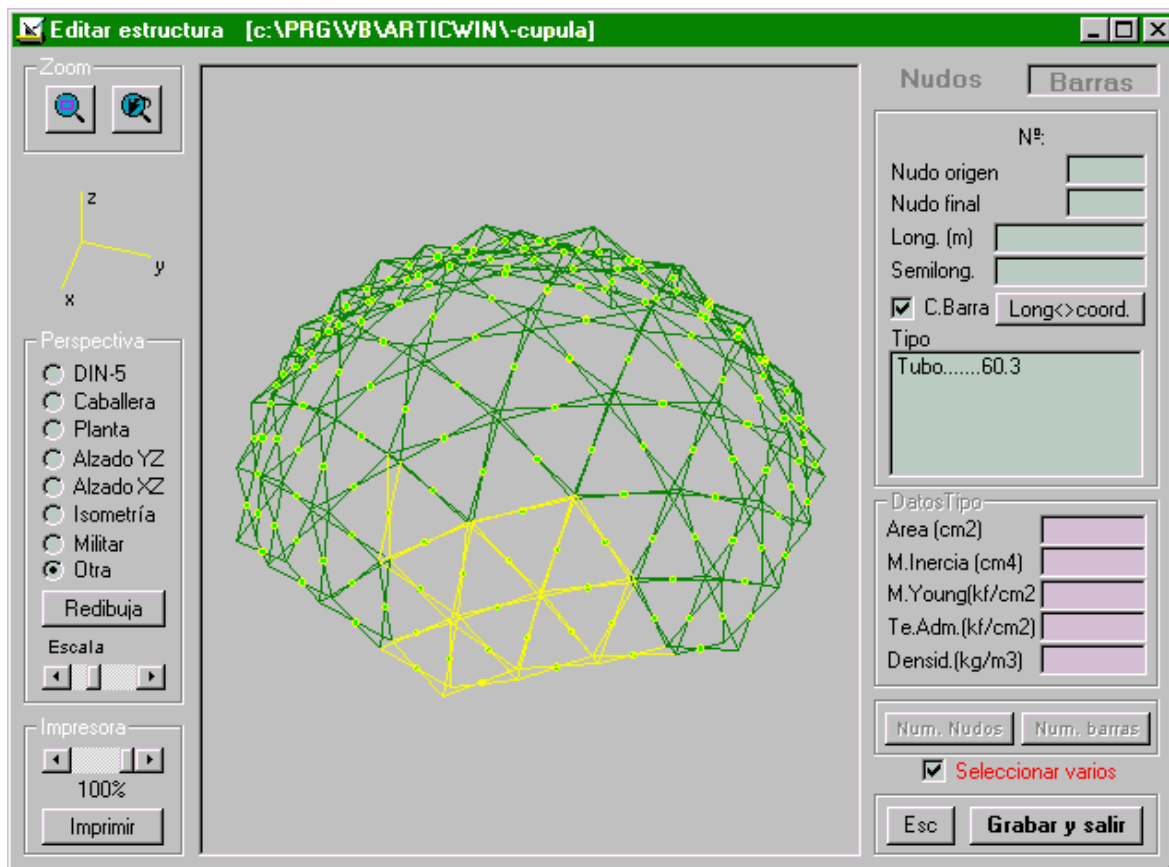








## 7.3.8 MÓDULO DE EDICIÓN DE LA ESTRUCTURA.



```

○ Option Explicit
○ DefDbl A-H
○ DefLng K-N
○ DefDbl O-Z
○ Dim Dire$
○ Dim Itot, AdmAbs, Adm, nchargd, naceld, Ajustabar, ppropio
○ 'Dim np As Long, ne As Long, ntp As Integer
○ 'Dim NR As Long, NS As Long, NV As Long
○ Dim brr() As Long, Cor() As Double, CorL() As Double, C2dnu() As Single, C2dba() As
  Single
○ Dim brrsA() As Long
○ Dim Carg() As Single, Coac() As Single, dinC() As Long, dinA() As Long, lon0#(),
  Slon#()
○ Dim tipotit$, p$(), Tipos() As Single, CoaL() As Single
○ Dim zoo() As Single, Z$
○ Dim distancia As Single, taman As Single
○ Dim hm1 As Single, hm2 As Single, Esc As Single, hx1 As Single, hx2 As Single, hx3 As
  Single, hx4 As Single, hx5 As Single
○ Dim bra As Long, pto As Long
○ Dim dibX As Single, dibY As Single 'pulsación ratón
○ Dim ZoColor As Single
○ Dim SeColor As Single, SeColorb As Single
○ Dim indicetipo As Integer
○ Dim kk$ 'n° de puntos o barras que se pueden añadir
○ Dim pto1 As Long, pto2 As Long 'extremos barra nueva
○ Dim uni(3, 2) As Single 'vectores unitarios X,Y,Z
○ Dim PrtPunto As Boolean
○ Dim tipbH As Integer
○
○
○
○ Function colbarra(tipo As Long) As Long
○   colbarra = QBColor(((tipo) Mod (6)) + 1)
○ End Function
○
○
○ Sub buscarbarra(x As Single, y As Single)
○   Dim aa As Single, bb As Single, encontrado As Boolean
○   Static x0 As Integer, y0 As Integer

```

```

○ If (Abs(x - x0) > distancia) Or (Abs(y - y0) > distancia) Then bra = 0
○ x0 = x: y0 = y
○
○ encontrado = False
○ Do While (bra < ne) And (encontrado = 0)
○   bra = bra + 1
○   aa = Abs(x - C2dba(bra, 1)): bb = Abs(y - C2dba(bra, 2))
○   If aa < distancia And bb < distancia Then
○     encontrado = True
○     FramBar.Enabled = True
○     FramTip.Enabled = True
○     printbarra 1
○   End If
○ Loop
○
○ If encontrado = False Then
○   printbarra 0 'borra datos barra
○   FramBar.Enabled = False
○   FramTip.Enabled = False
○   bra = 0
○ End If
○ End Sub
○
○ Sub buscapunto(x As Single, y As Single)
○ Dim aa, bb, encontrado As Boolean
○ Static x0 As Integer, y0 As Integer
○ If (Abs(x - x0) > distancia) Or (Abs(y - y0) > distancia) Then pto = 0
○ x0 = x: y0 = y
○ encontrado = False
○ Do While (pto < np) And (encontrado = 0)
○   pto = pto + 1
○   aa = Abs(x - C2dnu(pto, 1)): bb = Abs(y - C2dnu(pto, 2))
○   If aa < distancia And bb < distancia Then
○     encontrado = True
○     FramNud.Enabled = True
○     printpunto 1
○   End If
○ Loop
○
○ If encontrado = 0 Then
○   pto = 0
○   printpunto False 'borra datos punto
○   FramNud.Enabled = False
○ End If
○ End Sub
○
○ Sub dibujaba(Pct As Object, Esc As Single)
○ Dim x1 As Single, y1 As Single, i As Integer
○ Dim px, py
○ px = Pct.ScaleWidth: py = Pct.ScaleHeight
○ For i = 1 To ne
○   x1 = C2dba(i, 1) * Esc: y1 = C2dba(i, 2) * Esc
○   'Dib.Line (x1 - taman, y1 - taman)-(x1 + taman, y1 + taman), QBColor(10), BF
○   'Dibuja puntos centrales de las barras
○   If Not ((x1 < 0) Or (x1 > px)) Then
○     If Not ((y1 < 0) Or (y1 > py)) Then
○       Pct.Line (x1 - taman, y1 - taman)-(x1 + taman, y1 + taman), QBColor(10), BF
○       'Dibuja puntos centrales de las barras
○     End If
○   End If
○ Next i
○ End Sub
○ Private Sub DibEjes()
○ Dim x As Single, y As Single, Z As Single, a As Single, b As Single
○ Dim x1 As Single, y1 As Single, x2 As Single, y2 As Single
○ Dim xx As Single, yy As Single, px As Single, py As Single
○ Dim hmlj As Single, hm2j As Single, escj As Single, DiColor As Single, i As Integer
○ Dim m2d(3, 2), ej$(3)
○ px = Editar.Ejes.ScaleWidth
○ py = Editar.Ejes.ScaleHeight
○ DiColor = QBColor(14)
○
○ Ejes.Cls
○ 'paso a 2d...
○ axo3d2dj 0, 0, 1, 0, 0, 0, a, b
○ If a < x1 Then x1 = a
○ If a > x2 Then x2 = a
○ If b < y1 Then y1 = b
○ If b > y2 Then y2 = b

```

```

O axo3d2dj 0, 0, 1, 1, 0, 0, a, b: ej$(1) = "x"
O If a < x1 Then x1 = a
O If a > x2 Then x2 = a
O If b < y1 Then y1 = b
O If b > y2 Then y2 = b
O axo3d2dj 0, 0, 1, 0, 1, 0, a, b: ej$(2) = "y"
O If a < x1 Then x1 = a
O If a > x2 Then x2 = a
O If b < y1 Then y1 = b
O If b > y2 Then y2 = b
O axo3d2dj 0, 0, 1, 0, 0, 1, a, b: ej$(3) = "z"
O If a < x1 Then x1 = a
O If a > x2 Then x2 = a
O If b < y1 Then y1 = b
O If b > y2 Then y2 = b
O
O 'calcula márgenes y escala...
O xx = x2 - x1: yy = y2 - y1 'máximas dimensiones en 2d
O If (xx / px) < (yy / py) Then escj = 0.7 * py / yy Else escj = 0.7 * px / xx 'escala
O hmlj = (px - escj * xx) / 2 - x1 * escj 'centrado imagen
O hm2j = (py - escj * yy) / 2 - y1 * escj
O
O 'calculo dimensiones 2d...
O x = 0: y = 0: Z = 0
O axo3d2dj hmlj, hm2j, escj, x, y, Z, a, b: m2d(0, 1) = a: m2d(0, 2) = b
O x = 1: y = 0: Z = 0
O axo3d2dj hmlj, hm2j, escj, x, y, Z, a, b: m2d(1, 1) = a: m2d(1, 2) = b
O x = 0: y = 1: Z = 0
O axo3d2dj hmlj, hm2j, escj, x, y, Z, a, b: m2d(2, 1) = a: m2d(2, 2) = b
O x = 0: y = 0: Z = 1
O axo3d2dj hmlj, hm2j, escj, x, y, Z, a, b: m2d(3, 1) = a: m2d(3, 2) = b
O
O 'dibuja ejes...
O x1 = m2d(0, 1)
O y1 = m2d(0, 2)
O For i = 1 To 3
O x2 = m2d(i, 1)
O y2 = m2d(i, 2)
O Ejes.Line (x1, y1)-(x2, y2), DiColor
O Ejes.CurrentX = x2 + taman * 3
O Ejes.CurrentY = y2 - taman * 3
O Ejes.Print ej$(i)
O Next i
O End Sub
O
O Sub axo3d2dj(hmlj As Single, hm2j As Single, escj As Single, x As Single, y As Single,
Z As Single, a As Single, b As Single)
O 'x,y,z coordenadas 3d
O ' a,b coordenadas 2d
O 'pasa de 3d a 2d las coordenadas de axonometrías para los ejes.....
O
O a = hmlj + escj * (hx4 * y - hx5 * x)
O b = hm2j - escj * (-hx1 * x - hx2 * y + hx3 * Z)
O
O End Sub
O
O Sub dibujaestr(Pct As Object, Esc As Single)
O 'Dibujo de la estructura.....
O On Error GoTo errores
O Dim i, n1, n2, x1 As Single, x2 As Single, y1 As Single, y2 As Single, DiColor
O Dim px As Single, py As Single
O
O px = Pct.ScaleWidth: py = Pct.ScaleHeight
O Pct.Cls
O
O For i = 1 To ne
O n1 = brr(i, 1): n2 = brr(i, 2)
O x1 = C2dnu(n1, 1) * Esc
O x2 = C2dnu(n2, 1) * Esc
O y1 = C2dnu(n1, 2) * Esc
O y2 = C2dnu(n2, 2) * Esc
O If (Selecci.value = 1) And (C2dba(i, 0) = 1) Then
O DiColor = SeColorb
O Else
O DiColor = colbarra(brr(i, 3))
O End If
O
O 'dibuja línea .....
O If Not ((x1 < 0 And x2 < 0) Or (x1 > px And x2 > px)) Then

```

```

○         If Not ((y1 < 0 And y2 < 0) Or (y1 > py And y2 > py)) Then
○             Pct.Line (x1, y1)-(x2, y2), DiColor
○         End If
○     End If
○
○ Next i
○
○ If Pct Is Printer Then
○     dibujanu Pct, Esc
○     dibuEnlace Pct, Esc
○ Else
○     Select Case FramNud.Visible
○         Case True 'nudos
○             dibujanu Pct, Esc
○         Case Else 'barras
○             dibujaba Pct, Esc
○             dibuEnlace Pct, Esc
○     End Select
○     dibselecc
○ End If
○
○ If 1 = 2 Then
○ errores:
○     If Err = 6 Then 'overflow
○         Err = 0
○         zoomnum 0, 0, 0, 0
○         Resume fin
○     Else
○         MsgBox "Error sin tratamiento previsto", vbCritical: End
○     End If
○ End If
○ fin:
○ End Sub
○
○ Sub dibujanu(Pct As Object, Esc As Single)
○ ' Dibuja puntos
○ Dim x1, y1, i As Long, j As Long
○ Dim px, py, Sc As Single, DiColor As Single
○ 'para dibujo cargas y coacciones...
○ Dim a As Single, b As Single, C As Single, Cx As Single, j2 As Single
○ Dim a2 As Single, b2 As Single, a3 As Single, b3 As Single, a4 As Single, b4 As
○         Single, a5 As Single, b5 As Single
○ Dim CaColor As Single, CoColor As Single
○
○ DiColor = QBColor(11)
○ px = Pct.ScaleWidth: py = Pct.ScaleHeight
○
○ 'dibuja puntos..
○ For i = 1 To np
○     x1 = C2dnu(i, 1) * Esc: y1 = C2dnu(i, 2) * Esc
○     If Not ((x1 < 0) Or (x1 > px)) Then
○         If Not ((y1 < 0) Or (y1 > py)) Then
○             Pct.Line (x1 - taman, y1 - taman)-(x1 + taman, y1 + taman), DiColor, BF
○         End If
○     End If
○ Next i
○
○ 'dibuja cargas...
○ If HScal.value > 0 Then 'si la escala de dibujo de coacciones y cargas es >0
○     CaColor = QBColor(9)
○     CoColor = QBColor(12)
○     Sc = HScal.value / 50
○     For i = 1 To np
○         x1 = C2dnu(i, 1) * Esc: y1 = C2dnu(i, 2) * Esc
○         If Not ((x1 < 0) Or (x1 > px)) Then
○             If Not ((y1 < 0) Or (y1 > py)) Then
○                 ' DibFlecha x1, y1, Carg(i, 1), Carg(i, 2), Carg(i, 3)
○                 For j = 1 To 3
○                     j2 = (j Mod 3) + 1
○                     C = Carg(i, j)
○                     Cx = Coac(i, j)
○                     If C <> 0 Or Cx <> 1234 Then
○                         a2 = uni(j, 1)
○                         b2 = uni(j, 2)
○                         a3 = uni(j, 1) * 1.3 'punta flecha
○                         b3 = uni(j, 2) * 1.3
○                         a4 = uni(j2, 1) * 0.3
○                         b4 = uni(j2, 2) * 0.3
○                         a5 = -uni(j2, 1) * 0.3

```

```

O         b5 = -uni(j2, 2) * 0.3
O         If C <> 0 Then 'dibujo carga...
O             C = Sc * Esc * Sgn(C) * Log(1 + Abs(C)) 'ampliación carga...
O             Pct.Line (x1, y1)-(x1 + a2 * C, y1 + b2 * C), CaColor
O             Pct.Line -(x1 + (a2 + a4) * C, y1 + (b2 + b4) * C), CaColor
O             Pct.Line -(x1 + a3 * C, y1 + b3 * C), CaColor
O             Pct.Line -(x1 + (a2 + a5) * C, y1 + (b2 + b5) * C), CaColor
O             Pct.Line -(x1 + a2 * C, y1 + b2 * C), CaColor
O         End If
O         C = Sc * Esc * 0.4 'ampliación coacción
O         If Cx <> 1234 Then 'dibujo coacción...
O             Pct.Line (x1, y1)-(x1 + (a2 + a4) * C, y1 + (b2 + b4) * C), CoColor
O             Pct.Line -(x1 + (a2 + a5) * C, y1 + (b2 + b5) * C), CoColor
O             Pct.Line -(x1, y1), CoColor
O         End If
O     End If
O     Next j
O End If
O End If
O Next i
O End If
O
O End Sub
O
O Sub longitudes()
O ' Calcula las longitudes segfn coordenadas actuales.....
O Dim x1, x2, y1, y2, z1, z2, n1, n2, Le#, i
O For i = 1 To ne
O     n1 = brr(i, 1): n2 = brr(i, 2)
O     x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
O     x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
O     Le# = Sqr((x2 - x1) ^ 2 + (y2 - y1) ^ 2 + (z2 - z1) ^ 2)
O     lon0#(i) = Le#
O Next i
O Beep
O End Sub
O
O
O
O Sub Paso2D(a1 As Single, a2 As Single, Esc0 As Single)
O 'Calculo de coordenadas para pasar la estructura a dos dimensiones....
O 'px,py = dimensiones zona dibujo
O Dim px As Single, py As Single, x As Single, y As Single, Z As Single
O Dim x1 As Single, y1 As Single, z1 As Single, x2 As Single, y2 As Single, z2 As Single
O Dim xx As Single, yy As Single, i As Long, n1 As Long, n2 As Long
O Dim a As Single, b As Single, ap As Single, bp As Single
O px = Editar.Dib.ScaleWidth
O py = Editar.Dib.ScaleHeight
O hm1 = a1: hm2 = a2: Esc = Esc0
O
O If hm1 = 0 And hm2 = 0 Then 'si son cero se recalcula para que quepa en pantalla
O     x = Cor(1, 1)
O     y = Cor(1, 2):
O     Z = Cor(1, 3) 'valores iniciación.
O     axo3d2d x, y, Z, a, b
O     x1 = a: x2 = a: y1 = b: y2 = b
O
O     For i = 1 To np 'calcula valores máximos y mínimos...
O         x = Cor(i, 1): y = Cor(i, 2): Z = Cor(i, 3)
O         axo3d2d x, y, Z, a, b
O         If a < x1 Then x1 = a
O         If a > x2 Then x2 = a
O         If b < y1 Then y1 = b
O         If b > y2 Then y2 = b
O     Next i
O     xx = x2 - x1: yy = y2 - y1 'máximas dimensiones en 2d
O     If (xx / px) < (yy / py) Then Esc = 0.9 * py / yy Else Esc = 0.9 * px / xx 'escala
O     hm1 = (px - Esc * xx) / 2 - x1 * Esc 'centrado imagen
O     hm2 = (py - Esc * yy) / 2 - y1 * Esc
O End If '.....
O
O For i = 1 To np
O     x = Cor(i, 1): y = Cor(i, 2): Z = Cor(i, 3): axo3d2d x, y, Z, a, b
O     C2dnu(i, 1) = a
O     C2dnu(i, 2) = b
O Next i
O 'Cálculo de los puntos medios de las barras.....
O For i = 1 To ne
O     x = CorL(i, 1): y = CorL(i, 2): Z = CorL(i, 3)

```

```

O     axo3d2d x, y, Z, a, b
O     C2dba(i, 1) = a
O     C2dba(i, 2) = b
O Next i
O 'Vectores unitarios X,Y,Z-----
O axo3d2d 0, 0, 0, a, b 'origen de coordenadas...
O axo3d2d 1, 0, 0, ap, bp: uni(1, 1) = ap - a: uni(1, 2) = bp - b 'vect. unitarios...
O axo3d2d 0, 1, 0, ap, bp: uni(2, 1) = ap - a: uni(2, 2) = bp - b
O axo3d2d 0, 0, 1, ap, bp: uni(3, 1) = ap - a: uni(3, 2) = bp - b
O '-----
O End Sub
O
O
O
O
O Sub printbarra(ver As Integer)
O 'Visualiza por pantalla los valores de los diversos parametros.....
O 'ver=1 imprime datos barra
O 'ver=0 borra datos barra
O Dim i As Integer, Tip As Integer, j As Integer
O If ver = 1 Then
O     'FramBar.Enabled = True
O     'FramTip.Enabled = True
O
O     bnum.Caption = Format$(bra, "####")
O     nuof(1).Text = Format$(brr(bra, 1), "#####")
O     nuof(2).Text = Format$(brr(bra, 2), "#####")
O     lonb.Text = Format$(lon0#(bra), "##0.00000000")
O     Slong.Text = Format$(Slon#(bra), "##0.00000000")
O     Tip = brr(bra, 3)
O     tipb.ListIndex = Tip - 1
O     For j = 1 To 5
O         TipoDat(j).Text = CStr(Tipos(Tip, j))
O     Next j
O Else
O     bnum.Caption = ""
O     nuof(1).Text = ""
O     nuof(2).Text = ""
O     lonb.Text = ""
O     Slong.Text = ""
O     tipb.ListIndex = -1
O     For j = 1 To 5
O         TipoDat(j).Text = ""
O     Next j
O
O     'FramBar.Enabled = False
O     'FramTip.Enabled = False
O End If
O End Sub
O
O
O
O Sub printpunto(ver As Integer)
O 'Visualiza por pantalla los valores de los diversos parametros.....
O 'ver=1 imprime datos nudo
O 'ver=0 borra datos nudo
O Dim aa As Single, i As Integer
O PrtPunto = True 'se está ejecutando el proceso printpunto
O If ver = 1 Then
O     'FramNud.Enabled = True
O     nnum = Format$(pto, "####")
O     For i = 1 To 3
O         crdn(i) = Format$(Cor(pto, i), "####0.000000")
O     Next
O     For i = 1 To 3
O         crga(i) = Format$(Carg(pto, i), "####0.000")
O         If Coac(pto, i) <> 1234 Then cccn(i) = Format$(Coac(pto, i), "###0.000") Else
O             cccn(i) = "Libre"
O         Next
O     'coacciones dinámica...
O     For i = 4 To 9
O         cccn(i) = Format$(Coal(pto, i - 3), "###0.000")
O     Next
O     'grbn.Value = din%(pto, 3)
O     For i = 1 To 3
O         card(i).ListIndex = (dinC(pto, i)) + 3
O         aced(i).ListIndex = (dinA(pto, i)) + 3
O     Next i
O Else
O     nnum = ""

```

```

O     For i = 1 To 3
O         crdn(i) = ""
O     Next
O     For i = 1 To 3
O         crga(i) = ""
O         cccn(i) = ""
O     Next
O     For i = 4 To 9
O         cccn(i) = ""
O     Next
O     'grbn.Value = 0
O     For i = 1 To 3
O         card(i).ListIndex = -1
O         aced(i).ListIndex = -1
O     Next i
O     'FramNud.Enabled = False
O End If
O PrtPunto = False
O End Sub
O
O
O
O Sub zoomnum(i, hm1p, hm2p, escp)
O ' i= 1 zoom adelante, guarda datos
O ' i=-1 zoom atrás, recupera datos
O ' i= 0 Reinicia a cero
O Static num 'posición del zoom actual
O 'SHARED zoo(), hm1, hm2, esc 'hm1 hm2 y esc de cada zoom
O Select Case i
O     Case 1
O         If num < 30 Then
O             num = num + 1
O             zoo(num, 1) = hm1: zoo(num, 2) = hm2: zoo(num, 3) = Esc
O             hm1 = hm1p: hm2 = hm2p: Esc = escp
O         End If
O     Case -1
O         If num > 0 Then
O             hm1 = zoo(num, 1): hm2 = zoo(num, 2): Esc = zoo(num, 3)
O             num = num - 1
O         End If
O     Case 0
O         num = 0
O         hm1 = 0: hm2 = 0: Esc = 1
O End Select
O Paso2D hm1, hm2, Esc
O dibujaestr Dib, 1
O End Sub
O
O
O
O Sub grab()
O ReDim NA(np), NN(np) 'variables locales
O Dim i As Integer, j As Integer, Z$
O Dim E1, aa, bb, cc, NM, P1, P2, pp, a$
O DiscoX "TIPOS.DAT", "G", "SX", iii, Tipos(), ddd, p$(), tipotit$
O 'Eliminación puntos aislados y barras....
O E1 = 0
O For i = 1 To ne ' Se anotan los ptos. extremos de cada barra.
O     If brr(i, 1) <> brr(i, 2) Then 'si nudo or=extr se elimina la barra
O         E1 = E1 + 1 'E1= nfmero total de barras
O         aa = brr(i, 1): bb = brr(i, 2)
O         NA(aa) = aa: NA(bb) = bb
O     End If
O Next i
O P1 = 0: P2 = 0
O Do
O     P2 = P2 + 1
O     If NA(P2) <> 0 Then P1 = P1 + 1: NM = NA(P2): NA(P2) = 0: NA(P1) = NM: NN(NM) = P1
O Loop While P2 < np
O np = P1 'número de puntos realmente existente
O
O '-----
O ' Grabación...
O DiscoX "datos.dat", "G", "-", iii, sss, ddd, xxx, ttt$
O 'datos referentes a barras.....
O a$ = nom$ + "\BARRAS.DAT": Open a$ For Output As #1
O a$ = nom$ + "\LONGITUD.DAT": Open a$ For Output As #2
O a$ = nom$ + "\SLONGIT.DAT": Open a$ For Output As #3
O a$ = nom$ + "\COORDENL.DAT": Open a$ For Output As #4

