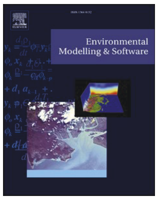
Contents lists available at ScienceDirect

Environmental Modelling and Software

journal homepage: www.elsevier.com/locate/envsoft



# Operationalizing the use of TLS in forest inventories: The R package FORTLS

Juan Alberto Molina-Valero [a,*], Adela Martínez-Calvo [a], María José Ginzo Villamayor [b], Manuel Antonio Novo Pérez [c], Juan Gabriel Álvarez-González [a], Fernando Montes [d], César Pérez-Cruzado [e]

[a] Unidad de Gestión Ambiental y Forestal Sostenible (UXAFORES), Departamento de Ingeniería Agroforestal, Escuela Politécnica Superior de Ingeniería, Universidade de Santiago de Compostela, Benigno Ledo s/n, Campus Terra, 27002 Lugo, Spain
[b] Departamento de Estadística, Análisis Matemático y Optimización, Facultad de Matemáticas, Universidade de Santiago de Compostela, Rúa Lope Gómez de Marzoa s/n, Campus Vida, 15782 Santiago de Compostela, Spain
[c] Departamento de Matemáticas, Facultad de Informática, Universidade da Coruña, Campus Elviña s/n, 15071 A Coruña, Spain
[d] Centro de Investigación Forestal (INIA, CSIC), Ctra. De la Coruña km 7,5, 28040 Madrid, Spain
[e] Proyectos y Planificación (PROEPLA), Departamento de Producción Vegetal y Proyectos de Ingeniería, Escuela Politécnica Superior de Ingeniería, Universidade de Santiago de Compostela, Benigno Ledo s/n, Campus Terra, 27002 Lugo, Spain

ARTICLE INFO

*Keywords:*
Forest monitoring
Forest stands parameters
LiDAR
Precision forestry
Remote sensing
Terrestrial-based-technologies

ABSTRACT

Terrestrial Laser Scanning (TLS) devices show great potential for application in Forest Inventories (FIs) as they are capable of registering high resolution point clouds rapidly and automatically. Nevertheless, operational use of TLS for FI purposes has been hampered by the absence of algorithms for processing the acquired data, particularly in the single-scan mode, as occlusions result in loss of information. The R package FORTLS has been developed to overcome this obstacle, as it automates the processing of single-scan TLS point cloud data for forestry purposes and includes several features that deal with occlusions. FORTLS makes use of the main advantage of the single-scan scenario in FI, thus improving the efficiency of data acquisition and post-processing. All of these features of the FORTLS package are potentially valuable for the operational use of TLS in FIs, in combination with inference techniques derived from model-based and model