```

```

O For i = 1 To ne
O If brr(i, 1) <> brr(i, 2) Then
O aa = NN(brr(i, 1)): bb = NN(brr(i, 2))
O brr(i, 0) = 1 'marca las barras que se mantienen, las eliminadasn serán cero.
O 'If aa > bb Then Swap a, b ' cc = aa: aa = bb: bb = cc
O Write #1, aa
O Write #1, bb
O Write #1, brr(i, 3)
O Print #2, FormatP(lon0#(i), " 0.0000000E+00")
O Print #3, FormatP(Slon#(i), " 0.0000000E+00")
O For j = 1 To 3
O Print #4, FormatP(CorL(i, j), " 0.0000000E+00")
O Next j
O End If
O Next i
O Close #1
O Close #2
O Close #3
O Close #4
O 'enlaces barras.....
O a$ = nom$ + "\BARRASA.DAT": Open a$ For Output As #1
O pp = 0
O For i = 1 To nea
O aa = brrsA(i, 1): bb = brrsA(i, 2)
O If brr(aa, 0) = 1 And brr(bb, 0) = 1 Then 'solo si las dos barras extremas existen
O pp = pp + 1
O Write #1, aa
O Write #1, bb
O End If
O Next i
O Close #1
O nea = pp 'n° de enlaces realmente existentes
O 'datos generales.....
O a$ = nom$ + "\DATOS.DAT": Open a$ For Output As #1
O Write #1, np
O Write #1, E1
O Write #1, nea, Itot, AdmAbs, Adm, ncargd, naceld, Ajustabar, PPropio
O Close #1
O 'datos referentes a puntos.....
O a$ = nom$ + "\COORDENA.DAT": Open a$ For Output As #1
O For j = 1 To 3: For i = 1 To np
O pp = NA(i): Print #1, FormatP(Cor(pp, j), " 0.000000000E+00")
O Next i: Next j
O Close #1
O a$ = nom$ + "\CARGAS.DAT": Open a$ For Output As #1
O For i = 1 To np: For j = 1 To 3
O pp = NA(i): Print #1, FormatP(Carg(pp, j), " 0.000000E+00")
O Next j: Next i
O Close #1
O a$ = nom$ + "\COACCION.DAT": Open a$ For Output As #1
O For i = 1 To np: For j = 1 To 3
O pp = NA(i): Write #1, Coac(pp, j)
O Next j: Next i
O Close #1
O a$ = nom$ + "\COACCLMT.DAT": Open a$ For Output As #1
O For i = 1 To np: For j = 1 To 3
O pp = NA(i)
O Print #1, FormatP(CoaL(pp, j), " 0.0000E+00") + FormatP(CoaL(pp, j + 3), "
0.0000E+00")
O Next j: Next i
O Close #1
O a$ = nom$ + "\CARGASD.DAT": Open a$ For Output As #1
O For i = 1 To np
O pp = NA(i)
O For j = 1 To 3
O Write #1, dinC(pp, j)
O Next j: Next i
O Close #1
O a$ = nom$ + "\ACELED.DAT": Open a$ For Output As #1
O For i = 1 To np
O pp = NA(i)
O For j = 1 To 3
O aa = dinA(pp, j)
O If aa = 0 Then aa = 1234
O Write #1, aa
O Next j: Next i
O Close #1
O Crea_Resto_Archivos
O End Sub

```



```

○
○
○
○ Private Sub Actulong_Click()
○ Dim i As Long, n1 As Long, n2 As Long, Le#
○ Dim x1 As Single, y1 As Single, z1 As Single
○ Dim x2 As Single, y2 As Single, z2 As Single
○ Dim xp As Double, yp As Double, zp As Double
○ If Selecci.value = 1 Then 'proceso de selección
○ For i = 1 To ne
○ If C2dba(i, 0) = 1 Then 'barra seleccionada
○ n1 = brr(i, 1): n2 = brr(i, 2)
○ x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
○ x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
○ xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
○ Le# = Sqr(xp * xp + yp * yp + zp * zp)
○ lon0#(i) = Le#
○ Slon#(i) = Le# / 2
○ CorL(i, 1) = x1 + xp / 2: CorL(i, 2) = y1 + yp / 2: CorL(i, 3) = z1 + zp / 2
○ End If
○ Next i
○ Else
○ n1 = brr(bra, 1): n2 = brr(bra, 2)
○ x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
○ x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
○ xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
○ Le# = Sqr(xp * xp + yp * yp + zp * zp)
○ lon0#(bra) = Le#
○ Slon#(bra) = Le# / 2
○ CorL(bra, 1) = x1 + xp / 2: CorL(bra, 2) = y1 + yp / 2: CorL(bra, 3) = z1 + zp / 2
○ End If
○ printbarra 0
○ FramBar.Enabled = False
○ FramTip.Enabled = False
○ End Sub
○
○
○
○ Private Sub Barras_Click()
○ Nudos.BorderStyle = 0
○ Nudos.ForeColor = &H80000011 'disabled text
○ Barras.BorderStyle = 1
○ Barras.ForeColor = &H80FF80 ' &H80000012 'button text
○ FramNud.Visible = False
○ FramNud.Enabled = False
○ DoEvents
○ FramBar.Visible = True
○ FramBar.Enabled = False
○ FramTip.Visible = True
○ FramTip.Enabled = False
○ Dib.Cls
○ dibujaestr Dib, 1
○ End Sub
○
○
○
○ Private Sub cccn_DblClick(index As Integer)
○ If UCase$(cccn(index).Text) = "LIBRE" Then cccn(index).Text = "0" Else
○ cccn(index).Text = "Libre"
○ End Sub
○
○
○
○ Private Sub Form_Resize()
○ Dim aa As Single, Dx As Single, Dy As Single
○ If WindowState <> 1 Then 'no minimizada
○ 'ajustar tamaño form al tamaño de la pantalla.....
○ aa = FrIzq.Left
○ FrDer.Left = Me.Width - FrDer.Width - aa
○ Dx = FrDer.Left - FrIzq.Width - aa * 2.1
○ Dy = Me.ScaleHeight - Dib.Top * 1.9
○ If Dx < 100 Then Dx = 100
○ If Dy < 100 Then Dy = 100
○ Dib.Width = Dx
○ Dib.Height = Dy
○ zoomnum 0, 0, 0, 0: DibEjes
○ '.....
○ End If
○ End Sub

```

```

○
○
○
○ Private Sub Guardar_Click()
○ DoEvents
○ grab
○ End Sub
○
○
○
○ Private Sub HPrn_Change()
○ LbPorc.Caption = Str$(HPrn.value) + "%"
○ End Sub
○
○
○
○ Private Sub Imprimir_Click()
○ Dim Fe As Single 'factor de escala
○ Dim px As Single, py As Single, pxd As Single, pyd As Single, Esc As Single
○ Dim taman2 As Single
○ Fe = HPrn.value / 100
○ pxd = Dib.ScaleWidth: pyd = Dib.ScaleHeight
○ px = Printer.ScaleWidth: py = Printer.ScaleHeight
○ If (pxd / px) < (pyd / py) Then Esc = py / pyd Else Esc = px / pxd 'escala
○ Esc = Esc * Fe
○ taman2 = taman: taman = taman * Esc * 0.9
○ dibujaestr Printer, Esc
○ Printer.EndDoc
○ taman = taman2
○ End Sub
○
○
○
○ Private Sub Redibuja_Click()
○ Paso2D hm1, hm2, Esc
○ Dib.Cls
○ dibujaestr Dib, 1
○ End Sub
○
○
○
○ Private Sub cccn_LostFocus(index As Integer)
○ Dim i As Long, valor As Single
○ If cccn(index) = "" Then Exit Sub
○ Select Case index
○ Case 1 To 3
○ If UCase$(LTrim$(RTrim$(cccn(index)))) = "LIBRE" Then valor = 1234 Else valor =
○ ValP(cccn(index))
○ If Selecci.value = 1 Then 'proceso de selección
○ For i = 1 To np
○ If C2dnu(i, 0) = 1 Then 'punto seleccionado
○ Coac(i, index) = valor
○ End If
○ Next i
○ Else
○ Coac(pto, index) = valor
○ End If
○ Case 4 To 9
○ valor = ValP(cccn(index))
○ If Selecci.value = 1 Then 'proceso de selección
○ For i = 1 To np
○ If C2dnu(i, 0) = 1 Then 'punto seleccionado
○ CoaL(i, index - 3) = valor
○ End If
○ Next i
○ Else
○ CoaL(pto, index - 3) = valor
○ End If
○ End Select
○ End Sub
○
○
○ Private Sub crdn_LostFocus(index As Integer)
○ Dim valor As Single, a As Double
○ Dim i As Long, n1 As Long, n2 As Long, Le#
○ Dim x1 As Single, y1 As Single, z1 As Single
○ Dim x2 As Single, y2 As Single, z2 As Single
○ Dim xp As Double, yp As Double, zp As Double
○
○ If crdn(index) = "" Then Exit Sub

```

```

O valor = ValP(crdn(index))
O If Selecci.value = 1 Then 'proceso de selección
O   For i = 1 To np
O     If C2dnu(i, 0) = 1 Then 'punto seleccionado
O       Cor(i, index) = valor
O     End If
O   Next i
O Else
O   Cor(pto, index) = valor
O End If
O
O For i = 1 To ne 'actualiza coord. puntos centrales barras.
O   n1 = brr(i, 1): n2 = brr(i, 2)
O   x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
O   x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
O   xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
O   Le# = Sqr(xp * xp + yp * yp + zp * zp)
O   a = Slon#(i)
O   If (a > 0.75 * Le#) Or (a < 0.25 * Le#) Then a = Le# / 2 'para que no quede muy
O     excéntrico...
O   x2 = xp / Le# * a: y2 = yp / Le# * a: z2 = zp / Le# * a
O   CorL(i, 1) = x1 + x2: CorL(i, 2) = y1 + y2: CorL(i, 3) = z1 + z2
O Next i
O End Sub
O
O
O Private Sub crga_LostFocus(index As Integer)
O Dim i As Long, valor As Single
O If crga(index) = "" Then Exit Sub
O valor = ValP(crga(index))
O If Selecci.value = 1 Then 'proceso de selección
O   For i = 1 To np
O     If C2dnu(i, 0) = 1 Then 'punto seleccionado
O       Carg(i, index) = (valor)
O     End If
O   Next i
O Else
O   Carg(pto, index) = (valor)
O End If
O End Sub
O
O
O Private Sub card_Click(index As Integer)
O Dim i As Long
O If PrtPunto = False Then 'no está en el proceso printpunto
O   If Selecci.value = 1 Then 'proceso de selección
O     For i = 1 To np
O       If C2dnu(i, 0) = 1 Then 'punto seleccionado
O         dinC(i, index) = card(index).ListIndex - 3
O       End If
O     Next i
O   Else
O     dinC(pto, index) = card(index).ListIndex - 3
O   End If
O End If
O End Sub
O
O
O Private Sub aced_Click(index As Integer)
O Dim i As Long
O If PrtPunto = False Then 'no está en el proceso printpunto
O   If Selecci.value = 1 Then 'proceso de selección
O     For i = 1 To np
O       If C2dnu(i, 0) = 1 Then 'punto seleccionado
O         dinA(i, index) = aced(index).ListIndex - 3
O       End If
O     Next i
O   Else
O     dinA(pto, index) = aced(index).ListIndex - 3
O   End If
O End If
O End Sub
O
O Private Sub Dib_MouseDown(Button As Integer, Shift As Integer, x As Single, y As
O   Single)

```

```

O Dim pp As Long
O Dib.ToolTipText = ""
O Select Case FramNud.Visible 'para almacenar valores cambiados
O   Case True
O     crdn_LostFocus 1
O     crdn_LostFocus 2
O     crdn_LostFocus 3
O     crga_LostFocus 1
O     crga_LostFocus 2
O     crga_LostFocus 3
O     cccn_LostFocus 1
O     cccn_LostFocus 2
O     cccn_LostFocus 3
O   Case False
O     nuof_LostFocus (1)
O     nuof_LostFocus (2)
O     lonb_LostFocus
O     TipoDat_LostFocus (1)
O     TipoDat_LostFocus (2)
O     TipoDat_LostFocus (3)
O     TipoDat_LostFocus (4)
O     TipoDat_LostFocus (5)
O End Select
O Select Case Button
O   Case vbLeftButton 'botón izquierdo
O     If Selecci.value = 0 And ZoomV1.Visible Then
O       Select Case FramNud.Visible
O         Case True 'nudos
O           buscapunto x, y
O         Case False 'barras
O           buscarbarra x, y
O       End Select
O     End If
O   Case vbRightButton 'si pulsamos el botón derecho...
O
O     If pp <> 0 Then ptol = pp
O End Select
O dibX = x: dibY = y 'guarda coordenadas pulsación ratón
O Dib.CurrentX = x
O Dib.CurrentY = y
O End Sub
O
O
O Private Sub Dib_MouseMove(Button As Integer, Shift As Integer, x As Single, y As
Single)
O Dim DMInicial As Integer, ReColor As Single
O If Button = 0 Then Exit Sub 'Si no está pulsado algún botón del ratón
O
O   DMInicial = Dib.DrawMode
O   Dib.DrawMode = 7 'Xor
O Select Case Button
O   Case vbLeftButton 'botón izquierdo
O     ReColor = 0
O     If Selecci.value = 1 Then ReColor = SeColor
O     If ZoomV2.Enabled Then ReColor = ZoColor 'predomina el zoom
O     If ReColor <> 0 Then
O       'Dibujar cajas
O       If 1 = 1 Then
O         Dib.Line (dibX, dibY)-(Dib.CurrentX, Dib.CurrentY), ReColor, B
O         Dib.Line (dibX, dibY)-(x, y), ReColor, B
O       End If
O     End If
O   Case vbRightButton 'si pulsamos el botón derecho...
O
O     Dib.Line (dibX, dibY)-(Dib.CurrentX, Dib.CurrentY), SeColor
O     Dib.Line (dibX, dibY)-(x, y), SeColor
O   End If
O End Select
O Dib.DrawMode = DMInicial
O End Sub
O
O
O Private Sub Dib_MouseUp(Button As Integer, Shift As Integer, x As Single, y As Single)
O Dim x1 As Single, x2 As Single, y1 As Single, y2 As Single
O Dim xx As Single, yy As Single, i As Long
O Dim hmlp, hm2p, escp
O
O Dim DMInicial As Integer, STInicial As Integer
O Dim aa, bb, DiColor As Single

```

```

O Dim n1 As Long, n2 As Long, px, py, pp As Long
O Dim bx1, bx2, by1, by2
O Select Case Button
O Case vbLeftButton 'botón izquierdo
O
O   x1 = dibX: y1 = dibY
O   x2 = x: y2 = y
O   'ordena de menor a mayor...
O   If x1 > x2 Then aa = x2: x2 = x1: x1 = aa
O   If y1 > y2 Then aa = y2: y2 = y1: y1 = aa
O   If ZoomV2.Enabled Then
O     If (x2 - x1) < distancia Then x1 = x2 - distancia * 20: x2 = x2 + distancia * 20
O     If (y2 - y1) < distancia Then y1 = y2 - distancia * 20: y2 = y2 + distancia * 20
O     x1 = (x1 - hm1) / Esc   'coordenadas sin escalar
O     x2 = (x2 - hm1) / Esc
O     y1 = (y1 - hm2) / Esc
O     y2 = (y2 - hm2) / Esc
O     xx = x2 - x1: yy = y2 - y1 'máximas dimensiones en 2d
O     If (xx / px) < (yy / py) Then
O       escp = py / yy
O     Else
O       escp = px / xx
O     End If
O     hm1p = (px - escp * xx) / 2 - x1 * escp 'centrado imagen
O     hm2p = (py - escp * yy) / 2 - y1 * escp
O     zoomnum 1, hm1p, hm2p, escp
O     ZoomV2_MouseDown 0, 0, 0, 0
O   Else
O     If Selecci.value = 1 Then
O       DMIinicial = Dib.DrawMode
O       Dib.DrawMode = 7 'Xor
O       Dib.Line (dibX, dibY)-(Dib.CurrentX, Dib.CurrentY), SeColor, B 'borra ventana
O         selección
O       Dib.DrawMode = DMIinicial
O       If (x2 - x1) < distancia Then x1 = x2 - distancia: x2 = x2 + distancia
O       If (y2 - y1) < distancia Then y1 = y2 - distancia: y2 = y2 + distancia
O       If FramNud.Visible = True Then 'nudos.....
O         DMIinicial = Dib.DrawMode
O         STInicial = Dib.FillStyle
O         Dib.DrawMode = 7 'Xor
O         Dib.FillStyle = 0 'continuo
O         Dib.FillColor = SeColor 'Color selección
O         i = 1
O         Do While i <= np
O           aa = C2dnu(i, 1): bb = C2dnu(i, 2)
O           If aa > x1 And aa < x2 Then
O             If bb > y1 And bb < y2 Then
O               C2dnu(i, 0) = Abs(C2dnu(i, 0) - 1) 'alterna valor
O               Dib.Circle (aa, bb), taman * 6, SeColor 'dibuja o borra círculo
O             End If
O           End If
O           i = i + 1
O         Loop
O         Dib.DrawMode = DMIinicial
O         Dib.FillStyle = STInicial 'estilo inicial
O         printpunto 0 'borra datos punto
O       Else 'barras.....
O         i = 1
O         Do While i <= ne
O           aa = C2dba(i, 1): bb = C2dba(i, 2)
O           If aa > x1 And aa < x2 Then
O             If bb > y1 And bb < y2 Then
O               n1 = brr(i, 1): n2 = brr(i, 2)
O               bx1 = C2dnu(n1, 1): bx2 = C2dnu(n2, 1)
O               by1 = C2dnu(n1, 2): by2 = C2dnu(n2, 2)
O               C2dba(i, 0) = Abs(C2dba(i, 0) - 1)
O               If C2dba(i, 0) = 1 Then DiColor = SeColor Else DiColor =
O                 colbarra(brr(i, 3))
O               Dib.Line (bx1, by1)-(bx2, by2), DiColor
O             End If
O           End If
O           i = i + 1
O         Loop
O         printbarra 0 'borra datos barra
O       End If
O     End If
O   End If
O Case vbRightButton 'botón derecho
O   'añadir o eliminar barras...

```

```

O      If ptol <> 0 Then
O          locpunto x, y, pp
O          OtraBarra (ptol), pp
O          ptol = 0
O      End If
O  End Select
O  End Sub
O
O
O  Private Sub dibselecc()
O      Dim DMInicial As Integer, STInicial As Integer
O      Dim aa, bb, DiColor As Single
O      Dim i As Long
O      If Selecci.value = 1 Then
O          If FramNud.Visible = True Then 'nudos.....
O              DMInicial = Dib.DrawMode
O              STInicial = Dib.FillStyle
O              Dib.DrawMode = 7 'Xor
O              Dib.FillStyle = 0 'continuo
O              Dib.FillColor = SeColor 'Color selección
O              i = 1
O              Do While i <= np
O                  If C2dnu(i, 0) = 1 Then
O                      aa = C2dnu(i, 1): bb = C2dnu(i, 2)
O                      Dib.Circle (aa, bb), taman * 6, SeColor 'dibuja o borra círculo
O                  End If
O                  i = i + 1
O              Loop
O              Dib.FillStyle = STInicial 'estilo inicial
O              Dib.DrawMode = DMInicial
O          End If
O      End If
O  End Sub
O
O
O
O  Private Sub Form_Activate()
O      Me.Caption = "Editar estructura" + " [" + nom$ + "]"
O      Perspec_Click (7)
O      Nudos_Click
O  End Sub
O
O
O
O  Private Sub Form_Load()
O      Dim a$, i, j, aa
O      'Lectura de datos
O      Dire$ = nom$ + "\"
O      PrtPunto = False
O      DiscoX "datos.dat", "L", "-", iii, sss, ddd, xxx, ttt$
O      For i = 1 To 3
O          card(i).Clear
O          card(i).AddItem "Tabla 1"
O          card(i).AddItem "Tabla 2"
O          card(i).AddItem "Tabla 3"
O          card(i).AddItem "Ninguna"
O          card(i).AddItem "Rampa 1"
O          card(i).AddItem "Rampa 2"
O          card(i).AddItem "Rampa 3"
O          card(i).AddItem "Seno 1"
O          card(i).AddItem "Seno 2"
O          card(i).AddItem "Seno 3"
O          aced(i).Clear
O          aced(i).AddItem "Tabla 1"
O          aced(i).AddItem "Tabla 2"
O          aced(i).AddItem "Tabla 3"
O          aced(i).AddItem "Libre "
O          aced(i).AddItem "Rampa 1"
O          aced(i).AddItem "Rampa 2"
O          aced(i).AddItem "Rampa 3"
O          aced(i).AddItem "Seno 1"
O          aced(i).AddItem "Seno 2"
O          aced(i).AddItem "Seno 3"
O      Next i
O      tipb.Clear
O      DiscoX "TIPOS.DAT", "L", "SX", iii, Tipos(), ddd, p$(i), tipotit$
O      For i = 1 To ntp
O          tipb.AddItem p$(i)
O      Next i

```

```

O   kk$ = "100" 'número de puntos o barra máximo que podemos introducir en una sesión...
O   ReDim C2dnu(np + Val(kk$), 2), C2dba(ne + Val(kk$), 2)
O   ReDim lon0#(ne + Val(kk$)) ', din%(np + Val(kk$), 3)
O   DiscoX "BARRAS.DAT", "L", "LA", brr(), sss, ddd, xxx, kk$
O   If nea > 0 Then
O     DiscoX "BARRAS.DAT", "L", "L", brrsA(), sss, ddd, xxx, kk$
O   End If
O   DiscoX "CARGAS.DAT", "L", "SA", iii(), Carg(), ddd(), xxx(), kk$
O   DiscoX "LONGITUD.DAT", "L", "D", iii, sss, lon0#(), xxx, kk$
O   DiscoX "SLONGIT.DAT", "L", "D", iii, sss, Slon#(), xxx, kk$
O   DiscoX "CARGASD.DAT", "L", "L", dinC(), sss, ddd, xxx, kk$
O   DiscoX "ACELED.DAT", "L", "L", dinA(), sss, ddd, xxx, kk$
O   DiscoX "COACCION.DAT", "L", "SA", iii, Coac(), ddd, xxx, kk$
O   DiscoX "COACCLMT.DAT", "L", "SA", iii, CoaL(), ddd, xxx, kk$
O   DiscoX "COORDENA.DAT", "L", "D3A", iii, sss, Cor(), xxx, kk$
O   DiscoX "COORDENL.DAT", "L", "D", iii, sss, CorL(), xxx, kk$
O   'Inicialización.....
O   distancia = 45 'n° de pixels de distancia a los que se puede seleccionar un punto
O   taman = 12 'tamaño del los puntos que se dibujan
O   SeColor = RGB(90, 90, 255) 'color caja selección: amarillo (dibujo con XOR)
O   SeColorb = QBColor(14) 'color barras seleccionadas: amarillo
O   ZoColor = QBColor(10) 'RGB(130, 130, 0) 'color caja zoom: magenta
O   ReDim zoc(30, 3)
O   pto = 0: bra = 0
O   hx1 = 0.7: hx2 = 0.22: hx3 = 0.75: hx4 = 1: hx5 = 0.3 'Otra
O   Call Paso2D(0, 0, 1)
O   buscabarra -1000, -1000
O   buscapunto -1000, -1000
O   tipbH = tipb.Height
O   End Sub
O
O
O
O   Sub axo3d2d(x As Single, y As Single, z As Single, a As Single, b As Single)
O   'x,y,z coordenadas 3d
O   ' a,b coordenadas 2d
O   'pasa de 3d a 2d las coordenadas de axonometrías.....
O   a = hml + Esc * (hx4 * y - hx5 * x)
O   b = hm2 - Esc * (-hx1 * x - hx2 * y + hx3 * z)
O   End Sub
O
O
O
O   Private Sub Form_Unload(Cancel As Integer)
O   Menu.Show
O   End Sub
O
O
O
O   Private Sub lonb_LostFocus()
O   Dim Vlonb As Double, i As Long
O   If lonb = "" Then Exit Sub
O   Vlonb = ValP(lonb)
O   If Vlonb > 0 Then
O     If Selecci.value = 1 Then 'proceso de selección
O       For i = 1 To ne
O         If C2dba(i, 0) = 1 Then 'barra seleccionada
O           lon0#(bra) = (Vlonb)
O         End If
O       Next i
O     Else
O       lon0#(bra) = (Vlonb)
O     End If
O   End If
O   End Sub
O
O
O
O   Private Sub Nudos_Click()
O   Dim i As Long
O   Barras.BorderStyle = 0
O   Barras.ForeColor = &H80000011 'disabled text
O   Nudos.BorderStyle = 1
O   Nudos.ForeColor = &HFFFFFF80 ' &H80000012 'button text
O   FramBar.Visible = False
O   FramBar.Enabled = False
O   FramTip.Visible = False
O   FramTip.Enabled = False
O   DoEvents

```

```

○ FramNud.Visible = True
○ FramNud.Enabled = False
○ Dib.Cls
○ dibujaestr Dib, 1
○ End Sub
○
○
○
○ Private Sub Numeraba_Click()
○ 'Numera barras.....
○ Dim i As Long, a$
○ Dim xw, yw
○ Dib.FontSize = 6
○ Dib.FontName = "Courier New"
○ Dib.FontSize = 6
○ Dib.ForeColor = QBColor(10)
○ For i = 1 To ne
○   a$ = LTrim$(Str$(i))
○   xw = Int(C2dba(i, 1) - taman * 3)
○   yw = Int(C2dba(i, 2) - taman * 3)
○   If xw > 0 And xw < (Dib.ScaleWidth) Then
○     If yw > 0 And yw < (Dib.ScaleHeight) Then
○       Dib.Line (xw, yw)-(xw + taman * Len(a$) * 6, yw + taman * 7), QBColor(2), BF
○       Dib.CurrentX = xw
○       Dib.CurrentY = yw - taman * 2
○       Dib.Print a$
○     End If
○   End If
○ Next i
○ End Sub
○
○
○
○ Private Sub Numeranu_Click()
○ 'Numera nudos.....
○ Dim i As Long, a$
○ Dim xw, yw
○ Dib.FontSize = 6
○ Dib.FontName = "Courier New"
○ Dib.FontSize = 6
○ Dib.ForeColor = QBColor(11)
○ For i = 1 To np
○   a$ = LTrim$(Str$(i))
○   xw = Int(C2dnu(i, 1) - taman * 3)
○   yw = Int(C2dnu(i, 2) - taman * 3)
○   If xw > 0 And xw < (Dib.ScaleWidth) Then
○     If yw > 0 And yw < (Dib.ScaleHeight) Then
○       Dib.Line (xw, yw)-(xw + taman * Len(a$) * 6, yw + taman * 7), QBColor(1), BF
○       Dib.CurrentX = xw
○       Dib.CurrentY = yw - taman * 2
○       Dib.Print a$
○     End If
○   End If
○ Next i
○ End Sub
○
○
○
○ Private Sub nuof_LostFocus(index As Integer)
○ Dim Vnuof As Long, i As Long
○ If nuof(index) = "" Then Exit Sub
○ Vnuof = ValP(nuof(index))
○ If Vnuof > 0 And Vnuof <= np Then
○   If Selecci.value = 1 Then 'proceso de selección
○     For i = 1 To ne
○       If C2dba(i, 0) = 1 Then 'barra seleccionada
○         brr(i, index) = (Vnuof)
○       End If
○     Next i
○   Else
○     brr(bra, index) = (Vnuof)
○   End If
○ End If
○ End Sub
○
○
○
○ Private Sub Perspec_Click(index As Integer)
○ 'Selección del tipo de perspectiva a realizar

```



```

○ Dim n As Integer
○ Select Case index
○   Case 0: hx1 = 0.395: hx2 = 0.124: hx3 = 0.55: hx4 = 0.496: hx5 = 0.225 'DIN-5
○   Case 1: hx1 = 0.5: hx2 = 0: hx3 = 1: hx4 = 1: hx5 = 0.5 'Caballera
○   Case 2: hx1 = 0: hx2 = -1: hx3 = 0: hx4 = 0: hx5 = -1 'Planta
○   Case 3: hx1 = 0: hx2 = 0: hx3 = 1: hx4 = 1: hx5 = 0 'Alzado YZ
○   Case 4: hx1 = 0: hx2 = 0: hx3 = 1: hx4 = 0: hx5 = -1 'Alzado XZ
○   Case 5: hx1 = 0.5: hx2 = 0.5: hx3 = 1: hx4 = 0.866: hx5 = 0.866 'Isometria
○   Case 6: hx1 = 0.866: hx2 = 0.5: hx3 = 0.5: hx4 = 0.866: hx5 = 0.5 'Militar
○   Case 7: hx1 = 0.7: hx2 = 0.22: hx3 = 0.75: hx4 = 1: hx5 = 0.3 'Otra
○ End Select
○ 'kx4 = kx4 * dfx: kx5 = kx5 * dfy: ' Correcciones por la deformacion de la pantalla.
○ zoomnum 0, 0, 0, 0
○ DibEjes
○ End Sub
○
○
○
○ Private Sub Salir_Click()
○ Unload Me
○ End Sub
○
○
○
○ Private Sub Selecci_Click()
○ Dim i As Long
○ If Selecci.value = 1 Then
○   Nudos.Enabled = False
○   NumeraNu.Enabled = False
○   For i = 1 To np
○     C2dnu(i, 0) = 0
○   Next i
○   Barras.Enabled = False
○   Numeraba.Enabled = False
○   For i = 1 To ne
○     C2dba(i, 0) = 0
○   Next i
○   Select Case FramNud.Visible
○     Case True
○       printpunto (0)
○       FramNud.Enabled = True
○     Case False
○       printbarra (0)
○       FramBar.Enabled = True
○       FramTip.Enabled = False
○     End Select
○ Else
○   Nudos.Enabled = True
○   NumeraNu.Enabled = True
○   Barras.Enabled = True
○   Numeraba.Enabled = True
○   Select Case FramNud.Visible
○     Case True
○       printpunto (0)
○       FramNud.Enabled = False
○     Case False
○       printbarra (0)
○       FramBar.Enabled = False
○       FramTip.Enabled = False
○     End Select
○   dibujaestr Dib, 1
○ End If
○ End Sub
○
○
○
○ Private Sub Slong_LostFocus()
○ Dim Vlonb As Double
○ Dim i As Long, n1 As Long, n2 As Long, Le#
○ Dim x1 As Single, y1 As Single, z1 As Single
○ Dim x2 As Single, y2 As Single, z2 As Single
○ Dim xp As Double, yp As Double, zp As Double
○ If Slong = "" Then Exit Sub
○ Vlonb = ValP(Slong)
○ If Vlonb > 0 Then
○   If Selecci.value = 1 Then 'proceso de selección
○     For i = 1 To ne
○       If C2dba(i, 0) = 1 Then 'barra seleccionada
○         Slon#(i) = (Vlonb)

```

```

○
○
○   n1 = brr(i, 1): n2 = brr(i, 2)
○   x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
○   x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
○   xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
○   Le# = Sqr(xp * xp + yp * yp + zp * zp)
○   x2 = xp / Le# * Slon#(i): y2 = yp / Le# * Slon#(i): z2 = zp / Le# * Slon#(i)
○   CorL(i, 1) = x1 + x2: CorL(i, 2) = y1 + y2: CorL(i, 3) = z1 + z2
○   End If
○ Next i
○ Else
○   i = bra
○   Slon#(i) = (Vlonb)
○   n1 = brr(i, 1): n2 = brr(i, 2)
○   x1 = Cor(n1, 1): y1 = Cor(n1, 2): z1 = Cor(n1, 3)
○   x2 = Cor(n2, 1): y2 = Cor(n2, 2): z2 = Cor(n2, 3)
○   xp = x2 - x1: yp = y2 - y1: zp = z2 - z1
○   Le# = Sqr(xp * xp + yp * yp + zp * zp)
○   x2 = xp / Le# * Slon#(i): y2 = yp / Le# * Slon#(i): z2 = zp / Le# * Slon#(i)
○   CorL(i, 1) = x1 + x2: CorL(i, 2) = y1 + y2: CorL(i, 3) = z1 + z2
○ End If
○ End If
○ End Sub
○
○
○ Private Sub Textnom_LostFocus()
○ tipb.List(indicetipo) = Textnom.Text
○ p$(indicetipo + 1) = Textnom.Text
○ tipb.Height = tipbH
○ End Sub
○
○
○ Private Sub tipb_Click()
○ Dim tipo As Integer
○ tipo = tipb.ListIndex + 1
○ TipoDat(1).Text = CStr(Tipos(tipo, 1))
○ TipoDat(2).Text = CStr(Tipos(tipo, 2))
○ TipoDat(3).Text = CStr(Tipos(tipo, 3))
○ TipoDat(4).Text = CStr(Tipos(tipo, 4))
○ TipoDat(5).Text = CStr(Tipos(tipo, 5))
○ End Sub
○
○
○ Private Sub tipb_DblClick()
○ tipb.Height = 350
○ Textnom.SetFocus
○ Textnom.Text = tipb.List(tipb.ListIndex)
○ indicetipo = tipb.ListIndex
○ End Sub
○
○
○ Private Sub tipb_LostFocus()
○ Dim i As Long
○ If Selecci.value = 1 Then      'proceso de selección
○   For i = 1 To ne
○     If C2dba(i, 0) = 1 Then    'barra seleccionada
○       brr(i, 3) = tipb.ListIndex + 1
○     End If
○   Next i
○ Else
○   brr(bra, 3) = tipb.ListIndex + 1
○ End If
○ End Sub
○
○
○ Private Sub TipoDat_LostFocus(index As Integer)
○ Dim tipo As Integer
○ tipo = tipb.ListIndex + 1
○ Tipos(tipo, index) = ValP(TipoDat(index).Text)
○ End Sub
○
○
○
○

```

```

○ Private Sub ZoomP_Click()
○ zoomnum -1, 0, 0, 0
○ End Sub
○
○
○
○ Private Sub ZoomV1_Click()
○ ZoomV1.Visible = False
○ ZoomV1.Enabled = False
○ ZoomV2.Visible = True
○ ZoomV2.Enabled = True
○ End Sub
○
○
○
○ Private Sub ZoomV2_MouseDown(Button As Integer, Shift As Integer, x As Single, y As
○ Single)
○ ZoomV2.Visible = False
○ ZoomV2.Enabled = False
○ ZoomV1.Visible = True
○ ZoomV1.Enabled = True
○ End Sub
○
○
○
○ Private Sub locpunto(x As Single, y As Single, pnto As Long)
○ 'busca punto, y si lo encuentra devuelve su número y sus coordenadas 2d
○ 'si no lo encuentra devuelve pnto=0)
○ Dim aa As Single, bb As Single, encontrado As Boolean
○ encontrado = False
○ Do While (pnto < np) And (encontrado = 0)
○ pnto = pnto + 1
○ aa = Abs(x - C2dnu(pnto, 1)): bb = Abs(y - C2dnu(pnto, 2))
○ If aa < distancia And bb < distancia Then
○ encontrado = True
○ End If
○ Loop
○ If encontrado = 0 Then
○ pnto = 0
○ Else
○ x = C2dnu(pnto, 1)
○ y = C2dnu(pnto, 2)
○ End If
○ End Sub
○
○
○
○ Private Sub OtraBarra(pt1 As Long, pt2 As Long)
○ Dim barr As Long, Existe As Boolean, i As Long, j As Long
○ Dim x As Single, y As Single, Z As Single, a As Single, b As Single
○ Dim x1 As Single, y1 As Single, z1 As Single
○ Dim x2 As Single, y2 As Single, z2 As Single
○ Dim Clbarr As Long
○ 'borra última línea.....
○ Dib.DrawMode = 7 'Xor
○ Dib.Line (dibX, dibY)-(Dib.CurrentX, Dib.CurrentY), SeColor
○ Dib.DrawMode = 13
○ If pt2 > 0 And pt2 <> pt1 Then 'existe el punto final....
○ x = Cor(pt2, 1): y = Cor(pt2, 2): Z = Cor(pt2, 3)
○ axo3d2d x, y, Z, a, b 'coord. 2d para dibujo barra...
○ If pt2 < pt1 Then Swap pt2, pt1 'ordena de menor a mayor.
○ barr = 0: Existe = False
○ Do
○ barr = barr + 1
○
○ Loop While Existe = 0 And barr < ne
○ If Existe = True Then 'se elimina la barra ...
○ For i = barr To ne - 1
○ For j = 1 To 3
○ brr(i, j) = brr(i + 1, j)
○ Next j
○ C2dba(i, 1) = C2dba(i + 1, 1)
○ C2dba(i, 2) = C2dba(i + 1, 2)
○ lon0#(i) = lon0#(i + 1)
○ Next i
○ ne = ne - 1
○ barr = 0
○ Clbarr = Dib.BackColor 'para dibujo
○ Else ' se añade barra de tipo 1.....

```

```

O     ne = ne + 1
O     barr = ne
O     brr(barr, 1) = pt1
O     brr(barr, 2) = pt2
O     brr(barr, 3) = 1
O     Clbarr = colbarra(1) 'para dibujo
O     End If
O     Dib.Line (dibX, dibY)-(a, b), Clbarr
O End If
O If pt2 = 0 Then 'si no se soltó en un pto determinado...
O     np = np + 1
O     pt2 = np
O     x = Cor(pt1, 1) - 1
O     y = Cor(pt1, 2) + 1
O     z = Cor(pt1, 3) + 1
O     '.....
O     Cor(pt2, 1) = x
O     Cor(pt2, 2) = y
O     Cor(pt2, 3) = z
O     Coac(pt2, 1) = 1234
O     Coac(pt2, 2) = 1234
O     Coac(pt2, 3) = 1234
O     CoaL(pt2, 1) = -2000
O     CoaL(pt2, 2) = -2000
O     CoaL(pt2, 3) = -2000
O     CoaL(pt2, 4) = 2000
O     CoaL(pt2, 5) = 2000
O     CoaL(pt2, 6) = 2000
O     '.....
O     axo3d2d x, y, z, a, b
O     Clbarr = colbarra(1) 'para dibujo
O     C2dnu(pt2, 1) = a
O     C2dnu(pt2, 2) = b
O     ne = ne + 1
O     barr = ne
O     brr(barr, 1) = pt1
O     brr(barr, 2) = pt2
O     brr(barr, 3) = 1
O     Dib.Line (dibX, dibY)-(a, b), Clbarr
O     If FramNud.Visible Then
O         Dib.Line (a - taman, b - taman)-(a + taman, b + taman), QBColor(11), BF
O     End If
O End If
O If barr <> 0 Then 'si se añadió una barra...
O     'barra en 2d.....
O     x1 = Cor(pt1, 1): y1 = Cor(pt1, 2): z1 = Cor(pt1, 3)
O     x2 = Cor(pt2, 1): y2 = Cor(pt2, 2): z2 = Cor(pt2, 3)
O     x = ((x1 + x2) / 2)
O     y = ((y1 + y2) / 2)
O     z = ((z1 + z2) / 2)
O     axo3d2d x, y, z, a, b
O     C2dba(barr, 1) = a
O     C2dba(barr, 2) = b
O     If FramBar.Visible Then
O         Dib.Line (a - taman, b - taman)-(a + taman, b + taman), QBColor(10), BF
O     End If
O     lon0#(barr) = Sqr((x2 - x1) ^ 2 + (y2 - y1) ^ 2 + (z2 - z1) ^ 2)
O     Slon#(barr) = lon0#(barr) * 0.5
O     CorL(barr, 1) = (x2 - x1) / 2
O     CorL(barr, 2) = (y2 - y1) / 2
O     CorL(barr, 3) = (z2 - z1) / 2
O End If
O End Sub
O
O
O
O Private Sub dibuEnlace(Pct As Object, Esc As Single)
O Dim i As Integer, n1 As Integer, n2 As Integer
O Dim x1 As Single, y1 As Single
O Dim x2 As Single, y2 As Single
O Dim px, py, DiColor
O DiColor = QBColor(14)
O px = Pct.ScaleWidth: py = Pct.ScaleHeight
O For i = 1 To nea
O     n1 = brsA(i, 1): n2 = brsA(i, 2)
O     x1 = C2dba(n1, 1) * Esc: y1 = C2dba(n1, 2) * Esc
O     x2 = C2dba(n2, 1) * Esc: y2 = C2dba(n2, 2) * Esc
O     If Not ((x1 < 0 And x2 < 0) Or (x1 > px And x2 > px)) Then
O         If Not ((y1 < 0 And y2 < 0) Or (y1 > py And y2 > py)) Then

```

```

O         Pct.Circle (x1, y1), taman * 2, DiColor 'punto enlazado 1
O         Pct.Circle (x2, y2), taman * 2, DiColor 'punto enlazado 2
O         Pct.Line (x1, y1)-(x2, y2), DiColor 'Dibuja puntos centrales de las barras
O     End If
O End If
O Next i
O End Sub
O
O
O
O Private Sub prn_dibujaestr(Fe As Single)
O Dim i, n1, n2, x1 As Single, x2 As Single, y1 As Single, y2 As Single, DiColor
O
O Dim px As Single, py As Single, pxd As Single, pyd As Single, Esc As Single
O pxd = Dib.ScaleWidth: pyd = Dib.ScaleHeight
O px = Printer.ScaleWidth: py = Printer.ScaleHeight
O If (pxd / px) < (pyd / py) Then Esc = py / pyd Else Esc = px / pxd 'escala
O Esc = Esc * Fe
O For i = 1 To ne
O     n1 = brr(i, 1): n2 = brr(i, 2)
O     x1 = C2dnu(n1, 1) * Esc
O     x2 = C2dnu(n2, 1) * Esc
O     y1 = C2dnu(n1, 2) * Esc
O     y2 = C2dnu(n2, 2) * Esc
O     If (Selecci.value = 1) And (C2dba(i, 0) = 1) Then
O         DiColor = SeColorb
O     Else
O         DiColor = colbarra(brr(i, 3))
O     End If
O     'dibuja línea .....
O     If Not ((x1 < 0 And x2 < 0) Or (x1 > px And x2 > px)) Then
O         If Not ((y1 < 0 And y2 < 0) Or (y1 > py And y2 > py)) Then
O             Printer.Line (x1, y1)-(x2, y2), DiColor
O         End If
O     End If
O Next i
O prn_dibujanu Esc
O prn_dibuenlace Esc
O End Sub
O
O
O Private Sub prn_dibujanu(Esc As Single)
O ' Dibuja puntos
O Dim x1, y1, i As Long, j As Long
O Dim px, py, Sc As Single, DiColor As Single
O 'para dibujo cargas y coacciones...
O Dim a As Single, b As Single, C As Single, Cx As Single, j2 As Single
O Dim a2 As Single, b2 As Single, a3 As Single, b3 As Single, a4 As Single, b4 As
Single, a5 As Single, b5 As Single
O Dim CaColor As Single, CoColor As Single
O Dim Esc2 As Single
O DiColor = QBColor(11)
O px = Printer.ScaleWidth: py = Printer.ScaleHeight
O Esc2 = Esc * 0.6
O 'dibuja puntos..
O For i = 1 To np
O     x1 = C2dnu(i, 1) * Esc: y1 = C2dnu(i, 2) * Esc
O     If Not ((x1 < 0) Or (x1 > px)) Then
O         If Not ((y1 < 0) Or (y1 > py)) Then
O             Printer.Line (x1 - taman * Esc2, y1 - taman * Esc2)-(x1 + taman * Esc2, y1 +
taman * Esc2), DiColor, BF
O         End If
O     End If
O Next i
O 'dibuja cargas...
O If HScal.value > 0 Then 'si la escala de dibujo de coacciones y cargas es >0
O     CaColor = QBColor(1)
O     CoColor = QBColor(12)
O     Sc = HScal.value / 50
O     For i = 1 To np
O         x1 = C2dnu(i, 1) * Esc: y1 = C2dnu(i, 2) * Esc
O         If Not ((x1 < 0) Or (x1 > px)) Then
O             If Not ((y1 < 0) Or (y1 > py)) Then
O                 For j = 1 To 3
O                     j2 = (j Mod 3) + 1
O                     C = Carg(i, j)
O                     Cx = Coac(i, j)
O                     If C <> 0 Or Cx <> 1234 Then

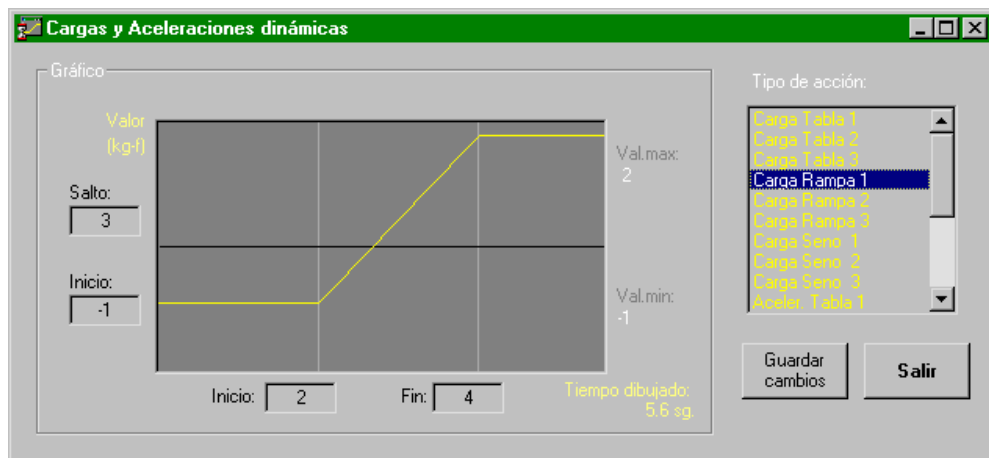
```

```

O      a2 = uni(j, 1) * Sc * Esc
O      b2 = uni(j, 2) * Sc * Esc
O      a3 = uni(j, 1) * Sc * 1.3 * Esc 'punta flecha
O      b3 = uni(j, 2) * Sc * 1.3 * Esc
O      a4 = uni(j2, 1) * Sc * 0.3 * Esc
O      b4 = uni(j2, 2) * Sc * 0.3 * Esc
O      a5 = -uni(j2, 1) * Sc * 0.3 * Esc
O      b5 = -uni(j2, 2) * Sc * 0.3 * Esc
O      If C <> 0 Then 'dibujo carga...
O          C = Sgn(C) * Log(1 + Abs(C)) 'ampliación carga...
O          Printer.Line (x1, y1)-(x1 + a2 * C, y1 + b2 * C), CaColor
O          Printer.Line -(x1 + (a2 + a4) * C, y1 + (b2 + b4) * C), CaColor
O          Printer.Line -(x1 + a3 * C, y1 + b3 * C), CaColor
O          Printer.Line -(x1 + (a2 + a5) * C, y1 + (b2 + b5) * C), CaColor
O          Printer.Line -(x1 + a2 * C, y1 + b2 * C), CaColor
O      End If
O      C = 0.5 'ampliación coacción
O      If Cx <> 1234 Then 'dibujo coacción...
O          Printer.Line (x1, y1)-(x1 + a2 * C, y1 + b2 * C), CoColor
O          Printer.PSet (x1 + (a2 + a4) * C, y1 + (b2 + b4) * C), CoColor
O          'Dib.Line -(x1 + a3 * c, y1 + b3 * c), CoColor
O          Printer.Line -(x1 + (a2 + a5) * C, y1 + (b2 + b5) * C), CoColor
O          'Dib.Line -(x1 + a2 * c, y1 + b2 * c), CoColor
O      End If
O      End If
O      Next j
O      End If
O      End If
O      Next i
O      End If
O      End Sub
O
O
O
O      Private Sub prn_dibuenlace(Esc As Single)
O      Dim i As Integer, n1 As Integer, n2 As Integer
O      Dim x1 As Single, y1 As Single
O      Dim x2 As Single, y2 As Single
O      Dim px, py, DiColor, Esc2 As Single
O      DiColor = QBColor(14)
O      px = Printer.ScaleWidth: py = Printer.ScaleHeight
O      Esc2 = Esc * 0.6
O      For i = 1 To nea
O          n1 = brrsA(i, 1): n2 = brrsA(i, 2)
O          x1 = C2dba(n1, 1) * Esc: y1 = C2dba(n1, 2) * Esc
O          x2 = C2dba(n2, 1) * Esc: y2 = C2dba(n2, 2) * Esc
O          If Not ((x1 < 0 And x2 < 0) Or (x1 > px And x2 > px)) Then
O              If Not ((y1 < 0 And y2 < 0) Or (y1 > py And y2 > py)) Then
O                  Printer.Circle (x1, y1), taman * 2 * Esc2, DiColor 'punto enlazado 1
O                  Printer.Circle (x2, y2), taman * 2 * Esc2, DiColor 'punto enlazado 2
O                  Printer.Line (x1, y1)-(x2, y2), DiColor 'Dibuja puntos centrales de las
O                      barras
O              End If
O          End If
O      Next i
O      End Sub
O
O

```

## 7.3.9 MÓDULO DE EDICIÓN DE LAS CARGAS DINÁMICAS.



```

O 'cargas.....
O Dim f0(-3 To 6), f1(-3 To 6), t0(-3 To 6), t1(-3 To 6)
O Dim npqd(3) 'n° de puntos de las tablas 1,2,3
O Dim tipoq$(3) 'títulos de las tablas
O Dim tabqd1(), tabqd2(), tabqd3() 'tablas cargas 1,2,3
O 'aceleraciones.....
O Dim af0(-3 To 6), af1(-3 To 6), at0(-3 To 6), at1(-3 To 6)
O Dim npad(3) 'n° de puntos de las tablas 1,2,3
O Dim tipoa$(3) 'títulos de las tablas
O Dim tabad1(), tabad2(), tabad3() 'tablas aceleración 1,2,3
O Dim Trabajando As Boolean
O
O
O
O Sub LeeDat()
O Dim ab$
O 'Lectura de las cargas y aceleraciones DINAMICAS de los puntos.....
O 'Para la carga i (-3 a 6):
O 'Prqd(i) - fichero con las propiedades Generales de las cargas dinámicas
O 'npqd(i) - n° de puntos de la tabla de la carga dinámica
O 'tipoq$(i) - tipo de tabla de carga dinámica
O 'tabqdi() - tablas de cargas dinámicas
O 'Datos de las CARGAS DINÁMICAS en f0 y f1 (Kg ) y t0 y t1 (seg).....
O ab$ = nom$ + "\" + "Prqd.Dat"
O Open ab$ For Input As #1
O For i = -3 To 6
O Input #1, f0(i), f1(i), t0(i), t1(i) ' fuerza en Kgf, tiempo en seg
O Next i
O Close #1
O 'Tabla 1
O ab$ = Trim(nom$) + "\" + "Cargd1.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(1), tipoq$(1)
O ReDim tabqd1(npqd(1))
O For i = 1 To npqd(1)
O Input #1, tabqd1(i)
O Next i
O Close #1
O 'Tabla 2
O ab$ = Trim(nom$) + "\" + "Cargd2.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(2), tipoq$(2)
O ReDim tabqd2(npqd(2))
O For i = 1 To npqd(2)
O Input #1, tabqd2(i)
O Next i
O Close #1
O 'Tabla 3
O ab$ = Trim(nom$) + "\" + "Cargd3.Dat"
O Open ab$ For Input As #1
O Input #1, npqd(3), tipoq$(3)
O ReDim tabqd3(npqd(3))
O For i = 1 To npqd(3)
O Input #1, tabqd3(i)
O Next i

```





```

O      If (tiempo < t0(1)) Then b = f0(1)
O      If (tiempo <= t1(1) And tiempo >= t0(1)) Then b = f0(1) + f1(1) * (tiempo -
      t0(1)) / (t1(1) - t0(1))
O      If (tiempo > t1(1)) Then b = f1(1) + f0(1)
O      Case (2)          ' Función rampa 2
O      If (tiempo < t0(2)) Then b = f0(2)
O      If (tiempo <= t1(2) And tiempo >= t0(2)) Then b = f0(2) + f1(2) * (tiempo -
      t0(2)) / (t1(2) - t0(2))
O      If (tiempo > t1(2)) Then b = f1(2) + f0(2)
O      Case (3)          ' Función rampa 3
O      If (tiempo < t0(3)) Then b = f0(3)
O      If (tiempo <= t1(3) And tiempo >= t0(3)) Then b = f0(3) + f1(3) * (tiempo -
      t0(3)) / (t1(3) - t0(3))
O      If (tiempo > t1(3)) Then b = f1(3) + f0(3)
O      Case (4)          ' Función seno 1
O      If (tiempo < t0(4)) Then b = f0(4)
O      If (tiempo >= t0(4)) Then b = f0(4) + f1(4) * Sin(2# * pi * (tiempo - t0(4)) /
      t1(4))
O      Case (5)          ' Función seno 2
O      If (tiempo < t0(5)) Then b = f0(5)
O      If (tiempo >= t0(5)) Then b = f0(5) + f1(5) * Sin(2# * pi * (tiempo - t0(5)) /
      t1(5))
O      Case (6)          ' Función seno 1
O      If (tiempo < t0(6)) Then b = f0(6)
O      If (tiempo >= t0(6)) Then b = f0(6) + f1(6) * Sin(2# * pi * (tiempo - t0(6)) /
      t1(6))
O      Case Default
O      b = 0#
O      End Select
O      qDin = b
O      End Function
O
O      '.....
O      '.....
O      Function aDin(tiempo, ntipo)
O      'Cargas dinámicas en Kgf
O      Dim NN, delt, ti, b
O
O      Select Case (ntipo)
O      Case (0)
O      b = 0#
O      Case (1)          ' Función rampa 1
O      If (tiempo < at0(1)) Then b = af0(1)
O      If (tiempo <= at1(1) And tiempo >= at0(1)) Then
O      b = af0(1) + af1(1) * (tiempo - at0(1)) / (at1(1) - at0(1))
O      End If
O      If (tiempo > at1(1)) Then b = af1(1) + af0(1)
O      Case (2)          ' Función rampa 2
O      If (tiempo < at0(2)) Then b = af0(2)
O      If (tiempo <= at1(2) And tiempo >= at0(2)) Then
O      b = af0(2) + af1(2) * (tiempo - at0(2)) / (at1(2) - at0(2))
O      End If
O      If (tiempo > at1(2)) Then b = af1(2) + af0(2)
O      Case (3)          ' Función rampa 3
O      If (tiempo < at0(3)) Then b = af0(3)
O      If (tiempo <= at1(3) And tiempo >= at0(3)) Then
O      b = af0(3) + af1(3) * (tiempo - at0(3)) / (at1(3) - at0(3))
O      End If
O      If (tiempo > at1(3)) Then b = af1(3) + af0(3)
O      Case (4)          ' Función seno 1
O      If (tiempo < at0(4)) Then b = af0(4)
O      If (tiempo >= at0(4)) Then
O      b = af0(4) + af1(4) * Sin(2# * pi * (tiempo - at0(4)) / at1(4))
O      End If
O      Case (5)          ' Función seno 2
O      If (tiempo < at0(5)) Then b = af0(5)
O      If (tiempo >= at0(5)) Then
O      b = af0(5) + af1(5) * Sin(2# * pi * (tiempo - at0(5)) / at1(5))
O      End If
O      Case (6)          ' Función seno 3
O      If (tiempo < at0(6)) Then b = af0(6)
O      If (tiempo >= at0(6)) Then
O      b = af0(6) + af1(6) * Sin(2# * pi * (tiempo - at0(6)) / at1(6))
O      End If
O      Case (-3)          ' Función tabla 3 Interpolación lineal
O      If (tiempo < at0(-3)) Then b = af0(-3)
O      If (tiempo >= at0(-3) And tiempo <= at1(-3)) Then
O      delt = (at1(-3) - at0(-3)) / (npad(1) - 1)
O      NN = Int(((tiempo - at0(-3)) / delt))

```

```

O      ti = tiempo - at0(-3) - NN * delt
O      b = af0(-3) + af1(-3) * (tabad1(NN + 1) + ti / delt * (tabad1(NN + 2) -
          tabad1(NN + 1)))
O      End If
O      If (tiempo > at1(-3)) Then b = af0(-3) + af1(-3) * tabad1(npad(1))
O      Case (-2)          ' Función tabla 2 Interpolación lineal
O      If (tiempo < at0(-2)) Then b = af0(-2)
O      If (tiempo >= at0(-2) And tiempo <= at1(-2)) Then
O      delt = (at1(-2) - at0(-2)) / (npad(2) - 1)
O      NN = Int(((tiempo - at0(-2)) / delt))
O      ti = tiempo - at0(-2) - NN * delt
O      b = af0(-2) + af1(-2) * (tabad2(NN + 1) + ti / delt * (tabad2(NN + 2) -
          tabad2(NN + 1)))
O      End If
O      If (tiempo > at1(-2)) Then b = af0(-2) + af1(-2) * tabad2(npad(2))
O      Case (-1)          ' Función tabla 1 Interpolación lineal
O      If (tiempo < at0(-1)) Then b = af0(-1)
O      If (tiempo >= at0(-1) And tiempo <= at1(-1)) Then
O      delt = (at1(-1) - at0(-1)) / (npad(3) - 1)
O      NN = Int(((tiempo - at0(-1)) / delt))
O      ti = tiempo - at0(-1) - NN * delt
O      b = af0(-1) + af1(-1) * (tabad3(NN + 1) + ti / delt * (tabad3(NN + 2) -
          tabad3(NN + 1)))
O      End If
O      If (tiempo > at1(-1)) Then b = af0(-1) + af1(-1) * tabad3(npad(3))
O      Case Default
O      b = 0#
O      End Select
O      aDin = b
O      End Function
O
O
O      Private Sub CmdGuarda_Click()
O      Dim ab$
O      ab$ = nom$ + "\" + "Prqd.Dat"
O      Open ab$ For Output As #1
O      For i = -3 To 6 ' fuerza en Kgf, tiempo en seg
O      Print #1, f0(i)
O      Print #1, f1(i)
O      Print #1, t0(i)
O      Print #1, t1(i)
O      Next i
O      Close #1
O      'Datos de las Aceleraciones en m/s*s f0 y f1 y t0 y t1 seg.....
O      ab$ = Trim(nom$) + "\" + "Arqd.Dat"
O      Open ab$ For Output As #1
O      For i = -3 To 6 ' fuerza en Kgf, Kgf, tiempo en seg, seg
O      Print #1, af0(i)
O      Print #1, af1(i)
O      Print #1, at0(i)
O      Print #1, at1(i)
O      Next i
O      Close #1
O      End Sub
O
O
O      Private Sub CmdSalir_Click()
O      Unload Me
O      End Sub
O
O
O
O      Private Sub Form_Load()
O      Trabajando = False
O      LbUnid.Caption = ""
O      LbQmax.Caption = ""
O      LbQmin.Caption = ""
O      LbTtot.Caption = ""
O      LbTitTab.Caption = ""
O      LstDin.Clear
O      LstDin.AddItem "Carga Tabla 1" '-1
O      LstDin.AddItem "Carga Tabla 2"
O      LstDin.AddItem "Carga Tabla 3"
O      LstDin.AddItem "Carga Rampa 1"
O      LstDin.AddItem "Carga Rampa 2"
O      LstDin.AddItem "Carga Rampa 3"
O      LstDin.AddItem "Carga Seno 1"
O      LstDin.AddItem "Carga Seno 2"

```

```

○     LstDin.AddItem "Carga Seno 3"
○     LstDin.AddItem "Aceler. Tabla 1"
○     LstDin.AddItem "Aceler. Tabla 2"
○     LstDin.AddItem "Aceler. Tabla 3"
○     LstDin.AddItem "Aceler. Rampa 1"
○     LstDin.AddItem "Aceler. Rampa 2"
○     LstDin.AddItem "Aceler. Rampa 3"
○     LstDin.AddItem "Aceler. Seno 1"
○     LstDin.AddItem "Aceler. Seno 2"
○     LstDin.AddItem "Aceler. Seno 3"
○ LeeDat
○ End Sub
○
○
○ Private Sub Form_Unload(Cancel As Integer)
○ Menu.Show
○ End Sub
○
○
○ Private Sub LstDin_Click()
○ Dim n As Integer, a$, tm0 As Single, tm1 As Single, tmt As Single, qr0 As Single, qr1
○     As Single, qrmax As Single, qrmin As Single
○
○ Trabajando = True
○
○ n = LstDin.ListIndex + 1
○ If n < 10 Then 'carga
○     a$ = "carga"
○     LbUnid.Caption = "(kg-f)"
○ Else 'aceleración
○     n = n - 9
○     a$ = "acele"
○     LbUnid.Caption = "(m/s2)"
○ End If
○ n = n - 3 'valores de -2 a 6
○ If n < 1 Then
○     n = n - 1 'valores de -3a-1 la6
○     LbTitTab.Caption = tipoq$(n + 4)
○ Else
○     LbTitTab.Caption = ""
○ End If
○ If n < 4 Then
○     LbQFin.Caption = "Salto:"
○     LbTfin.Caption = "Fin:"
○ Else
○     LbQFin.Caption = "Amplitud:"
○     LbTfin.Caption = "Período:"
○ End If
○ dibuj Picture1, a$, n, tm0, tm1, tmt, qr0, qr1, qrmax, qrmin
○ TxQ0 = qr0: TxQ1 = qr1
○ TxT0 = tm0: TxT1 = tm1
○ LbTtot = Str$(tmt) + " sg."
○ LbQmax = Str$(qrmax)
○ LbQmin = Str$(qrmin)
○ Trabajando = False
○ End Sub
○
○
○ Sub dibuj(Pic As PictureBox, a$, n As Integer, tm0 As Single, tm1 As Single, tmt As
○     Single, qr0 As Single, qr1 As Single, qrmax As Single, qrmin As
○     Single)
○ 'IN- a$= carga o acele,n= n° de tabla
○ 'OUT-tm0, tm1, tmt, qr0, qr1, qrmax, qrmin
○
○ Dim x1, x2, y1, y2, xt, yt
○ 'Dim qr0, qr1, qrt, tm0, tm1, tmt
○ Dim pt() As Single
○ Dim Tot As Integer, i As Integer, paso As Single, colr As Single, Max As Single
○ Dim Scalx As Single
○ Tot = 150 'n° de tramos a dibujar
○ ReDim pt(Tot) 'coordenadas Y de los puntos.
○ qrmin = 10000: qrmax = -10000
○ 'cargamos pt
○ Select Case a$
○     Case "carga"
○         qr0 = f0(n): qr1 = f1(n): tm0 = t0(n): tm1 = t1(n)
○     Case "acele"
○         qr0 = af0(n): qr1 = af1(n): tm0 = at0(n): tm1 = at1(n)

```

```

O Case Else: Stop
O End Select
O Pic.Cls
O If tm1 <= 0 Then Exit Sub
O colr = RGB(220, 220, 220)
O If n < 4 Then
O tmt = tm1 * 1.4
O Scalx = Pic.ScaleWidth / (tmt)
O Pic.Line (tm0 * Scalx, 0)-(tm0 * Scalx, Pic.ScaleHeight), colr
O Pic.Line (tm1 * Scalx, 0)-(tm1 * Scalx, Pic.ScaleHeight), colr
O Else
O tmt = (tm0 + tm1) * 1.4
O Scalx = Pic.ScaleWidth / (tmt)
O Pic.Line (tm0 * Scalx, 0)-(tm0 * Scalx, Pic.ScaleHeight), colr
O Pic.Line ((tm0 + tm1) * Scalx, 0)-((tm0 + tm1) * Scalx, Pic.ScaleHeight), colr
O End If
O paso = tmt / Tot
O For i = 0 To Tot
O Select Case a$
O Case "carga"
O pt(i) = qDin(i * paso, n)
O Case "acele"
O pt(i) = aDin(i * paso, n)
O End Select
O If pt(i) > qrmax Then qrmax = pt(i)
O If pt(i) < qrmin Then qrmin = pt(i)
O Next i
O colr = Pic.ForeColor
O dibmatr pt(), Pic, 0.9, colr, 1, Max
O End Sub
O
O
O
O Sub dibmatr(mat() As Single, Pict As Object, Scl As Single, Colo As Single, Eje As
Single, Max As Single)
O '-----
O ' Dibuja las matriz mat(), en el objeto Pict, con el color Colo
O ' devuelve el valor del máximo absoluto en Max
O ' el dibujo queda centrado con el eje X a la mitad de la altura.
O ' Scl - factor escala Y. El valor 1 ajusta verticalmente al máximo.
O ' Eje - si es >0 dibuja el eje X con el color Eje
O '-----
O Dim Iter As Long, Scalx As Single, Scaly As Single
O Dim Ox As Single, Oy As Single, Dx As Single, Dy As Single
O Dim i As Long
O Dim aa As Single, bb As Single
O Iter = UBound(mat)
O Ox = Pict.ScaleLeft: Oy = Pict.ScaleTop
O Dx = Pict.ScaleWidth - Ox: Dy = Pict.ScaleHeight - Oy
O Oy = Oy + Dy / 2 'eje centrado
O aa = 1
O bb = 0
O 'calculamos los valores máximos para escalarlos.....
O Max = mat(1)
O For i = 1 To Iter
O If Abs(mat(i)) > Abs(Max) Then Max = mat(i)
O Next i
O If Max = 0 Then Max = 1 'para que no casque
O Scalx = Dx / (Iter - 1) * aa 'escalas de representación
O Scaly = Scl * Dy / (2 * Abs(Max))
O If bb <> 0 Then Scaly = Dy / (2 * bb)
O 'dibujo.....
O If Eje >= 0 Then Pict.Line (Ox + Dx, Oy)-(Ox, Oy), Eje
O Pict.PSet (Ox, Oy - mat(1) * Scaly), Colo
O For i = 1 To Iter
O Pict.Line -(Ox + i * Scalx, Oy - mat(i) * Scaly), Colo
O Next i
O End Sub
O
O
O
O Private Sub TxQ0_Change()
O CambiaDatos
O End Sub
O
O
O
O Private Sub TxQ1_Change()
O CambiaDatos

```

```

O End Sub
O
O
O
O Private Sub TxT0_Change()
O CambiaDatos
O End Sub
O
O
O
O Private Sub TxT1_Change()
O CambiaDatos
O End Sub
O
O
O
O Sub CambiaDatos()
O Dim n As Integer, i As Integer
O If Trabajando = False Then
O n = LstDin.ListIndex + 1
O If n > 0 Then
O If n < 10 Then 'carga
O i = n - 3 'valores de -2 a 6
O If i < 1 Then i = i - 1 'valores de -3a-1 la6
O f0(i) = ValP(TxQ0.Text): f1(i) = ValP(TxQ1.Text)
O t0(i) = ValP(TxT0.Text): t1(i) = ValP(TxT1.Text)
O Else 'aceleración
O i = n - 9
O i = i - 3 'valores de -2 a 6
O If i < 1 Then i = i - 1 'valores de -3a-1 la6
O af0(i) = ValP(TxQ0.Text): af1(i) = ValP(TxQ1.Text)
O at0(i) = ValP(TxT0.Text): at1(i) = ValP(TxT1.Text)
O End If
O End If
O LstDin_Click
O End If
O End Sub
O
O
O
O

```

## 7.3.10 MÓDULO DE CÁLCULO DINÁMICO EN FORTRAN

```

O
O Module Nucleo
O
O ! Fecha de inicio de la versión 3.1 con nudo central desplazado 6-11-97
O ! Fecha finalización versión 6-1-98
O ! Fecha de inicio de la versión 3.2 con límite de desplazamiento en coacciones 2-4-99
O ! Fecha finalización versión 6-4-99
O ! Programa con Runge-Kutta en las cuatro primeras iteraciones y Adams-Moulton en las
O siguientes
O ! Considera el axil de las barras paralelo a la barra sin flexionar.
O ! :::::::::::::::::::::::::::::::::::::: DINAMI.F90 ::::::::::::::::::::::::::::::::::::::
O ! Versión que intercala o no (nnd$), un nudo intermedio durante el cálculo,
O ! Los nudos añadidos (intermedios) se comportan como barras
O ! Los nudos extremos se comportan como puntos materiales
O ! .....
O
O IMPLICIT none
O Real(KIND=8), allocatable :: ca(:,:),va(:,:),anx(:,:),bnx(:,:)
O Real(KIND=8), allocatable :: p(:,:), alon(:,:), slon1(:), amas(:,), qst(:,:), ainer(:,),
O c(:,:), v(:,:),tabqd1(:), tabqd2(:), tabqd3(:),tabad1(:), tabad2(:),
O tabad3(:)
O Real(KIND=8) :: f0(-3:6), f1(-3:6), t0(-3:6), t1(-3:6)
O Real(KIND=8) :: af0(-3:6), af1(-3:6), at0(-3:6), at1(-3:6)
O Real(KIND=8) :: dt, alambda, velmax,vcritica,g
O Real(KIND=8) :: dpi,kc,porcen
O Real(KIND=8),allocatable :: tabym(:),c0(:,:)
O Real(KIND=8) :: tt(3,3),tti(3,3)
O Integer :: np2, ne2, np, ne, ndim,ntp, npqd(3),itergrb,npad(3),npym
O Integer , allocatable :: nq(:,:), nll(:,:), nx(:,:),na(:,:)
O Integer*4 :: itermax, nporcen
O Character(LEN=15) , allocatable :: pp$(:)
O Character*2 :: ppropio$,nnd$,axil$, cont$,xpos$,xvel$,xaxi$,xcor$,xfle$
O Character*3 :: extension2$

```

```

O Character*10 :: amortt$
O Character(LEN=40) :: tipotit$, comentario$
O Character(LEN=3) :: extension$
O Character(LEN=50) :: nom$
O
O ! alon - longitud en reposo, mínima y máxima
O ! qst - carga estática en nudo i con dirección x,y,x en Kgf
O ! dtqd - intervalo de tiempo de la tabla de la carga dinámica
O ! nc - n° de puntos de la tabla de la carga dinámica
O !tipoq$ - tipo de carga dinámica
O ! nq - tipo de fuerza dinámica
O ! amas - masa de los nudos
O ! nll - nudos que delimitan las barras y tipo de barra
O ! nx - coacciones
O ! anx - coacciones inferiores
O ! bnx - coacciones superiores
O ! kc - constante de elasticidad de muelle de las coacciones
O
O Contains !comienzo de las subrutinas.....
O ! .....
O Subroutine Principal(dlg)
O use dflogm !permite acceso a dflogm ???
O type (dialog) :: dlg
O !!include "resource.fd"
O Call Lectura_Datos
O Call Nucleol(dlg)
O deallocate (ca,va,anx,bnx,p,alon,slon1,amas,qst,ainer,c,v,tabqd1,tabqd2,tabqd3,
O tabad1,tabad2,tabad3)
O deallocate (tabym,c0)
O deallocate (nq,nll,nx,na)
O deallocate (pp$)
O End Subroutine Principal
O
O Subroutine Lectura_Datos
O Real(KIND=8) :: alfa,beta,coalfa,sialfa,cobeta,sibeta, vu, acos1
O Real(KIND=8) :: x1, x2, y1, y2, z1, z2, al, ale, amasabar, xp, yp, zp
O Real(KIND=8), allocatable :: qst2(:,),c2(:,),v2(:,)
O Integer :: inic, nfin, ntipo, medio, i, j, k, n1, n2,npa, nea,mpos
O CHARACTER(LEN=90) :: a$, ab$
O CHARACTER(LEN=40) :: tipoq$(3),tipoym$
O CHARACTER(LEN=40) :: tipoa$(3)
O vcritica = 0.1D-3 ! Velocidad que en amort. por rozam. dinám.pasa a estático
O dpi = 4.D0 * DATan(1.D0)
O g=9.80665D0
O porcen=0.D0
O ! Lectura de los datos generales.....
O ab$ = Trim(nom$) // "\" // "Datos.Dat"
O OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=110)
O READ(UNIT=1, FMT=1002, ERR=110) np
O READ(UNIT=1, FMT=1002, ERR=110) ne
O READ(UNIT=1, FMT=1002, ERR=110) nea
O Close(1)
O ! Datos generales
O ab$=Trim(nom$) // "\" // "Parametr."// extension$
O OPEN(UNIT=2, FILE =ab$,STATUS="OLD", ACTION="READ", ERR=130)
O READ(UNIT=2, FMT=1000,ERR=130) a$ ! Barra de títulos
O READ(UNIT=2, FMT=1003,ERR=130) dt
O READ(UNIT=2, FMT=1002,ERR=130) itergrb
O READ(UNIT=2, FMT=1006,ERR=130) amortt$
O READ(UNIT=2, FMT=1003,ERR=130) alambda
O READ(UNIT=2, FMT=1012,ERR=130) itermax
O READ(UNIT=2, FMT=1003,ERR=130) velmax
O READ(UNIT=2, FMT=1001,ERR=130) ppropio$
O READ(UNIT=2, FMT=1001,ERR=130) nnd$
O READ(UNIT=2, FMT=1005,ERR=130) extension2$
O READ(UNIT=2, FMT=1001,ERR=130) cont$
O READ(UNIT=2, FMT=1001,ERR=130) axil$
O READ(UNIT=2, FMT=1006,ERR=130) comentario$
O READ(UNIT=2, FMT=1001,ERR=130) xpos$
O READ(UNIT=2, FMT=1001,ERR=130) xvel$
O READ(UNIT=2, FMT=1001,ERR=130) xaxi$
O READ(UNIT=2, FMT=1001,ERR=130) xcor$
O READ(UNIT=2, FMT=1001,ERR=130) xfle$
O Close(2)
O
O ! Lectura de las cargas DINAMICAS de los puntos.....
O ! npqd n° de puntos de la tabla de la carga dinámica
O ! tipoq$ tipo de tabla de carga dinámica
O ! Prqd fichero con las propiedades de las cargas dinámicas

```

```

O ! tabqd tablas de cargas dinámicas
O ! Datos de las CARGAS DINÁMICAS en Kg f0 y f1 y t0 y t1 seg.
O   ab$ = Trim(nom$) // "\" // "Prqd.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=150)
O   Do i = -3 , 6
O     READ(UNIT=1,FMT=1003,ERR=150) f0(i) ! fuerza en Kgf
O     READ(UNIT=1,FMT=1003,ERR=150) f1(i) ! fuerza en Kgf
O     READ(UNIT=1,FMT=1003,ERR=150) t0(i) ! tiempo en seg
O     READ(UNIT=1,FMT=1003,ERR=150) t1(i) ! tiempo en seg
O   End do
O   Close(1)
O
O ! Tabla 1
O   ab$ = Trim(nom$) // "\" // "Cargd1.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ", ERR=160)
O   READ (UNIT=1,FMT=1002,ERR=160) npqd(1)
O   READ(UNIT=1,FMT=1006,ERR=160) tipoq$(1)
O   Allocate (tabqd1(npqd(1)))
O
O   Do i=1 , npqd(1)
O     READ(UNIT=1,FMT=1003,ERR=160) tabqd1(i)
O   End do
O   Close(1)
O
O ! Tabla 2
O   ab$ = Trim(nom$) // "\" // "Cargd2.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=170)
O   READ(UNIT=1,FMT=1002,ERR=170) npqd(2)
O   READ(UNIT=1,FMT=1004,ERR=170) tipoq$(2)
O   Allocate (tabqd2(npqd(2)))
O   Do i=1,npqd(2)
O     READ(UNIT=1,FMT=1003,ERR=170) tabqd2(i)
O   End do
O   Close(1)
O
O ! Tabla 3
O   ab$ = Trim(nom$) // "\" // "Cargd3.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=175)
O   READ(UNIT=1,FMT=1002,ERR=175) npqd(3)
O   READ(UNIT=1,FMT=1004,ERR=175) tipoq$(3)
O
O   Allocate (tabqd3(npqd(3)))
O   Do i=1,npqd(3)
O     READ(UNIT=1,FMT=1003,ERR=175) tabqd3(i)
O   End do
O   Close(1)
O
O ! Datos de las Aceleraciones en m/s*s f0 y f1 y t0 y t1 seg.
O   ab$ = Trim(nom$) // "\" // "Arqd.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=151)
O   Do i = -3 , 6
O     READ(UNIT=1,FMT=1003,ERR=151) af0(i) ! fuerza en Kgf
O     READ(UNIT=1,FMT=1003,ERR=151) af1(i) ! fuerza en Kgf
O     READ(UNIT=1,FMT=1003,ERR=151) at0(i) ! tiempo en seg
O     READ(UNIT=1,FMT=1003,ERR=151) at1(i) ! tiempo en seg
O   End do
O   Close(1)
O
O ! Tabla 1
O   ab$ = Trim(nom$) // "\" // "Ad1.Dat"
O   OPEN (UNIT=1, FILE= ab$ , STATUS="OLD", ACTION="READ", ERR=152)
O   READ (UNIT=1,FMT=1002,ERR=152) npad(1)
O   READ(UNIT=1,FMT=1006,ERR=152) tipoa$(1)
O   Allocate (tabad1(npad(1)))
O   Do i=1 , npad(1)
O     READ(UNIT=1,FMT=1003,ERR=152) tabad1(i)
O   End do
O   Close(1)
O
O ! Tabla 2
O   ab$ = Trim(nom$) // "\" // "Ad2.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=153)
O   READ(UNIT=1,FMT=1002,ERR=153) npad(2)
O   READ(UNIT=1,FMT=1004,ERR=153) tipoa$(2)
O   Allocate (tabad2(npad(2)))
O   Do i=1,npad(2)
O     READ(UNIT=1,FMT=1003,ERR=153) tabad2(i)
O   End do
O   Close(1)

```

```

O
O ! Tabla 3
O   ab$ = Trim(nom$) // "\" // "Ad3.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=154)
O   READ (UNIT=1,FMT=1002,ERR=154) npad(3)
O   READ (UNIT=1,FMT=1004,ERR=154) tipoa$(3)
O   Allocate (tabad3(npad(3)))
O   Do i=1,npad(3)
O     READ(UNIT=1,FMT=1003,ERR=154) tabad3(i)
O   End do
O   Close(1)
O
O ! Lectura de los tipos de barras.....
O   ab$ = Trim(nom$) // "\" // "Tipos.Dat"
O   OPEN(UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=180)
O   READ (UNIT=1,FMT=1004,ERR=180) tipotit$
O   READ (UNIT=1,FMT=1002,ERR=180) ntp
O   Allocate (p(ntp, 8))
O   Allocate (pp$(ntp))
O
O   Do i =1 , ntp
O   ! area-cm2 inertr-cm4 MY+Kg/c2 def+ MY-Kg/c2 def- Tmax-Kg/c2 P.esp-K/m3 Nombre
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 1)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 2)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 3)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 4)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 5)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 6)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 7)
O   READ (UNIT=1,FMT=1003,ERR=180) p(i, 8)
O   READ (UNIT=1,FMT=1006,ERR=180) pp$(i)
O   End do
O   Close(1)
O
O ! Tabla ym
O   ab$ = Trim(nom$) // "\" // "Young.Dat"
O   OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=155)
O   READ (UNIT=1,FMT=1002,ERR=155) npym
O   READ (UNIT=1,FMT=1004,ERR=155) tipoym$
O
O   Allocate (tabym(npym))
O   Do i=1,npym
O     READ(UNIT=1, FMT=1003,ERR=155) tabym(i)
O   End do
O   READ (UNIT=1,FMT=1003,ERR=155) kc
O   Close(1)
O
O ! np: n° de nodos extremos
O ! npa: n° de nodos añadidos
O ! ne: n° de barras
O ! nea: n° de barras añadidas
O ! np2: n° de nodos totales
O ! ne2: n° de barras totales
O
O   If (mnd$ == "Si") Then
O     npa = ne
O     ndim = 5
O   Else
O     npa = 0
O     ndim = 3
O   End If
O
O   np2 = np + npa !preparamos para añadir un nodo intermedio en cada barra....
O   ne2 = ne + nea
O
O ! lon(ne2) Longitudes de las barras aplicado el pretensado en m.
O ! p(ntp, *) Propiedades de las barras ( rea cm2, m.inercia cm4 (+/-),mod. Young
O           Kg/cm2, t.adm kg/cm2, peso esp.kg/m3)
O ! pp$(ntp) Nombres descriptivos de los tipos de barras
O
O   Allocate (nll(ne2, 3)) !Datos enteros de las barras (nodo or., fin, tipo)
O   Allocate (alon(ne2, 0:2)) !Alon(ne2,0) Longitudes iniciales de las barras en m.
O                               !Alon(ne2,1)=lon.min
O                               !Alon(ne2,2)=lon.máx
O   ALlocate (slon1(ne2)) !Semilongitud barra
O   Allocate (amas(np2)) !Masa de los nudos
O   Allocate (c(np2, 3)) !Coordenadas de los nodos, en m.
O   Allocate (ca(np2,2))
O   Allocate (c2(np2, 3))

```



```

O      Allocate (c0(np2, 3))
O      Allocate (v(np2, 3))          !Velocidad de cada nodo en m/s
O      Allocate (va(np2, 2))
O      Allocate (v2(np2, 3))
O      Allocate (nx(np2, 3))        !Coacciones en las posiciones de los nodos.
O      Allocate (anx(np2, 3))       !Coacciones inferiores en las posiciones de los nodos.
O      Allocate (bnx(np2, 3))       !Coacciones superiores en las posiciones de los nodos.
O      Allocate (na(np2,3))         !Aceleraciones de las cargas
O      Allocate (nq(np2, 3))
O      Allocate (qst(np2, 3))       !Cargas estáticas en nudos, en Kg.
O      Allocate (qst2(np2, 3))
O      Allocate (ainer(np2))        !Momento de Inercia longitudinal del nudo intermedio
O
O ! ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O !                               Lectura de valores iniciales
O ! ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O
O ! Lectura de matriz de giro
O
O      ab$ = Trim(nom$) // "\" // "Giro.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=195)
O      READ(UNIT=1,FMT=1003,ERR=195) alfa
O      READ(UNIT=1,FMT=1003,ERR=195) beta
O      Close(1)
O
O ! alfa=0.D0
O ! alfa=0.864298369489712D-1
O ! alfa=datan(1.D0)
O ! beta=datan(1.D0)
O ! beta=0.D0
O ! beta=0.654537854162998D-1
O ! alfa=0.864298369489712D0
O ! beta=0.654537854162998D0
O
O      coalfa=Dcos(alfa)
O      cobeta=Dcos(beta)
O      sialfa=Dsin(alfa)
O      sibeta=Dsin(beta)
O ! coalfa=0 996267266173759D+0
O ! cobeta=0.997858665642831D+0
O ! sialfa=8.632227030532232D-2
O ! sibeta=6.540705926357934D-2
O
O      tt(1,1)=cobeta
O      tt(1,2)=sibeta
O      tt(1,3)=0.D0
O      tt(2,1)=-sibeta*coalfa
O      tt(2,2)=coalfa*cobeta
O      tt(2,3)=sialfa
O      tt(3,1)=sialfa*sibeta
O      tt(3,2)=-sialfa*cobeta
O      tt(3,3)=coalfa
O
O      tti=transpose(tt)
O
O ! Lectura de las cargas estáticas externas de los
O      puntos.....
O      ab$ = Trim(nom$) // "\" // "Cargas.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=190)
O      READ(UNIT=1,FMT=1003,ERR=190) ((qst2(i,j), j = 1, 3), i =1, np)
O      Close(1)
O      qst2=qst2*1000.D0          ! Paso de Kg a Tn
O      if (ndim==5) then
O        Do i=1,np
O          qst(i,1)=tt(1,1)*qst2(i,1)+tt(1,2)*qst2(i,2)
O          qst(i,2)=tt(2,1)*qst2(i,1)+tt(2,2)*qst2(i,2)+tt(2,3)*qst2(i,3)
O          qst(i,3)=tt(3,1)*qst2(i,1)+tt(3,2)*qst2(i,2)+tt(3,3)*qst2(i,3)
O        End do
O      Else
O        qst=qst2
O      End if
O
O ! Lectura de las coacciones y aceleraciones exteriores de los nudos .....
O      ab$ = Trim(nom$) // "\" // "Coaccion.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=200)
O      do i=1,np
O        do j=1,3
O          READ(UNIT=1,FMT=1014,ERR=200) nx(i,j)
O        end do

```

```

O      end do
O      Close(1)
O
O      ab$ = Trim(nom$) // "\" // "Coacclmt.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ", ERR=202)
O      do i=1,np
O      do j=1,3
O      READ(UNIT=1,FMT=1015,ERR=202) anx(i,j),bnx(i,j)
O      end do
O      end do
O      Close(1)
O
O      ! los nudos intermedios no se coaccionan
O      If (ndim == 5) Then
O      nx(np+1:np2,1:3)=1234
O      End If
O
O      !lectura de las aceleraciones dinámicas
O      ab$ = Trim(nom$) // "\" // "Aceded.Dat"
O      OPEN (UNIT=1, FILE= ab$ , STATUS="OLD", ACTION="READ", ERR=205)
O      READ(UNIT=1,FMT=1002,ERR=205) ((na(i,j), j = 1, 3), i =1, np)
O      Close(1)
O
O      ! LOCALIZACIÓN DE LAS CARGAS DINÁMICAS
O
O      ab$ = Trim(nom$) // "\" // "Cargasd.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=210)
O      READ(UNIT=1,FMT=1002,ERR=210) ((nq(i,j), j = 1, 3), i =1, np)
O      Close(1)
O
O      ! LONGITUDES BARRAS En equilibrio inicial sin
O      tensión.....
O      ab$=Trim(nom$) // "\" // "Longitud.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=220)
O      READ(UNIT=1,FMT=1003,ERR=220) (alon(k, 0),k =1, ne)
O      Close(1)
O
O      ! Lectura de los datos enteros de las barras
O      ab$ = Trim(nom$) // "\" // "Barras.dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=230)
O      READ(UNIT=1,FMT=1002,ERR=230) ((nll(k,j), j = 1, 3), k =1, ne)
O      Close(1)
O
O      If (nea > 0) Then
O      ab$ = Trim(nom$) // "\" // "Barrasa.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=240)
O      READ(UNIT=1,FMT=1002,ERR=240) ((nll(k,j), j = 1, 2), k =ne+1, ne2)
O      FORALL(k = ne + 1:ne2 , j = 1 :2) nll(k,j)=nll(k,j)+np
O      Close(1)
O      End If
O
O      ! Inicialización de variables de posición, velocidad y aceleración
O      If (cont$ == "No") Then
O
O      !.....
O      ! CÁLCULO NUEVO
O      !.....
O      c2=0.D0
O      ab$ = Trim(nom$) // "\" // "Coordena.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=250)
O      READ(UNIT=1, FMT=1003,ERR=250) ((c2(i,j),i =1, np ), j = 1, 3)
O      Close(1)
O      v=0.D0
O
O      If (nnd$ == "Si") Then      ! Cálculo con nudo intermedio
O      !SEMILONGITUDES BARRAS En equilibrio inicial sin tensión.....
O      ab$=Trim(nom$) // "\" // "Slongit.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=300)
O      READ(UNIT=1,FMT=1003,ERR=300) (slon1(k),k =1, ne)
O      Close(1)
O      ca=0.D0
O      va=0.D0
O      c(np+1:np2,1:3)=0.D0
O
O      !Introduce las coordenadas del punto intermedio.....
O      !Los nuevos ne puntos se introducen después de los np existentes
O      ab$ = Trim(nom$) // "\" // "Coordenl.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=310)
O      READ(UNIT=1,FMT=1003,ERR=310) ((c2(np+k,j),j=1,3 ), k = 1, ne)

```

```

O      Close(1)
O      c0=c2
O      !Giro de nudos
O      Do i=1,np2
O          c(i,1)=tt(1,1)*c2(i,1)+tt(1,2)*c2(i,2)
O          c(i,2)=tt(2,1)*c2(i,1)+tt(2,2)*c2(i,2)+tt(2,3)*c2(i,3)
O          c(i,3)=tt(3,1)*c2(i,1)+tt(3,2)*c2(i,2)+tt(3,3)*c2(i,3)
O      End do
O
O      ! Calcula los ángulos iniciales
O      Do k=1,ne
O          inic = nll(k, 1)
O          nfina = nll(k, 2)
O          ntipo = nll(k, 3)
O          medio = np + k
O          x1 = c(inic, 1)
O          y1 = c(inic, 2)
O          z1 = c(inic, 3)
O          x2 = c(nfina, 1)
O          y2= c(nfina, 2)
O          z2= c(nfina, 3)
O          acos1= (x1 - x2)*(x1 - x2)+(y1 - y2)*(y1 - y2)+(z1 - z2)*(z1 - z2)
O          vu=DSqrt (acos1)
O          acos1=(z1 - z2)/vu
O          ca(k, 1) = Dacos(acos1)          !+ 0.5D0
O          !
O          vu=ca(k,1)
O          If (DAbs(y1 - y2) < 0.1D-15 .And. x2 < x1) Then
O              ca(k, 2) = dpi*0.5D0          !+ 0.5
O          End If
O          If (DAbs(y1 - y2) < 0.1D-15 .And. x2 >= x1) Then
O              ca(k, 2) = 1.5D0 * dpi          ! +0.5
O          End If
O          If ((y1 - Y2) > 0.D0) Then
O              vu=(x1- x2)
O              acos1=(y2 - y1)
O              vu=vu/acos1
O              ca(k, 2) = dpi + DAtan(vu)          ! - 0.5
O          End If
O          If ((y1 - y2) < 0.D0) Then
O              vu=(x1- x2)
O              acos1=(y2 - y1)
O              vu=vu/acos1
O              ca(k, 2) = DAtan(vu)          !- 0.5
O          End If
O          If (ca(k, 2) < 0.D0) Then
O              ca(k, 2) = ca(k, 2) + 2.D0*dpi
O          End If
O
O      !Pone la masa de las barras en los nudos
O      al = alon(k, 0)
O      If (pp$(ntipo) == "Sin.definir") Then
O          Stop !Error 101
O      End if
O
O      amasabar = p(ntipo, 1) * al * p(ntipo, 8) / 1.D4 ! en Kg....
O
O      amas(inic) = (1.D0-0.6D0)*slon1(k) * amasabar/al          ! Distribución de masas1
O      amas(medio) = 0.6D0 * amasabar          ! Distribución de masas
O      amas(nfina) =(1.D0-0.6D0)*(al-slon1(k)) * amasabar/al          ! Distribución de
O          masas 2
O
O      !Pone el momento de inercia longitudinal
O      ! ainer(medio) = amas(medio) * al * al * 0.55D0 * 0.55D0 / 12.D0
O      ainer(medio) = amas(medio) * al * al*0.6D0*0.6D0/ 12.D0
O      !Pone las longitudes actuales en alon(i,1) y alon(i,2)
O      xp = x2 - x1
O      yp = y2 - y1
O      zp = z2 - z1          !Vector 1-2
O      ale = DSqrt(xp * xp + yp * yp + zp * zp)          !long.actual
O      alon(k, 1) = ale
O      alon(k, 2) = ale
O      End do
O
O      Else
O
O      ! Lectura de los datos de las barras.. (2 nudos).....
O      c=c2
O      Do k=1,ne
O      !Pone la masa de las barras en los nudos.....

```

```

O      ntipo = nll(k, 3)
O      al = alon(k, 0)
O      n1 = nll(k, 1)          ! nudo inicial
O      n2 = nll(k, 2)          ! nudo final
O      If (pp$(ntipo) == "Sin.definir") Then
O          Stop !Error 101
O      End if
O      amasabar = p(ntipo, 1) * al * p(ntipo, 8) / 1.D4
O      amas(n1) = 0.5D0 * amasabar ! en Kg...
O      amas(n2) = 0.5D0 * amasabar
O
O      !Pone las longitudes actuales en Alon(k,1) y Alon(k,2)
O      x1 = c(n1, 1)
O      y1 = c(n1, 2)
O      z1 = c(n1, 3)          !coordenadas nudos
O      x2 = c(n2, 1)
O      y2 = c(n2, 2)
O      z2 = c(n2, 3)
O      xp = x2 - x1
O      yp = y2 - y1
O      zp = z2 - z1          !Vector 1-2
O      ale = DSqrt(xp * xp + yp * yp + zp * zp)          !long.actual
O      alon(k, 1) = ale
O      alon(k, 2) = ale
O
O      End do
O      End If
O
O      Else
O      !.....
O      ! CÁLCULO CONTINUACIÓN
O      !.....
O      c2=0.D0
O      ab$ = Trim(nom$) // "\" // "Coordena.Dat"
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=250)
O      READ(UNIT=1,FMT=1003,ERR=250) ((c2(i,j),i =1, np), j = 1, 3)
O      Close(1)
O      mpos=1
O
O      If (nnd$ == "Si") Then ! CÁLCULO CON NUDOS INTERMEDIOS
O          c0=0.D0
O          !SEMILONGITUDES BARRAS En equilibrio inicial sin
O              tensión.....
O          ab$=Trim(nom$) // "\" // "Slongit.Dat"
O          OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=300)
O          READ(UNIT=1,FMT=1003,ERR=300) (slon1(k),k =1, ne)
O          Close(1)
O
O          !Introduce las coordenadas del punto intermedio.....
O          !Los nuevos ne puntos se introducen después de los np existentes
O          ab$ = Trim(nom$) // "\" // "Coordenl.Dat"
O          OPEN (UNIT=1, FILE = ab$ , STATUS="OLD", ACTION="READ",ERR=310)
O          READ(UNIT=1,FMT=1003,ERR=310) ((c2(np+k,j),j=1,3 ), k = 1, ne)
O          Close(1)
O          c0=c2
O          c=0.D0
O          v=0.D0
O          ca=0.D0
O          va=0.D0
O          ab$= Trim(nom$) // "\" // "contin." // extension2$
O          OPEN (UNIT=1, FILE = ab$ , STATUS="OLD",
O              ACTION="READ",ERR=260,ACCESS="DIRECT",RECL=2)
O
O          Do j=1,3
O          Do i=1,np2
O              READ(UNIT=1,REC=mpos,ERR=260) c(i, j)
O              mpos=mpos+1
O              READ(UNIT=1,REC=mpos,ERR=260) v(i, j) ! Lee desplazamientos y velocidades..
O              mpos=mpos+1
O          End do
O          End do
O
O          Do j =1,2
O          Do i=1,ne
O              READ(UNIT=1,REC=mpos,ERR=260) ca(i, j)
O              mpos=mpos+1
O              READ(UNIT=1,REC=mpos,ERR=260) va(i, j) ! Graba desplazamientos y velocidades..
O              mpos=mpos+1
O          End do
O          End do
O
O      End do

```

```

O      !Datos de los nuevos nudos...
O      Do k=1,ne
O          medio = np + k          !nudo intermedio
O          inic = nll(k, 1)
O          nfina = nll(k, 2)
O          ntipo = nll(k, 3)
O
O      !Pone las longitudes actuales en Alon(k,1) y Alon(k,2)
O      x1 = c(inic, 1)
O      y1 = c(inic, 2)
O      z1 = c(inic, 3)
O      x2 = c(nfina, 1)
O      y2 = c(nfina, 2)
O      z2 = c(nfina, 3)
O      xp = x2 - x1
O      YP = Y2 - y1
O      zp = z2 - z1      !Vector 1-2
O      ale = DSqrt(xp * xp + yp * yp + zp * zp)      !long.actual
O      alon(k, 1) = ale
O      alon(k, 2) = ale
O
O      !Pone la masa de las barras en los nudos
O      al = alon(k, 0)
O      If (pp$(ntipo) == "Sin.definir") Then
O          Stop !Error 101
O      End if
O      amasabar = p(ntipo, 1) * al * p(ntipo, 8) / 1.D4 ! en Kg...
O      amas(inic) = (1.D0-0.6D0)*slon1(k) * amasabar/al      ! Distrib. masas1
O      amas(medio) = 0.6D0 * amasabar      ! Distrib. masas
O      amas(nfina) =(1.D0-0.6D0)*(al-slon1(k)) * amasabar/al      ! Distrib. masas 2
O
O      !Pone el momento de inercia longitudinal
O      ainer(medio) = amas(medio) * 0.6D0*0.6D0*al * al / 12.D0
O      !ainer(medio) = amas(medio) * al * al * 0.55D0 * 0.55D0 / 12.D0
O      End do
O      CLOSE(1)
O
O      Else      ! Lectura de los datos de las barras...(2 nudos).....
O
O      ab$ = Trim(nom$) // "\" // "contin." // extension2$
O      OPEN (UNIT=1, FILE = ab$ , STATUS="OLD",
O          ACTION="READ",ERR=260,ACCESS="DIRECT",RECL=2)
O
O      Do j=1,3
O          Do i=1,np2
O              READ(UNIT=1,REC=mpos,ERR=260) c(i, j)
O              mpos=mpos+1
O              READ(UNIT=1,REC=mpos,ERR=260) v(i, j) ! Graba desplazamientos y velocidades..
O              mpos=mpos+1
O          End do
O      End do
O
O      Do k=1,ne
O          !Pone la masa de las barras en los nudos.....
O          ntipo = nll(k, 3)
O          al = alon(k, 0)
O          n1 = nll(k, 1)      ! nudo inicial
O          n2 = nll(k, 2)      ! nudo final
O          If (pp$(ntipo) == "Sin.definir") Then
O              Stop ! Error 101
O          End if
O          amasabar = p(ntipo, 1) * al * p(ntipo, 8) / 1.D4
O          amas(n1) = 0.5D0 * amasabar ! en Kg...
O          amas(n2) = 0.5D0 * amasabar
O
O          !Pone las longitudes actuales en Alon(k,1) y Alon(k,2)
O          x1 = c(n1, 1)
O          y1 = c(n1, 2)
O          z1 = c(n1, 3)      !coordenadas nudos
O          x2 = c(n2, 1)
O          y2 = c(n2, 2)
O          z2 = c(n2, 3)
O          xp = x2 - x1
O          YP = Y2 - y1
O          zp = z2 - z1      !Vector 1-2
O          ale = DSqrt(xp * xp + yp * yp + zp * zp)      !long.actual
O          alon(k, 1) = ale
O          alon(k, 2) = ale
O      End do
O      CLOSE(1)

```

```

○ End If
○ End If
○
○ ! Consideración del peso propio
○ If (ppropio$ == "Si") Then
○   if (ndim==5) then
○     Do i = 1,np2
○       qst(i,2)=qst(i, 2) - tt(2,3)*amas(i)
○       qst(i,3)=qst(i, 3) - tt(3,3)*amas(i)           ! Kgf
○     End do
○   else
○     Do i = 1,np2
○       qst(i,3)=qst(i, 3) - amas(i)           ! Kgf
○     End do
○   End if
○ End If
○ !.....
○ Deallocate (qst2,c2,v2)
○ Return
○
○ 110 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222,FMT=1000) "Error al acceder al fichero Datos"
○ Stop
○ 130 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Parametr"
○ Stop
○ 140 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Datos"
○ Stop
○ 150 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Prqd"
○ Stop
○ 151 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Arqd"
○ Stop
○ 152 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Ad1"
○ Stop
○ 153 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Ad2"
○ Stop
○ 154 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222,FMT=1000) "Error al acceder al fichero Ad3"
○ Stop
○ 155 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222,FMT=1000) "Error al acceder al fichero Young"
○ Stop
○ 160 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222,FMT=1000) "Error al acceder al fichero Cargd1"
○ Stop
○ 170 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE (UNIT=222, FMT=1000) "Error al acceder al fichero Cargd2"
○ Stop
○ 175 Continue
○ ab$ = Trim(nom$) // "\" // "Error.Dat"
○ OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
○ WRITE(UNIT=222,FMT=1000) "Error al acceder al fichero Cargd3"

```

```

O Stop
O 180 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Tipos"
O Stop
O 190 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Cargas"
O Stop
O 195 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Giro"
O Stop
O 200 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Coaccion"
O Stop
O 202 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Coacclmt"
O Stop
O 205 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Aceled"
O Stop
O 210 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Cargasd"
O Stop
O 220 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Longitud"
O Stop
O 230 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE (UNIT=222, FMT=1000) "Error al acceder al fichero Barras"
O Stop
O 240 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Barrasa"
O Stop
O 250 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Coordena"
O Stop
O 260 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero contin"
O Stop
O 300 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Slongit"
O Stop
O 310 Continue
O ab$ = Trim(nom$) // "\" // "Error.Dat"
O OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O WRITE(UNIT=222, FMT=1000) "Error al acceder al fichero Coordlenl"
O Stop
O
O 1000 Format (A80)
O 1001 Format (A2)
O 1002 Format (I6)
O 1003 Format (D16.8)
O 1004 Format (A80)
O 1005 Format (A3)
O 1006 Format (A20)

```

```

O 1007 Format(A35)
O 1012 Format(I10)
O 1014 Format(I12)
O 1015 Format(D12.5,D12.5)
O End Subroutine Lectura_Datos
O
O !*****
O !.....
O !.....
O !.....
O !.....
O !*****
O
O Subroutine Nucleol(dlg)
O
O use dfwin
O use dflogm
O !!use Fortran3WinGlobals
O
O REAL(KIND=8) :: cpa(ne, 2, 4), vpa(ne, 2, 4), cka(ne, 2), vka(ne, 2)
O REAL(KIND=8) :: AK1aa(ne, 2), AK2aa(ne, 2), AK3aa(ne, 2), AK4aa(ne, 2)
O REAL(KIND=8) :: AK1ba(ne, 2), AK2ba(ne, 2), AK3ba(ne, 2), AK4ba(ne, 2)
O REAL(KIND=8) :: AK1b(np2, 3), AK2b(np2, 3), AK3b(np2, 3), AK4b(np2, 3)
O REAL(KIND=8) :: apa(ne, 2, 4), aca(ne, 2)
O REAL(KIND=8) :: ap(np2, 3, 4), ac(np2, 3)
O REAL(KIND=8) :: AK1a(np2, 3), AK2a(np2, 3), AK3a(np2, 3), AK4a(np2, 3)
O REAL(KIND=8) :: cp(np2, 3, 4), vp(np2, 3, 4), ck(np2, 3), vk(np2, 3), fueraxil(ne,2),
O fuercort(ne,2), flec(ne,2)
O REAL(KIND=8) :: tiempo, desp, vabs, cx(3), vx(3), vy(3), cy(3), xa
O REAL(KIND=8) :: desplaz
O REAL(KIND=8), allocatable :: c2(:, :), v2(:, :)
O Integer :: i, j, k, nparar, ncim1, nci, ncil, nci2, nci3,
O mpos, mvel, maxi, mcor, mfle, nitera
O Integer*4 :: itete, litergrb
O Integer*4 :: iterac, hwnD
O integer*4 :: ret
O CHARACTER(LEN=62) :: a$, b$, d$, e$, f$, ab$
O
O logical :: lret, lNotQuit
O type(dialog) :: dlg
O !!include "resource.fd"
O
O ! COMIENZO DE LAS ITERACIONES.....
O iterac = 1_4 ! N° de la iteración actual.
O
O litergrb=itergrb
O !GOSUB 1500 ! Actualiza pretensado de las barras....
O
O a$ = Trim(nom$) // "\" // "resultc." // extension$
O b$ = Trim(nom$) // "\" // "resultv." // extension$
O d$ = Trim(nom$) // "\" // "axil." // extension$
O e$ = Trim(nom$) // "\" // "cort." // extension$
O f$ = Trim(nom$) // "\" // "flec." // extension$
O
O if (xpos$=="Si") Then
O OPEN (UNIT=1, FILE = a$, STATUS="REPLACE",
O ACTION="WRITE", ERR=360, ACCESS="DIRECT", RECL=2)
O End if
O If (xvel$=="Si") Then
O OPEN (UNIT=2, FILE = b$, STATUS="REPLACE",
O ACTION="WRITE", ERR=365, ACCESS="DIRECT", RECL=2)
O End if
O nparar = 0
O AK1a=0.D0
O AK2a=0.D0
O AK3a=0.D0
O AK4a=0.D0
O AK1aa=0.D0
O AK2aa=0.D0
O AK3aa=0.D0
O AK4aa=0.D0
O AK1b=0.D0
O AK2b=0.D0
O AK3b=0.D0
O AK4b=0.D0
O AK1ba=0.D0
O AK2ba=0.D0
O AK3ba=0.D0
O AK4ba=0.D0

```



```

O      vk=0.D0
O      ck=0.D0
O      vka=0.D0
O      cka=0.D0
O      ac=0.D0
O      aca=0.D0
O      cp=0.D0
O      cpa=0.D0
O      vp=0.D0
O      vpa=0.D0
O      ap=0.D0
O      apa=0.D0
O
O      If(ndim == 3) Then
O      if (xaxi$=="Si") Then
O      OPEN (UNIT=9, FILE = d$, STATUS="REPLACE", ACTION="WRITE",ERR=330,
          ACCESS="DIRECT",RECL=2)
O      End if
O      Else
O      Allocate (c2(np2,3),v2(np2,3))
O      if (xaxi$=="Si") Then
O      OPEN (UNIT=9, FILE = d$, STATUS="REPLACE", ACTION="WRITE", ERR=330,
          ACCESS="DIRECT", RECL=2)
O      End if
O      If (xcor$=="Si") Then
O      OPEN (UNIT=7, FILE = e$, STATUS="REPLACE", ACTION="WRITE", ERR=340,
          ACCESS="DIRECT", RECL=2 )
O      End if
O      If (xfle$=="Si") Then
O      OPEN (UNIT=8, FILE = f$, STATUS="REPLACE", ACTION="WRITE", ERR=350, ACCESS="DIRECT",
          RECL=2)
O      End if
O      c2=0.D0
O      v2=0.D0
O      End If
O
O      mpos=1
O      mvel=1
O      maxi=1
O      mcor=1
O      mfle=1
O
O      desplaz = 0.D0
O      lNotQuit = .true.
O      Do while (lNotQuit)
O      !! if (gDoWork) then
O      !!Do 101 While (nparar==0 .and. PeekMessage(msg, 0, 0, 0, PM_NOREMOVE) == 0 .and.
          gDoWork)
O      Do 101 While (nparar==0 )
O      ! Do work here.
O      ! RUNGE-KUTTA
O      if (ndim==3) then
O      cp(1:np, 1:3, iterac) = c(1:np, 1:3)
O      vp(1:np, 1:3, iterac) = v(1:np, 1:3)
O      ap(1:np, 1:3, iterac) = ac(1:np, 1:3)
O      else
O      cp(1:np2, 1:3, iterac) = c(1:np2, 1:3)
O      vp(1:np2, 1:3, iterac) = v(1:np2, 1:3)
O      ap(1:np2, 1:3, iterac) = ac(1:np2, 1:3)
O      cpa(1:ne, 1:2, iterac) = ca(1:ne, 1:2)
O      vpa(1:ne, 1:2, iterac) = va(1:ne, 1:2)
O      apa(1:ne, 1:2, iterac) = aca(1:ne, 1:2)
O      end if
O
O      !CALCULO DE K1.....
O      tiempo = dt * iterac
O      If (ndim == 3) Then
O      ck(1:np,1:3)=c(1:np,1:3)
O      vk(1: np, 1:3)=v(1: np, 1:3)
O
O      Call A2n(ck, vk, ac, tiempo, fueraxil) !calcula aceleración sin nudo intermedio
O      Do j=1,3
O      Do i=1,np2
O      AK1a(i, j) = vk(i, j) * dt
O      AK1b(i, j) = ac(i, j) * dt
O      End do
O      End do
O      Else
O      ! Nudo intermedio

```

```

O      ck(1:np2,1:3)=c(1:np2,1:3)
O      vk(1:np2,1:3)=v(1:np2,1:3)
O      cka(1:ne,1:2)=ca(1:ne,1:2)
O      vka(1:ne,1:2)=va(1:ne,1:2)
O      Call A3n(ck,cka, vk,vka, ac,aca, tiempo, fueraxil, fuercort, flec) !calcula
O          aceleración con nudo intermedio
O
O      Do j=1,3
O      Do i=1,np2
O          AK1a(i, j) = vk(i, j) * dt
O          AK1b(i, j) = ac(i, j) * dt
O      End do
O      End do
O
O      Do j =1,2
O      Do i=1,ne
O          AK1aa(i, j) = vka(i, j) * dt
O          AK1ba(i, j) = aca(i, j) * dt
O      End do
O      End do
O      End If
O
O      !CALCULO DE K2.....
O      tiempo = dt * iterac + dt * 0.5D0
O      If (ndim == 3) Then
O      Do j=1,3
O      Do i=1,np2
O          ck(i, j) = c(i, j) + 0.5D0 * AK1a(i, j)
O          vk(i, j) = v(i, j) + 0.5D0 * AK1b(i, j)
O      End do
O      End do
O
O      Call A2n(ck, vk, ac, tiempo, fueraxil)
O      Do j=1,3
O      Do i=1,np2
O          AK2a(i, j) = vk(i, j) * dt
O          AK2b(i, j) = ac(i, j) * dt
O      End do
O      End do
O      Else
O      ! Nudo intermedio
O      Do j=1,3
O      Do i=1,np2
O          ck(i, j) = c(i, j) + 0.5D0 * AK1a(i, j)
O          vk(i, j) = v(i, j) + 0.5D0 * AK1b(i, j)
O      End do
O      End do
O
O      Do j=1,2
O      Do i=1,ne
O          cka(i, j) = ca(i, j) + 0.5D0 * AK1aa(i, j)
O          vka(i, j) = va(i, j) + 0.5D0 * AK1ba(i, j)
O      End do
O      End do
O
O      Call A3n(ck,cka, vk,vka, ac,aca, tiempo, fueraxil, fuercort, flec) !calcula
O          aceleración...
O
O      Do j=1,3
O      Do i=1,np2
O          AK2a(i, j) = vk(i, j) * dt
O          AK2b(i, j) = ac(i, j) * dt
O      End do
O      End do
O
O      Do j =1,2
O      Do i=1,ne
O          AK2aa(i, j) = vka(i, j) * dt
O          AK2ba(i, j) = aca(i, j) * dt
O      End do
O      End do
O      End If
O
O      !CALCULO DE K3.....
O      tiempo = dt * iterac + dt * 0.5D0
O      If (ndim == 3) Then
O      Do j=1,3
O      Do i=1,np2
O          ck(i, j) = c(i, j) + 0.5D0 * AK2a(i, j)

```

```

O      vk(i, j) = v(i, j) + 0.5D0 * AK2b(i, j)
O      End do
O      End do
O
O      Call A2n(ck, vk, ac, tiempo, fueraxil) !calcula aceleración
O      Do j=1,3
O          Do i=1,np2
O              AK3a(i, j) = vk(i, j) * dt
O              AK3b(i, j) = ac(i, j) * dt
O          End do
O      End do
O
O      Else
O      !Nudo intermedio
O      Do j=1,3
O          Do i=1,np2
O              ck(i, j) = c(i, j) + 0.5D0 * AK2a(i, j)
O              vk(i, j) = v(i, j) + 0.5D0 * AK2b(i, j)
O          End do
O      End do
O
O      Do j=1,2
O          Do i=1,ne
O              cka(i, j) = ca(i, j) + 0.5D0 * AK2aa(i, j)
O              vka(i, j) = va(i, j) + 0.5D0 * AK2ba(i, j)
O          End do
O      End do
O      Call A3n(ck,cka, vk,vka, ac,aca, tiempo, fueraxil, fuerkort, flec)      !calcula
O                                          aceleración con nudo intermedio
O
O      Do j=1,3
O          Do i=1,np2
O              AK3a(i, j) = vk(i, j) * dt
O              AK3b(i, j) = ac(i, j) * dt
O          End do
O      End do
O
O      Do j =1,2
O          Do i=1,ne
O              AK3aa(i, j) = vka(i, j) * dt
O              AK3ba(i, j) = aca(i, j) * dt
O          End do
O      End do
O      End If
O
O      !CALCULO DE K4.....
O      tiempo = dt * iterac + dt
O      If (ndim == 3) Then
O          Do j=1,3
O              Do i=1,np2
O                  ck(i, j) = c(i, j) + AK3a(i, j)
O                  vk(i, j) = v(i, j) + AK3b(i, j)
O              End do
O          End do
O
O          Call A2n(ck, vk, ac, tiempo, fueraxil) !calc aceleración...
O          Do j=1,3
O              Do i=1,np2
O                  AK4a(i, j) = vk(i, j) * dt
O                  AK4b(i, j) = ac(i, j) * dt
O              End do
O          End do
O      Else
O
O          Do j=1,3
O              Do i=1,np2
O                  ck(i, j) = c(i, j) + AK3a(i, j)
O                  vk(i, j) = v(i, j) + AK3b(i, j)
O              End do
O          End do
O
O          Do j=1,2
O              Do i=1,ne
O                  cka(i, j) = ca(i, j) + AK3aa(i, j)
O                  vka(i, j) = va(i, j) + AK3ba(i, j)
O              End do
O          End do
O
O
O

```

```

O      Call A3n(ck,cka, vk,vka, ac,aca, tiempo, fueraxil, fuercort, flec) !calcula
O          aceleración...
O      Do j=1,3
O      Do i=1,np2
O          AK4a(i, j) = vk(i, j) * dt
O          AK4b(i, j) = ac(i, j) * dt
O      End do
O      End do
O
O      Do j=1,2
O      Do i=1,ne
O          AK4aa(i, j) = vka(i, j) * dt
O          AK4ba(i, j) = aca(i, j) * dt
O      End do
O      End do
O  End If
O
O  !CALCULO DE LOS DESPLAZAMIENTOS por Runge-Kutta.....
O  If (ndim == 3) Then
O  ! sin nudo intermedio
O      Do j=1,3
O      Do i=1,np2
O          desp = (AK1a(i, j) + 2.D0 * AK2a(i, j) + 2.D0 * AK3a(i, j) + AK4a(i, j)) / 6.D0
O          c(i, j) = c(i, j) + desp
O          v(i, j) = v(i, j) + (AK1b(i, j) + 2.D0 * AK2b(i, j) + 2.D0 * AK3b(i, j) +
O              AK4b(i, j))/6.D0
O          vabs = DAbs(v(i, j))
O          If (vabs > velmax) Then
O              v(i, j) = DSign(velmax,v(i, j))
O              Call Avisovel(i, 0) ! Límite de velocidad..
O          Stop
O          End If
O          If (Abs(desp) > DAbs(desplaz)) Then !desplazamiento máximo
O              desplaz = desp
O          End if
O      End do
O  End do
O
O  Do i=1,np
O      If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O      Else If (nx(i, 1) /= 1234 .And.nx(i, 2) == 1234.And.nx(i, 3) ==1234) Then
O          v(i,1)=0.D0
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234.And.nx(i, 3) == 1234) Then
O          v(i,2)=0.D0
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O          v(i,3)=0.D0
O      Else If (nx(i, 1) /= 1234 .And.nx(i, 2) /= 1234.And.nx(i, 3) == 1234) Then
O          v(i,1)=0.D0
O          v(i,2)=0.D0
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O          v(i,1)=0.D0
O          v(i,3)=0.D0
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O          v(i,2)=0.D0
O          v(i,3)=0.D0
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O          v(i,2)=0.D0
O          v(i,3)=0.D0
O          v(i,1)=0.D0
O      End If
O  End do
O
O      Call A2n(c, v, ac, tiempo, fueraxil)
O      !ARCHIVO c,v.....
O      If (Mod(iterac,litergrb) == 0_4) Then
O          Do j=1,3
O          Do i=1,np2
O              If (xpos$=="Si") Then
O                  WRITE(UNIT=1,REC=mpos,ERR=360) c(i, j)
O                  mpos=mpos+1
O              End if
O              If (xvel$=="Si") Then
O                  WRITE(UNIT=2,REC=mvel,ERR=365) v(i, j) ! Graba desplazamientos y velocidades..
O                  mvel=mvel+1
O              End if
O          End do
O      End do
O
O
O

```

```

O      Do k=1,ne
O      xa=fueraxil(k,1)/g
O      if (xaxi$=="Si") Then
O          WRITE(UNIT=9,REC=maxi,ERR=330) xa
O          maxi=maxi+1
O      End if
O      End do
O      nitera=Int(iterac/litergrb)
O      porcen= Real((100*iterac))/Real(itermax)
O      !! nporcen= aint(porcen)
O      !! lret = DlgSet(dlg, IDC_PROGRESS1, nporcen)
O      !! call DlgFlush(dlg)
O      Print 850,nitera,porcen !!
O      End If
O      Else
O      ! Nudo intermedio
O      Do j=1,3
O      Do i=1,np2
O      desp = (AK1a(i, j) + 2.D0 * AK2a(i, j) + 2.D0 * AK3a(i, j) + AK4a(i, j)) / 6.D0
O      c(i, j) = c(i, j) + desp
O      v(i, j) = v(i, j) + (AK1b(i, j) + 2.D0 * AK2b(i, j) + 2.D0 * AK3b(i, j) +
O          AK4b(i, j)) / 6.D0
O      vabs = DAbs(v(i, j))
O      If (vabs > velmax) Then
O          v(i, j) = DSign(velmax,v(i, j))
O          Call Avisovel(i, 0) ! Límite de velocidad...
O          Stop
O      End If
O      If (Dabs(desp) > DAbs(desplaz)) Then
O          desplaz = desp
O      End if
O      End do
O      End do
O
O      Do j=1,2
O      Do i=1,ne
O      ca(i, j) = ca(i, j) + (AK1aa(i, j) + 2.D0 * AK2aa(i, j) + 2.D0 * AK3aa(i, j) +
O          AK4aa(i, j)) / 6.D0
O      va(i, j) = va(i, j) + (AK1ba(i, j) + 2.D0 * AK2ba(i, j) + 2.D0 * AK3ba(i, j) +
O          AK4ba(i, j)) / 6.D0
O      End do
O      End do
O
O      Do i=1,np
O      If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(1)=c0(i,1)
O      vy(1)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti, cx)
O      vy=matmul(tti,vx)
O      cy(2)=c0(i,2)
O      vy(2)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(3)=c0(i,3)
O      vy(3)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)

```

```

O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti, vx)
O      cy(1)=c0(i,1)
O      vy(1)=0.D0
O      cy(2)=c0(i,2)
O      vy(2)=0.D0
O      cx=matmul(tt,cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(1)=c0(i,1)
O      vy(1)=0.D0
O      cy(3)=c0(i,3)
O      vy(3)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(2)=c0(i,2)
O      vy(2)=0.D0
O      cy(3)=c0(i,3)
O      vy(3)=0.D0
O      cx=matmul(tt,cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O
O      Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(1)=c0(i,1)
O      vy(1)=0.D0
O      cy(2)=c0(i,2)
O      vy(2)=0.D0
O      cy(3)=c0(i,3)
O      vy(3)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O      End If
O      End do
O
O      Call A3n(c,ca, v,va, ac,aca, tiempo, fueraxil, fuercort, flec)
O      !ARCHIVO c,v
O      If (Mod(iterac,litergrb) == 0_4) Then
O      Do i=1,np2
O      c2(i,1)=tt(1,1)*c(i,1)+tt(2,1)*c(i,2)+tt(3,1)*c(i,3)
O      c2(i,2)=tt(1,2)*c(i,1)+tt(2,2)*c(i,2)+tt(3,2)*c(i,3)
O      c2(i,3)=          tt(2,3)*c(i,2)+tt(3,3)*c(i,3)
O
O      v2(i,1)=tt(1,1)*v(i,1)+tt(2,1)*v(i,2)+tt(3,1)*v(i,3)
O      v2(i,2)=tt(1,2)*v(i,1)+tt(2,2)*v(i,2)+tt(3,2)*v(i,3)
O      v2(i,3)=          tt(2,3)*v(i,2)+tt(3,3)*v(i,3)
O      End do
O
O      Do j=1,3
O      Do i=1,np2

```

```

O      If (xpos$=="Si") Then
O        WRITE(UNIT=1,REC=mpos,ERR=360) c2(i, j)
O        mpos=mpos+1
O      End if
O      If (xvel$=="Si") Then
O        WRITE (UNIT=2, REC=mvel, ERR=365) v2(i, j) ! Graba desplazamientos y
O          velocidades..
O        mvel=mvel+1
O      End if
O    End do
O  End do
O
O  Do j=1,2
O    Do i=1,ne
O      If (xpos$=="Si") Then
O        WRITE(UNIT=1,REC=mpos,ERR=360) ca(i, j)
O        mpos=mpos+1
O      End if
O      If (xvel$=="Si") Then
O        WRITE(UNIT=2,REC=mvel,ERR=365) va(i, j)
O        mvel=mvel+1
O      End if
O    End do
O  End do
O
O  Do k=1,ne
O    Do j = 1 , 2
O      if (xaxi$=="si") Then
O        xa=fueraxil(k, j)/g
O        WRITE(UNIT=9,REC=maxi,ERR=330) xa
O        maxi=maxi+1
O      End if
O      if (xcor$=="si") Then
O        xa=fuercort(k, j)/g
O        WRITE(UNIT=7,REC=mcors,ERR=340) xa
O        mcor=mcor+1
O      End if
O      If (xfle$=="Si") Then
O        xa=flec(k, j)/g
O        WRITE(UNIT=8,REC=mfle,ERR=350) xa
O        mfle=mfle+1
O      End if
O    End do
O  End do
O
O  nitera=Int(iterac/litergrb)
O  porcen=Real((100*iterac))/Real(itermax)
O  !! nporcen= aint(porcen)
O  !! lret =DlgSet(dlg, IDC_PROGRESS1, nporcen)
O  !! Call DlgFlush(dlg)
O  Print 850, nitera, porcen !!
O
O  End If
O End If
O
O      If (iterac == 4_4) Then
O        nparar = 1 ! para arrancar con Adams-Moulton
O        !!gDowork=.FALSE.
O      End if
O      iterac = iterac + 1_4
O 101 end do
O end do
O
O !CALCULO DE LOS DESPLAZAMIENTOS Adams-Moulton.....
O nparar = 0
O !!gDoWork= .true.
O lNotQuit= .true.
O do while (lNotQuit .and. nparar==0)
O Do 102 While (nparar==0) !!
O tiempo = dt * iterac
O itete = iterac-2_4
O nciml = Mod((itete+1_4), 4_4) + 1_4
O nci = Mod(itete, 4_4) + 1_4
O nci1 = Mod((itete-1_4), 4_4) + 1_4
O nci2 = Mod((itete-2_4), 4_4) + 1_4
O nci3 = nciml
O
O !PREDICCION.....
O ! Dos nudos

```

```

O If (ndim == 3) Then
O   Do j=1,3
O     Do i=1,np2
O       ck(i, j) = cp(i, j, nci) + dt * (55 * vp(i, j, nci) - 59 * vp(i, j, nci1) + 37 *
O         vp(i, j, nci2) - 9 * vp(i, j, nci3)) / 24.D0
O       vk(i, j) = vp(i, j, nci) + dt * (55 * ap(i, j, nci) - 59 * ap(i, j, nci1) + 37 *
O         ap(i, j, nci2) - 9 * ap(i, j, nci3)) / 24.D0
O     End do
O   End do
O
O   Do i=1,np
O     If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O       vk(i,1)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O       vk (i,2)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O       vk(i,3)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O       vk(i,1)=0.D0
O       vk(i,2)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O       vk(i,1)=0.D0
O       vk(i,3)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O       vk(i,2)=0.D0
O       vk(i,3)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O       vk(i,1)=0.D0
O       vk(i,2)=0.D0
O       vk(i,3)=0.D0
O     End If
O   End do
O   Call A2n(ck, vk, ac, tiempo, fueraxil) !calcula aceleración...
O
O Else
O   !Nudo intermedio
O   Do j=1,3
O     Do i=1,np2
O       ck(i, j) = cp(i, j, nci) + dt * (55 * vp(i, j, nci) - 59 * vp(i, j, nci1) + 37 *
O         vp(i, j, nci2) - 9 * vp(i, j, nci3)) / 24.D0
O       vk(i, j) = vp(i, j, nci) + dt * (55 * ap(i, j, nci) - 59 * ap(i, j, nci1) + 37 *
O         ap(i, j, nci2) - 9 * ap(i, j, nci3)) / 24.D0
O     End do
O   End do
O
O   Do i=1,np
O     If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O       cx(1:3)=ck(i,1:3)
O       vx (1:3)=vk(i,1:3)
O       cy=matmul(tti, cx)
O       vy=matmul(tti, vx)
O       cy(1)=c0(i,1)
O       vy(1)=0.D0
O       cx=matmul(tt, cy)
O       vx=matmul(tt, vy)
O       ck(i,1:3)=cx(1:3)
O       vk(i,1:3)=vx(1:3)
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O       cx(1:3)=ck(i,1:3)
O       vx (1:3)=vk(i, 1:3)
O       cy=matmul(tti, cx)
O       vy=matmul(tti, vx)
O       cy(2)=c0(i,2)
O       vy(2)=0.D0
O       cx=matmul(tt, cy)
O       vx=matmul(tt, vy)
O       ck(i,1:3)=cx(1:3)
O       vk(i,1:3)=vx(1:3)
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O       cx(1:3)=ck(i,1:3)

```



```

O   vx(1:3)=vk(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   ck(i,1:3)=cx(1:3)
O   vk(i,1:3)=vx(1:3)
O
O   Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O   cx(1:3)=ck(i,1:3)
O   vx(1:3)=vk(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   ck(i,1:3)=cx(1:3)
O   vk(i,1:3)=vx(1:3)
O
O   Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O   cx(1:3)=ck(i,1:3)
O   vx(1:3)=vk(i,1:3)
O   cy=matmul(tti, cx)
O   vy=matmul(tti, vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt, cy)
O   vx=matmul(tt,vy)
O   ck(i,1:3)=cx(1:3)
O   vk(i,1:3)=vx(1:3)
O
O   Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O   cx(1:3)=ck(i,1:3)
O   vx(1:3)=vk(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti, vx)
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   ck(i,1:3)=cx(1:3)
O   vk(i,1:3)=vx(1:3)
O
O   Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O   cx(1:3)=ck(i,1:3)
O   vx(1:3)=vk(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt, cy)
O   vx=matmul(tt,vy)
O   ck(i,1:3)=cx(1:3)
O   vk(i,1:3)=vx(1:3)
O   End If
O   End do
O
O   Do j=1,2
O   Do i=1,ne
O   cka(i, j) = cpa(i, j, nci) + dt * (55 * vpa(i, j, nci) - 59 * vpa(i, j, nci1) + 37
O   * vpa(i, j, nci2) - 9 * vpa(i, j, nci3)) / 24.D0
O   vka(i, j) = vpa(i, j, nci) + dt * (55 * apa(i, j, nci) - 59 * apa(i, j, nci1) + 37
O   * apa(i, j, nci2) - 9 * apa(i, j, nci3)) / 24.D0
O   End do
O   End do
O

```

```

O Call A3n(ck,cka, vk,vka, ac,aca, tiempo, fueraxil, fuercort, flec) !calcula
aceleración...
O End If
O
O !CORRECCION.....
O !Dos nudos
O If (ndim == 3) Then
O Do j=1,3
O Do i=1,np2
O v(i, j) = vp(i, j, nci) + dt * (9 * ac(i, j) + 19 * ap(i, j, nci) - 5 * ap(i, j,
nci1) + ap(i, j, nci2)) / 24.D0
O vabs = DAbs(v(i, j))
O If (vabs > velmax) Then
O v(i, j) = DSign(velmax,v(i, j))
O Call Avisovel(i, 0) !Límite de velocidad...
O Stop
O End If
O desp = dt * (9 * v(i, j) + 19 * vp(i, j, nci) - 5 * vp(i, j, nci1) + vp(i, j,
nci2)) / 24.D0
O c(i, j) = cp(i, j, nci) + desp
O If (DAbs(desplaz) > DAbs(desplaz)) Then
O desplaz = desp
O End if
O End do
O End do
O
O Do i=1,np2
O If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O v(i, 1) = 0.D0
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234.And.nx(i, 3) == 1234) Then
O v(i, 2) =0.D0
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O v(i, 3)=0.D0
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O v(i, 1)=0.D0
O v(i, 2)=0.D0
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O v(i, 1)=0.D0
O v(i, 3)=0.D0
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O v(i, 2)=0.D0
O v(i, 3)=0.D0
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O v(i, 2) = 0.D0
O v(i, 3) = 0.D0
O v(i, 1) = 0.D0
O End If
O End do
O
O Call A2n(c, v, ac, tiempo, fueraxil) !calcula aceleración...
O cp(1:np2,1:3,ncim1)=c(1:np2,1:3)
O vp(1:np2,1:3,ncim1)=v(1:np2,1:3)
O ap(1:np2,1:3,ncim1)=ac(1:np2,1:3)
O
O Else
O ! Modelo con nudos intermedios
O Do j=1,3
O Do i=1,np2
O v(i, j) = vp(i, j, nci) + dt * (9 * ac(i, j) + 19 * ap(i, j, nci) - 5 * ap(i, j,
nci1) + ap(i, j, nci2)) / 24.D0
O ! Milne
O vabs = DAbs(v(i, j))
O If (vabs > velmax) Then
O v(i, j) = DSign(velmax,v(i, j))
O Call Avisovel(i, 0) ! Límite de velocidad...
O Stop
O End If
O desp = dt * (9 * v(i, j) + 19 * vp(i, j, nci) - 5 * vp(i, j, nci1) + vp(i, j,
nci2)) / 24.D0
O c(i, j) = cp(i, j, nci) + desp
O If (DAbs(desplaz) > DAbs(desplaz)) Then
O desplaz = desp
O End if
O End do
O End do
O
O Do i=1,np
O If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then

```

```

O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i,3) /= 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti, vx)
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti, cx)
O   vy=matmul(tti,vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cx=matmul(tt, cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(1)=c0(i,1)
O   vy(1)=0.D0
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O   cx(1:3)=c(i,1:3)
O   vx(1:3)=v(i,1:3)
O   cy=matmul(tti,cx)
O   vy=matmul(tti,vx)
O   cy(2)=c0(i,2)
O   vy(2)=0.D0
O   cy(3)=c0(i,3)
O   vy(3)=0.D0
O   cx=matmul(tt,cy)
O   vx=matmul(tt,vy)
O   c(i,1:3)=cx(1:3)
O   v(i,1:3)=vx(1:3)
O
O Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then

```

```

O      cx(1:3)=c(i,1:3)
O      vx(1:3)=v(i,1:3)
O      cy=matmul(tti,cx)
O      vy=matmul(tti,vx)
O      cy(1)=c0(i,1)
O      vy(1)=0.D0
O      cy(2)=c0(i,2)
O      vy(2)=0.D0
O      cy(3)=c0(i,3)
O      vy(3)=0.D0
O      cx=matmul(tt, cy)
O      vx=matmul(tt,vy)
O      c(i,1:3)=cx(1:3)
O      v(i,1:3)=vx(1:3)
O      End If
O      End do
O      Do i=1,ne
O      Do j=1,2
O      va(i, j) = vpa(i, j, nci) +dt * (9*aca(i, j) + 19* apa(i, j, nci) -5* apa(i, j,
ncil)+apa(i,j,nci2)) / 24.D0
O      ca(i, j) = cpa(i, j, nci) +dt * (9*va(i, j) + 19 * vpa(i, j, nci) -5* vpa(i, j,
ncil)+vpa(i,j,nci2)) / 24.D0
O      cpa(i, j, nci) = ca(i, j)
O      vpa(i, j, nci) = va(i, j)
O      End do
O      If (ca(i, 2) > 2.D0 * dpi) Then
O      cpa(i, 2,ncim1) = cpa(i, 2,ncim1) - 2.D0* dpi
O      cpa(i, 2,nci) = cpa(i, 2,nci) - 2.D0* dpi
O      cpa(i, 2,ncil) = cpa(i, 2,ncil) - 2.D0* dpi
O      cpa(i, 2,nci2) = cpa(i, 2,nci2) - 2.D0* dpi
O      End if
O      If (ca(i, 2) < 0.D0 ) Then
O      cpa(i, 2,ncim1) = cpa(i, 2,ncim1) + 2.D0* dpi
O      cpa(i, 2,nci) = cpa(i, 2,nci) + 2.D0* dpi
O      cpa(i, 2,ncil) = cpa(i, 2,ncil) + 2.D0* dpi
O      cpa(i, 2,nci2) = cpa(i, 2,nci2) + 2.D0* dpi
O      End if
O      End do
O      Call A3n(c,ca, v,va, ac,aca, tiempo, fueraxil, fuercort, flec) !calcula
aceleración...
O      cp(1:np2,1:3,ncim1)= c(1:np2,1:3)
O      vp(1:np2,1:3,ncim1)= v(1:np2,1:3)
O      ap(1:np2,1:3,ncim1)=ac(1:np2,1:3)
O      apa(1:ne,1:2,ncim1)=aca(1:ne,1:2)
O      End If
O
O      !ARCHIVO c,v.....
O      If (ndim == 3) Then
O      If (Mod(iterac,litergrb) == 0_4) Then
O      Do j=1,3
O      Do i=1,np2
O      If (xpos$=="Si") Then
O      WRITE(UNIT=1,REC=mpos,ERR=360) c(i, j)
O      mpos =mpos + 1
O      End if
O      If (xvel$=="Si") Then
O      WRITE (UNIT=2,REC=mvel, ERR=365) v(i, j) ! Graba desplazamientos y velocidades..
O      mvel=mvel+1
O      End if
O      End do
O      End do
O      Do k=1,ne
O      If (xaxi$=="Si") Then
O      xa=fueraxil(k,1)/g
O      WRITE(UNIT=9,REC=maxi,ERR=330) xa
O      maxi=maxi+1
O      End if
O      End do
O      End If
O      nitera=Int(iterac/litergrb)
O      porcen= Real((100*iterac))/Real(itermax)
O      Print 850,nitera,porcen !!
O      Else
O      ! nudo intermedio
O      If (Mod(iterac,litergrb) == 0_4) Then
O      Do i=1,np2
O      c2(i,1)=tt(1,1)*c(i,1)+tt(2,1)*c(i,2)+tt(3,1)*c(i,3)
O      c2(i,2)=tt(1,2)*c(i,1)+tt(2,2)*c(i,2)+tt(3,2)*c(i,3)
O      c2(i,3)=
tt(2,3)*c(i,2)+tt(3,3)*c(i,3)

```

```

O      v2(i,1)=tt(1,1)*v(i,1)+tt(2,1)*v(i,2)+tt(3,1)*v(i,3)
O      v2(i,2)=tt(1,2)*v(i,1)+tt(2,2)*v(i,2)+tt(3,2)*v(i,3)
O      v2(i,3)=          tt(2,3)*v(i,2)+tt(3,3)*v(i,3)
O      End do
O      Do j=1,3
O      Do i=1,np2
O      If (xpos$=="Si") Then
O      WRITE(UNIT=1,REC=mpos,ERR=360) c2(i, j)
O      mpos=mpos+1
O      End if
O      If (xvel$=="Si") Then
O      WRITE (UNIT=2,REC=mvel, ERR=365) v2(i, j) ! Graba desplazamientos y velocidades..
O      mvel=mvel+1
O      End if
O      End do
O      End do
O
O      Do j =1,2
O      Do i=1,ne
O      If (xpos$=="Si") Then
O      WRITE(UNIT=1,REC=mpos,ERR=360) ca(i, j)
O      mpos=mpos+1
O      End if
O      If (xvel$=="Si") Then
O      WRITE(UNIT=2, REC=mvel, ERR=365) va(i, j) ! Graba desplazamientos y velocidades..
O      mvel=mvel+1
O      End if
O      End do
O      End do
O
O      Do k=1,ne
O      Do j = 1,2
O      If (xaxi$=="Si") Then
O      xa=fueraxil(k, j)/g
O      WRITE(UNIT=9,REC=maxi,ERR=330) xa
O      maxi=maxi+1
O      End if
O      If (xcor$=="Si") Then
O      xa=fuercort(k, j)/g
O      WRITE(UNIT=7,REC=mcors,ERR=340) xa
O      mcor=mcor+1
O      End if
O      If (xfle$=="Si") Then
O      xa=flec(k, j)/g
O      WRITE(UNIT=8,REC=mfle,ERR=350) xa
O      mfle=mfle+1
O      End if
O      End do
O      End do
O      nitera=Int(iterac/litergrb)
O      porcen=Real((100*iterac))/Real(itermax)
O      Print 850,nitera,porcen !!
O      End If
O      End If
O      If (iterac == itermax) Then
O      nparar = 1 ! Finaliza si pasa el n°m x. de iteraciones
O      End if
O      iterac = iterac + 1_4
O      102 End DO
O      end do
O
O      Deallocate (c2,v2)
O      Close(1)
O      Close(2)
O      Close(9)
O      Close(7)
O      Close(8)
O
O      ! Fichero de continuación
O      a$= Trim(nom$) // "\" // "contin." // extension$
O      OPEN (UNIT=1, FILE = a$,STATUS="REPLACE",
O      ACTION="WRITE",ERR=370,ACCESS="DIRECT",RECL=2)
O      mpos=1
O      If (ndim == 3) Then
O      Do j=1,3
O      Do i=1,np2
O      WRITE(UNIT=1,REC=mpos,ERR=370) c(i, j)
O      mpos=mpos+1
O      WRITE (UNIT=1,REC=mpos, ERR=370) v(i, j) ! Graba desplazamientos y velocidades..

```

```

O      mpos=mpos+1
O      End do
O      End do
O      Else
O      ! nudo intermedio
O      Do j=1,3
O      Do i=1,np2
O      WRITE (UNIT=1, REC=mpos, ERR=370) c(i, j)
O      mpos=mpos+1
O      WRITE(UNIT=1,REC=mpos,ERR=370) v(i, j) ! Graba desplazamientos y velocidades..
O      mpos=mpos+1
O      End do
O      End do
O      Do j =1,2
O      Do i=1,ne
O      WRITE(UNIT=1,REC=mpos,ERR=370) ca(i, j)
O      mpos=mpos+1
O      WRITE(UNIT=1,REC=mpos,ERR=370) va(i, j) ! Graba desplazamientos y velocidades..
O      mpos=mpos+1
O      End do
O      End do
O      End If
O      Close(1)
O      Return
O
O      370 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222, FMT=1000) "Error al escribir el fichero contin"
O      Stop
O      360 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222,FMT=1000) "Error al escribir el fichero resultc"
O      Stop
O      365 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222, FMT=1000) "Error al escribir el fichero resultv"
O      Stop
O      350 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222, FMT=1000) "Error al escribir el fichero flec"
O      Stop
O      340 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222, FMT=1000) "Error al escribir el fichero cort"
O      Stop
O      330 Continue
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE= ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222, FMT=1000) "Error al escribir el fichero axil"
O      Stop
O
O      850 Format (I5,10X,F5.1,"%")
O      1000 Format (A80)
O      End Subroutine Nucleo1
O
O      !-----
O      Subroutine A2n(ct, vt, a, tiempo, fuer)
O      ! CALCULO DE ESFUERZOS MODELO DOS NUDOS.....
O      Real (KIND=8),INTENT(OUT) :: a(np,3)
O      Real (KIND=8) :: rs(np,3)
O      Real (KIND=8) x1, x2, y1, y2, z1, z2, al, ale, xp, yp, zp, cl, c2, c3, deff, frzamt,zz
O      Real (KIND=8),INTENT(IN) :: ct(np,3),vt(np,3)
O      Real (KIND=8) :: velo
O      Real (KIND=8),INTENT(IN) :: tiempo
O      Real (KIND=8),INTENT(OUT) :: fuer (ne,1)
O      Integer :: i, j, k, ntipof, n1, n2, ntipob
O      !-----
O      !Fuerzas externas en Nw
O      Do j=1,3
O      Do i=1,np
O      ntipof = nq(i, j)
O      rs(i, j) = qst(i, j) * g + qdin(tiempo, ntipof) * g
O      End do
O      End do

```

```

O !-----
O Do k=1,ne !.....
O   n1 = n11(k, 1)           ! nudo inicial
O   n2 = n11(k, 2)           ! nudo final
O   ntipob = n11(k, 3)
O   x1 = ct(n1, 1)
O   y1 = ct(n1, 2)
O   z1 = ct(n1, 3) !coordenadas nudos
O   x2 = ct(n2, 1)
O   y2 = ct(n2, 2)
O   z2 = ct(n2, 3)
O   al = alon(k, 0)          !long.inicial
O   xp = x2 - x1
O   YP = Y2 - y1
O   zp = z2 - z1           !Vector 1-2 ....
O   ale = Sqrt(xp * xp + yp * yp + zp * zp)          !long.actual
O   c1 = xp / ale
O   c2 = yp / ale
O   c3 = zp / ale          !cosenos directores barra
O
O   !Cálculo fuerzas axiles.....
O   deff = (al - ale) / al
O   zz=(ymdin(deff,ntipob) * g * p(ntipob, 1))*deff
O   !cables...
O   If ((p(ntipob,3) < 0.D0).And.(deff > 0.D0)) Then
O     zz=0.D0
O   End if
O   fuer(k,1)=zz
O
O   !Fuerzas desequilibradas (Nw)= cosenos directores * FUER.....
O   rs(n1, 1) = rs(n1, 1) - c1 * zz
O   rs(n1, 2) = rs(n1, 2) - c2 * zz
O   rs(n1, 3) = rs(n1, 3) - c3 * zz
O   rs(n2, 1) = rs(n2, 1) + c1 * zz
O   rs(n2, 2) = rs(n2, 2) + c2 * zz
O   rs(n2, 3) = rs(n2, 3) + c3 * zz
O
O   If (ale < alon(k, 1)) Then
O     alon(k, 1) = ale
O   End if
O   If (ale > alon(k, 2)) Then
O     alon(k, 2) = ale
O   End if
O End do
O
O ! Fuerzas desequilibradas+coacciones
O Do i=1,np
O   do j=1,3
O     if (ct(i,j) >= bnx(i,j)) then
O       rs(i,j)=rs(i,j)-kc*g*(ct(i,j)-bnx(i,j))
O     end if
O     if (ct(i,j) <= anx(i,j)) then
O       rs(i,j)=rs(i,j)-kc*g*(ct(i,j)-anx(i,j))
O     end if
O   end do
O End do
O
O !-----
O ! Cálculo aceleraciones
O Do j=1,3
O   Do i=1,np
O     velo = vt(i, j)
O     If (amortt$ == "Rozamiento") Then
O       If (DAbs(velo) < vcritica) Then
O         frzamt = alambda * 1.2D0 * amas(i)
O         If (frzamt >= DAbs(rs(i, j))) Then
O           frzamt = rs(i, j)
O         Else
O           frzamt = DSign(frzamt,rs(i, j))
O         End If
O       Else
O         frzamt = Dsign(alambda*amas(i),velo)
O       End If
O     Else
O       frzamt = alambda * velo * amas(i) ! Amortiguamiento viscoso
O     End If
O     a(i, j) = (rs(i, j) - frzamt) / amas(i)
O   End do
O End do

```

```

O
O !Coacciones.....
O Do i=1,np
O   If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O       a(i,1)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O       a(i,2)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O       a(i,3)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O       a(i,1)=0.D0
O       a(i,2)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O       a(i,1)=0.D0
O       a(i,3)=0.D0
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O       a(i,2)=0.D0
O       a(i,3)=0.D0
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O       a(i,1)=0.D0
O       a(i,2)=0.D0
O       a(i,3)=0.D0
O     End If
O   End do
O
O !Aceleraciones exteriores
O Do i=1,np
O   ntipof=na(i,1)
O   if (ntipof/=1234) then
O     a(i,1)=adin(tiempo,ntipof)
O   End if
O   ntipof=na(i,2)
O   if (ntipof/=1234) then
O     a(i,2)=adin(tiempo,ntipof)
O   End if
O   ntipof=na(i,3)
O   if (ntipof/=1234) then
O     a(i,3)=adin(tiempo,ntipof)
O   End if
O End do
O ! .....
O Return
O End Subroutine A2n
O
O ! ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O ! ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O ! ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O Subroutine A3n(ct,cta, vt,vta, a,aa, tiempo, fueraxil, fuercort, flector)
O ! CALCULO DE ESFUERZOS MODELO TRES NUDOS.....
O Real (KIND=8), INTENT (OUT) :: a(np2,3),aa(ne,2)
O Real (KIND=8) :: rs(np2,3),rsa(ne,2), zet, eta,ax(3),ay(3),cx(3),cy(3),rsx(3),rsy(3)
O Real (KIND=8) :: al, ale, ale1, ale3,h1,h2,h3, rig, vtt5,vtt4,stt5, stt4,sss,
O   ttt,taadin(3),aadin(3)
O Real (KIND=8) :: deff, frzamt, thetaj1,
O   thetaj3,qaxil,qaxi2,qcort1,qcort2,qflec1,qflec2,recale1,recale3
O Real (KIND=8) :: bb(3), aaa(3), cc(3), dd1(3),dd2(3), vx1(3), vy1(3), vz1(3), vx2(3),
O   vy2(3), vz2(3), fle(3)
O Real (KIND=8), INTENT (IN) ::cta(ne,2),vta(ne,2)
O Real (KIND=8), INTENT (IN) :: ct(np2,3),vt(np2,3)
O Real (KIND=8), INTENT (IN) :: tiempo
O Real (KIND=8), INTENT (OUT) :: fueraxil(ne,2),fuercort(ne,2),flector(ne,2)
O Real (KIND=8) :: velo,ym,x1, x2, x3, y1, y2, y3, z1, z2, z3,all,al3
O Real (KIND=8) :: cqdin(3),tcqdin(3)
O Integer i, j, k, ntipof,ntipog,ntipoh, n1, n2, n3, ntipob, nea
O CHARACTER (LEN=90) :: ab$
O   rs = 0.D0
O   rsa= 0.D0
O   a=0.D0
O   aa=0.D0
O !Fuerzas externas en Nw
O Do i=1,np
O   do k=1,3
O     ntipof=nq(i,k)
O     cqdin(k)=qdin(tiempo,ntipof)*g
O   End do
O   tcqdin=matmul(tt,cqdin)
O Do k=1,3
O   rs(i, k) = qst(i, k) * g + tcqdin(k)

```



```

O      End do
O      End do
O      aaa(3) = 0.D0
O
O      DO k=1,ne !.....
O      n1 = n11(k, 1)          ! nudo inicial
O      n3 = n11(k, 2)          ! nudo final
O      n2 = np + k            ! nudo intermedio
O      ntipob = n11(k, 3)     ! tipo barra
O      al = alon(k, 0)         ! long.equili. barra
O      al1=slon1(k)           ! long.equili .semibarra 1-2
O      al3=al-al1             ! long.equili.semibarra 2-3
O      !coordenadas
O      x1 = ct(n1, 1)         !nudo inicial
O      y1 = ct(n1, 2)
O      z1 = ct(n1, 3)
O      x2 = ct(n2, 1)         !nudo intermedio
O      y2 = ct(n2, 2)
O      z2 = ct(n2, 3)
O      zet = cta(k, 1)
O      eta = cta(k, 2)
O      x3 = ct(n3, 1)         !nudo final
O      y3 = ct(n3, 2)
O      z3 = ct(n3, 3)
O      !longitud real semibarra 1-2
O      ale1=Dsqr((x2 - x1)*(x2 - x1)+(y2 - y1)*(y2 - y1)+(z2 - z1)*(z2 - z1))
O      vx1(1) = (x2 - x1)
O      vx1(2) = (y2 - y1)
O      vx1(3) = (z2 - z1)
O      recale1=1.D0/ale1
O      vx1(1) = vx1(1) / ale1
O      vx1(2) = vx1(2) / ale1
O      vx1(3) = vx1(3) / ale1
O
O      !longitud real semibarra 2-3
O      ale3=Dsqr((x2 - x3)*(x2 - x3)+(y2 - y3)*(y2 - y3)+(z2 - z3)*(z2 - z3))
O      vx2(1) = (x2 - x3)
O      vx2(2) = (y2 - y3)
O      vx2(3) = (z2 - z3)
O      recale3=1.D0/ale3
O      vx2(1) = vx2(1) / ale3
O      vx2(2) = vx2(2) / ale3
O      vx2(3) = vx2(3) / ale3
O      !vectores unitarios locales
O      aaa(1) = DCos(eta)
O      aaa(2) = DSin(eta)
O      bb(1)= DSin(zet) * aaa(2)
O      bb(2)= -DSin(zet)* aaa(1)
O      bb(3)= DCos (zet)
O      Call provect(bb, aaa, cc)
O      !SEMIBARRA 1-2
O      Call provect(vx1, bb, dd1)
O      h1 = vmod(dd1)
O      !Cálculo fuerzas axiles1.....
O      deff = (al1-ale1)/ al1
O      ym=ymdin(deff, ntipob)
O      qaxil= (ym * g * p(ntipob, 1))*deff
O      fueraxil(k, 1) = qaxil
O      thetaj1 = 0.D0
O      fuercort(k, 1) = 0.D0
O      flector(k, 1) = 0.D0
O      qcort1=0.D0
O      qflec1=0.D0
O      If (h1 > 1.D-8) Then      !
O      vz1(1) = dd1(1) / h1
O      vz1(2) = dd1(2) / h1
O      vz1(3) = dd1(3) / h1
O      Call provect(dd1, vx1, vy1)
O      h3=vmod(vy1)
O      vy1(1)=vy1(1)/h3
O      vy1(2)=vy1(2)/h3
O      vy1(3)=vy1(3)/h3
O      thetaj1 = Dasin(h1)
O      rig = (ym * p(ntipob, 2) * 0.1D-3 * g) *recale1
O      sss = 2.D0*2.D0 * rig
O      ttt = 2.D0 * rig
O      If (axil$ == "Si") Then
O      Call Correc(qaxil, rig, ale1, sss, ttt)
O      End If

```

```

O      qcort1=(sss - (ttt * ttt / sss)) * thetaj1 *recale1
O      fuercort(k, 1) = qcort1
O      qflec1=(sss - (ttt * ttt / sss)) * thetaj1
O      flector(k, 1) =qflec1
O      End If
O
O
O      ! SEMIBARRA 2-3
O      Call provect(bb, vx2, dd2)
O      h2 = vmod(dd2)
O      !Cálculo fuerzas axiales2.....
O      deff = (al3-ale3)/ al3
O      ym=ymdin (deff, ntipob)
O      qaxi2= (ym * g * p(ntipob, 1))*deff
O      fueraxil(k, 2) = qaxi2
O      thetaj3 = 0.D0
O      fuercort(k, 2) = 0.D0
O      flector(k, 2) = 0.D0
O      qcort2=0.D0
O      qflec2=0.D0
O      If (h2 > 1.D-8) Then      !comprobar.....
O      vz2(1) = dd2(1) / h2
O      vz2(2) = dd2(2) / h2
O      vz2(3) = dd2(3) / h2
O      Call provect(dd2, vx2, vy2)
O      h3=vmod (vy2)
O      vy2(1)=vy2(1)/h3
O      vy2(2)=vy2(2)/h3
O      vy2(3)=vy2(3)/h3
O      thetaj3 = Dasin(h2)
O      rig = (ym * p(ntipob, 2) * 0.1D-3 * g) *recale3
O      sss = 2.D0*2.D0 * rig
O      ttt = 2.D0 * rig
O      If (axil$ == "Si") Then
O      Call Correc(qaxi2, rig, ale3, sss, ttt)
O      End If
O      qcort2=(sss - (ttt * ttt / sss)) * thetaj3 *recale3
O      fuercort(k, 2) = qcort2
O      qflec2=(sss - (ttt * ttt / sss)) * thetaj3
O      flector(k, 2) =qflec2
O      End If
O
O      ! Cortante y axil
O      rs(n1, 1) = rs(n1, 1) - vx1(1) * qaxi1 + vy1(1) * qcort1
O      rs(n1, 2) = rs(n1, 2) - vx1(2) * qaxi1 + vy1(2) * qcort1
O      rs(n1, 3) = rs(n1, 3) - vx1(3) * qaxi1 + vy1(3) * qcort1
O      rs(n3, 1) = rs(n3, 1) - vx2(1) * qaxi2 + vy2(1) * qcort2
O      rs(n3, 2) = rs(n3, 2) - vx2(2) * qaxi2 + vy2(2) * qcort2
O      rs(n3, 3) = rs(n3, 3) - vx2(3) * qaxi2 + vy2(3) * qcort2
O      rs(n2, 1) = rs(n2, 1) + vx1(1) * qaxi1 - vy1(1) * qcort1 + vx2(1) * qaxi2 -
O      vy2(1) * qcort2
O      rs(n2, 2) = rs(n2, 2) + vx1(2) * qaxi1 - vy1(2) * qcort1 + vx2(2) * qaxi2 -
O      vy2(2) * qcort2
O      rs(n2, 3) = rs(n2, 3) + vx1(3) * qaxi1 - vy1(3) * qcort1 + vx2(3) * qaxi2 -
O      vy2(3) * qcort2
O
O      ! flector
O      fle(1) = vz1(1) * qflec1 + vz2(1) * qflec2
O      fle(2) = vz1(2) * qflec1 + vz2(2) * qflec2
O      fle(3) = vz1(3) * qflec1 + vz2(3) * qflec2
O      rsa(k, 1) = fle(1) * aaa(1) + fle(2) * aaa(2)
O      rsa(k, 2) = fle(1) * cc(1) + fle(2) * cc(2) + fle(3) * cc(3)
O
O      ale = ale1 + ale3
O      If (ale < alon(k, 1)) Then
O      alon(k, 1) = ale !guarda longitud mínima (m)
O      End if
O      If (ale > alon(k, 2)) Then
O      alon(k, 2) = ale !guarda longitud máxima (m)
O      End if
O      End do
O
O      ! Fuerzas desequilibradas+coacciones
O      Do i=1,np
O      cx(1:3)=ct(i,1:3)
O      cy=matmul(tti,cx)
O      rsx=0.D0
O      Do j=1,3
O      if (cy(j) >= bnx(i,j)) then
O      rsx(j)=-kc*g*(cy(j)-bnx(i,j))

```

```

O     end if
O     if (cy(j) <= anx(i,j)) then
O         rsx(j)=-kc*g*(cy(j)-anx(i,j))
O     end if
O     end do
O     rsy=matmul(tt,rsx)
O     rs(i,1:3)=rs(i,1:3)+rsy(1:3)
O End do
O
O ! Cálculo aceleraciones.....
O Do i=1,np2
O     Do j=1,3
O         velo = vt(i, j)
O         If (amortt$ == "Rozamiento") Then
O             If (DAbs(velo) < vcritica) Then
O                 frzamt = alambda * 1.2 * amas(i)
O                 If (frzamt >= DAbs(rs(i, j))) Then
O                     frzamt = rs(i, j)
O                 Else
O                     frzamt = DSign(frzamt,rs(i, j))
O                 End If
O             Else
O                 frzamt = DSign(alambda*amas(i),velo)
O             End If
O         Else
O             frzamt = alambda * velo * amas(i) ! Amortiguamiento viscoso
O         End If
O         a(i, j) = (rs(i, j) - frzamt) / amas(i)
O     End do
O End do
O
O !Coacciones exteriores
O DO i=1,np
O     If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) == 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti,ax)
O         ay(1)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti,ax)
O         ay(2)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti,ax)
O         ay(3)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) == 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti, ax)
O         ay(1)=0.D0
O         ay(2)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) == 1234 .And. nx(i, 3) /= 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti,ax)
O         ay(1)=0.D0
O         ay(3)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) == 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti, ax)
O         ay(2)=0.D0
O         ay(3)=0.D0
O         ax=matmul(tt,ay)
O         a(i,1:3)=ax(1:3)
O     Else If (nx(i, 1) /= 1234 .And. nx(i, 2) /= 1234 .And. nx(i, 3) /= 1234) Then
O         ax(1:3)=a(i,1:3)
O         ay=matmul(tti, ax)
O         ay(1)=0.D0
O         ay(2)=0.D0
O         ay(3)=0.D0

```

```

O      ax=matmul(tt,ay)
O      a(i,1:3)=ax(1:3)
O      End If
O      End do
O
O      ! Aceleraciones exteriores
O      Do i=1,np
O      If (na(i, 1) /= 1234 .And.na(i, 2) == 1234 .And.na(i, 3) == 1234) Then
O      ntipof=na(i,1)
O      a(i,1)=(adin(tiempo,ntipof)-tt(2,1)*a(i,2)-tt(3,1)*a(i,3))/tt(1,1)
O      Else If (na(i,1) == 1234 .And. na(i,2) /= 1234 .And. na(i, 3) == 1234) Then
O      ntipof=na(i,2)
O      a(i,2)=(adin(tiempo,ntipof)-tt(1,2)*a(i,1)-tt(3,2)*a(i,3))/tt(2,2)
O      Else If (na(i, 1) == 1234 .And.na(i, 2) == 1234 .And. na(i, 3) /= 1234) Then
O      ntipof=na(i,3)
O      a(i,3)=(adin(tiempo,ntipof)-tt(1,3)*a(i,1)-tt(2,3)*a(i,2))/tt(3,3)
O      Else If (na(i, 1) /= 1234 .And.na(i, 2) /= 1234.And.na(i, 3) == 1234) Then
O      ntipof=na(i,1)
O      ntipog=na(i,2)
O      a(i,1)=(adin(tiempo,ntipof)-(tt(2,1)*adin(tiempo,ntipog)/tt(2,2))+a(i,3)*((tt(3,2)
O      *tt(2,1)/tt(2,2))-tt(3,1)))/(tt(1,1)-tt(2,1)*tt(1,2)/tt(2,2))
O      a(i,2)=(adin(tiempo,ntipog)-tt(3,2)*a(i,3)-tt(1,2)*a(i,1))/tt(2,2)
O      Else If (na(i, 1) /= 1234 .And. na(i, 2) == 1234 .And. na(i, 3) /= 1234) Then
O      ntipof=na(i,1)
O      ntipog=na(i,3)
O      a(i,1)=(adin(tiempo,ntipof)-adin(tiempo,ntipog)*tt(3,1)/tt(3,3)+a(i,2)*((tt(3,1)
O      *tt(2,3)/tt(3,3))-tt(2,1)))/tt(1,1)
O      a(i,3)=(adin(tiempo,ntipog)-a(i,2)*tt(2,3))/tt(3,3)
O      Else If (na(i, 1) == 1234 .And. na(i, 2) /= 1234 .And. na(i, 3) /= 1234) Then
O      ntipof=na(i,2)
O      ntipog=na(i,3)
O      a(i,2)=(adin(tiempo,ntipof)-a(i,1)*tt(1,2)-adin(tiempo,ntipog)*tt(3,2)/tt(3,3)) /
O      (tt(2,2)-tt(3,2)*tt(2,3)/tt(3,3))
O      a(i,3)=(adin(tiempo,ntipog)-tt(2,3)*a(i,2))/tt(3,3)
O      Else If (na(i, 1) /= 1234 .And. na(i, 2) /= 1234 .And. na(i, 3) /= 1234) Then
O      ntipof=na(i,1)
O      aadin(1)=adin(tiempo,ntipof)
O      ntipog=na(i,2)
O      aadin(2)=adin(tiempo,ntipog)
O      ntipoh=na(i,3)
O      aadin(3)=adin(tiempo,ntipoh)
O      taadin=matmul(tt,aadin)
O      a(i,1)=taadin(1)
O      a(i,2)=taadin(2)
O      a(i,3)=taadin(3)
O      End If
O      End do
O
O      !Barras enlazadas
O      nea = ne2 - ne
O      If (nea > 0) Then
O      Do k = ne + 1 , ne2
O      n1 = nll(k, 1)      ! nudo inicial
O      n2 = nll(k, 2)      ! nudo final
O      Do j=1,3
O      a(n1, j) = a(n1, j) * amas(n1)
O      a(n2, j) = a(n2, j) * amas(n2)
O      a(n1, j) = (a(n1, j) + a(n2, j)) / (amas(n1) + amas(n2))
O      a(n2, j) = a(n1, j)
O      End do
O      End do
O      End If
O      ! Cálculo aceleraciones angulares
O      Do k=1,ne
O      vtt5 = vta(k,2)
O      vtt4 = vta(k,1)
O      stt4 = cta(k,1)
O      stt5=Dcos(stt4)
O      stt4=Dsin(stt4)
O      if (Dabs(stt4)<1.D-5) then
O      ab$ = Trim(nom$) // "\" // "Error.Dat"
O      OPEN (UNIT=222, FILE = ab$ , STATUS="REPLACE", ACTION="Write")
O      WRITE(UNIT=222,FMT=1000) "Error verticalidad"
O      close(222)
O      Stop
O      End if
O      frzamt=ainer(np+k)*alambda*vtt4
O      aa(k, 1) = ((rsa(k, 1)-frzamt) / ainer(np+k)) + vtt5 * vtt5 * stt5 * stt4
O      frzamt=ainer(np+k)*alambda*vtt5*stt4

```

```

O      aa(k, 2) = (((rsa(k, 2)-frzamt) / ainer(np+k)) - 2.D0 * vtt4 * vtt5 * stt5) / stt4
O      ! aa(k,1)=0.D0
O      ! aa(k,2)=0.D0
O      End do
O      Return
O      !6000 ! TRATAMIENTO DE ERRORES.....
O      1000 Format(A80)
O      End Subroutine A3n
O
O      !.....
O      !.....
O      !.....
O
O      Subroutine Correc(p, rig, ale, sss, ttt)
O      Real(KIND=8),INTENT(IN) :: p,rig,ale
O      Real(KIND=8),INTENT(OUT) :: sss,ttt
O      ! Factores de corrección debidos al axil
O      Real(KIND=8) :: u,w
O      If (p < -1.D-10) Then
O          u = DSqrt(-p * ale / rig)
O          w = 2.D0 - 2.D0 * Dcosh(u) + u * Dsinh(u)
O          sss = u * (u * Dcosh(u) - Dsinh(u)) * rig / w
O          ttt = u * (Dsinh(u) - u) * rig / w
O      Else if (p > 1.D-10) Then
O          u = DSqrt(p * ale / rig)
O          w = 2.D0 - 2.D0 * DCos(u) - u * Dsin(u)
O          sss = u * (Dsin(u) - u * DCos(u)) * rig / w
O          ttt = u * ((u - DSin(u))) * rig / w
O      Else
O          sss = 4.D0* rig
O          ttt = 2.D0 * rig
O      End if
O      Return
O      End Subroutine Correc
O      !.....
O      !.....
O      !.....
O
O      Function vmod(u) Result(b)
O      REAL(KIND=8),Intent(IN) :: u(3)
O      REAL(KIND=8) :: a,b
O      a = u(1) * u(1) + u(2) * u(2) + u(3) * u(3)
O      b= DSqrt(a)
O      Return
O      End Function vmod
O      !.....
O      !.....
O      !.....
O
O      Subroutine Avisovel(i,l)
O      Integer,INTENT(in) :: i,l
O      Character(LEN=62)::a$
O      a$ = Trim(nom$) // "\Error.Dat" !anota lo ocurrido en NOTAS.TXT...
O      OPEN (UNIT=3, FILE = a$ , STATUS="REPLACE", ACTION="WRITE",ERR=400)
O      If (l == 0) Then
O          WRITE(UNIT=3,FMT=88,ERR=400) "El nudo n° ", i, " sobrepasó velocidad máxima !!!"
O      Else
O          WRITE(UNIT=3, FMT=88,ERR=400) "El nudo n° ", i, " no converge !!!"
O      End If
O      Close(3)
O      Return
O      400 Continue
O      Print *, "Error al acceder al fichero Error"
O      stop
O      88 Format(A11,I4,A35)
O      End Subroutine Avisovel
O      !.....
O      !.....
O      !.....
O
O      Subroutine provect(u, v, w)
O      Real(KIND=8),INTENT(IN) :: u(3),v(3)
O      Real(KIND=8),INTENT(OUT) :: w(3)
O      ! Multiplica vectorialmente las componentes u por las v obteniendo el resultado w
O      w(1) = u(2) * v(3) - v(2) * u(3)
O      w(2) = v(1) * u(3) - u(1) * v(3)
O      w(3) = u(1) * v(2) - u(2) * v(1)
O      Return
O      End Subroutine provect

```

```

O !.....
O !.....
O
O Function qdin(tiempo, ntipo) Result(b)
O ! Cargas dinámicas en Kgf
O Real(KIND=8),Intent (IN) :: tiempo
O Real(KIND=8) :: b
O Real(KIND=8) :: delt, ti
O Integer,Intent (IN) :: ntipo
O Integer :: nn
O Select Case (ntipo)
O Case (0)
O   b = 0.D0
O Case (1)           ! Función rampa 1
O   If (tiempo < t0(1)) Then
O     b = f0(1)
O   End if
O   If (tiempo <= t1(1) .And. tiempo >= t0(1)) Then
O     b= f0(1) + f1(1) * (tiempo - t0(1)) / (t1(1) - t0(1))
O   End if
O   If (tiempo > t1(1)) Then
O     b = f1(1) + f0(1)
O   End if
O Case (2)           ! Función rampa 2
O   If (tiempo < t0(2)) Then
O     b = f0(2)
O   End if
O   If (tiempo <= t1(2) .And. tiempo >= t0(2)) Then
O     b= f0(2) + f1(2) * (tiempo - t0(2)) / (t1(2) - t0(2))
O   End if
O   If (tiempo > t1(2)) Then
O     b = f1(2) + f0(2)
O   End if
O Case (3)           ! Función rampa 3
O   If (tiempo < t0(3)) Then
O     b = f0(3)
O   End if
O   If (tiempo <= t1(3) .And. tiempo >= t0(3)) Then
O     b= f0(3) + f1(3) * (tiempo - t0(3)) / (t1(3) - t0(3))
O   End if
O   If (tiempo > t1(3)) Then
O     b = f1(3) + f0(3)
O   End if
O Case (4)           ! Función seno 1
O   If (tiempo < t0(4)) Then
O     b = f0(4)
O   End if
O   If (tiempo >= t0(4)) Then
O     b = f0(4) + f1(4) * DSin(2.D0 * dpi * (tiempo - t0(4)) / t1(4))
O   End if
O Case (5)           ! Función seno 2
O   If (tiempo < t0(5)) Then
O     b = f0(5)
O   End if
O   If (tiempo >= t0(5)) Then
O     b = f0(5) + f1(5) * DSin(2.D0 * dpi * (tiempo - t0(5)) / t1(5))
O   End if
O Case (6)           ! Función seno 1
O   If (tiempo < t0(6)) Then
O     b = f0(6)
O   End if
O   If (tiempo >= t0(6)) Then
O     b = f0(6) + f1(6) * DSin(2.D0 * dpi * (tiempo - t0(6)) / t1(6))
O   End if
O Case (-1)          ! Función tabla 1 Interpolación lineal
O   If (tiempo < t0(-1)) Then
O     b = f0(-1)
O   End if
O   If (tiempo >= t0(-1) .And. tiempo <= t1(-1)) Then
O     delt = (t1(-1) - t0(-1)) / npqd(1)
O     nn = IDInt(( tiempo - t0(-1)) / delt)
O     ti = tiempo - t0(-1) - nn * delt
O     b = f0(-1) + f1(-1) * (tabqd1(nn) + ti * (tabqd1(nn + 1) - tabqd1(nn)))
O   End If
O   If (tiempo > t1(-1)) Then
O     b = f1(-1) + f0(-1)
O   End if
O Case (-2)          ! Función tabla 2 Interpolación lineal
O   If (tiempo < t0(-2)) Then

```

```

O      b = f0(-2)
O    End if
O    If (tiempo >= t0(-2) .And. tiempo <= t1(-2)) Then
O      delt = (t1(-2) - t0(-2)) / npqd(2)
O      nn = IDINT(((tiempo - t0(-2)) / delt))
O      ti = tiempo - t0(-2) - nn * delt
O      b = f0(-2) + f1(-2) * (tabqd2(nn) + ti * (tabqd2(nn + 1) - tabqd2(nn)))
O    End If
O    If (tiempo > t1(-2)) Then
O      b = f1(-2) + f0(-2)
O    End if
O    Case (-3)          ! Función tabla 3 Interpolación lineal
O    If (tiempo < t0(-3)) Then
O      b = f0(-3)
O    End if
O    If (tiempo >= t0(-3) .And. tiempo <= t1(-3)) Then
O      delt = (t1(-3) - t0(-3)) / npqd(3)
O      nn = IDINT(((tiempo - t0(-3)) / delt))
O      ti = tiempo - t0(-3) - nn * delt
O      b = f0(-3) + f1(-3) * (tabqd2(nn) + ti * (tabqd3(nn + 1) - tabqd3(nn)))
O    End If
O    If (tiempo > t1(-3)) Then
O      b = f1(-3) + f0(-3)
O    End if
O    Case Default
O      b = 0.D0
O    End Select
O    Return
O  End Function qdin
O
O  !::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
O  Function adin(tiempo, ntipo) Result(b)
O  ! Cargas dinámicas en KgF
O  Real(KIND=8),Intent (IN) :: tiempo
O  Real(KIND=8) :: b
O  Real(KIND=8) :: delt, ti
O  Integer,Intent (IN) :: ntipo
O  Integer :: nn
O
O  Select Case (ntipo)
O  Case (0)
O    b = 0.D0
O  Case (1)          ! Función rampa 1
O    If (tiempo < at0(1)) Then
O      b = af0(1)
O    End if
O    If (tiempo <= at1(1) .And. tiempo >= at0(1)) Then
O      b = af0(1) + af1(1) * (tiempo - at0(1)) / (at1(1) - at0(1))
O    End if
O    If (tiempo > at1(1)) Then
O      b = af1(1) + af0(1)
O    End if
O  Case (2)          ! Función rampa 2
O    If (tiempo < at0(2)) Then
O      b = af0(2)
O    End if
O    If (tiempo <= at1(2) .And. tiempo >= at0(2)) Then
O      b = af0(2) + af1(2) * (tiempo - at0(2)) / (at1(2) - at0(2))
O    End if
O    If (tiempo > at1(2)) Then
O      b = af1(2) + af0(2)
O    End if
O  Case (3)          ! Función rampa 3
O    If (tiempo < at0(3)) Then
O      b = af0(3)
O    End if
O    If (tiempo <= at1(3) .And. tiempo >= at0(3)) Then
O      b = af0(3) + af1(3) * (tiempo - at0(3)) / (at1(3) - at0(3))
O    End if
O    If (tiempo > at1(3)) Then
O      b = af1(3) + af0(3)
O    End if
O  Case (4)          ! Función seno 1
O    If (tiempo < at0(4)) Then
O      b = af0(4)
O    End if
O    If (tiempo >= at0(4)) Then
O      b = af0(4) + af1(4) * DSin(2.D0 * dpi * (tiempo - at0(4)) / at1(4))
O    End if

```

```

O Case (5) ! Función seno 2
O If (tiempo < at0(5)) Then
O b = af0(5)
O End if
O If (tiempo >= at0(5)) Then
O b = af0(5) + afl(5) * DSin(2.D0 * dpi * (tiempo - at0(5)) / at1(5))
O End if
O Case (6) ! Función seno 3
O If (tiempo < at0(6)) Then
O b = af0(6)
O End if
O If (tiempo >= at0(6)) Then
O b = af0(6) + afl(6) * DSin(2.D0 * dpi * (tiempo - at0(6)) / at1(6))
O End if
O Case (-1) ! Función tabla 1 Interpolación lineal
O If (tiempo < at0(-1)) Then
O b = af0(-1)
O End if
O If (tiempo >= at0(-1) .And. tiempo <= at1(-1)) Then
O delt = (at1(-1) - at0(-1)) / npad(1)
O nn = IDInt(((tiempo - at0(-1)) / delt))
O ti = tiempo - at0(-1) - nn * delt
O b = af0(-1) + afl(-1) * (tabad1(nn) + ti * (tabad1(nn + 1) - tabad1(nn)))
O End If
O If (tiempo > at1(-1)) Then
O b = afl(-1) + af0(-1)
O End if
O Case (-2) ! Función tabla 2 Interpolación lineal
O If (tiempo < at0(-2)) Then
O b = af0(-2)
O End if
O If (tiempo >= at0(-2) .And. tiempo <= at1(-2)) Then
O delt = (at1(-2) - at0(-2)) / npad(2)
O nn = IDINT(((tiempo - at0(-2)) / delt))
O ti = tiempo - at0(-2) - nn * delt
O b = af0(-2) + afl(-2) * (tabad2(nn) + ti * (tabad2(nn + 1) - tabad2(nn)))
O End If
O If (tiempo > at1(-2)) Then
O b = afl(-2) + af0(-2)
O End if
O Case (-3) ! Función tabla 3 Interpolación lineal
O If (tiempo < at0(-3)) Then
O b = af0(-3)
O End if
O If (tiempo >= at0(-3) .And. tiempo <= at1(-3)) Then
O delt = (at1(-3) - at0(-3)) / npad(3)
O nn = IDINT(((tiempo - at0(-3)) / delt))
O ti = tiempo - at0(-3) - nn * delt
O b = af0(-3) + afl(-3) * (tabad2(nn) + ti * (tabad3(nn + 1) - tabad3(nn)))
O End If
O If (tiempo > at1(-3)) Then
O b = afl(-3) + af0(-3)
O End if
O Case Default
O b = 0.D0
O End Select
O Return
O End Function adin
O
O End Module Nucleo
O
O

```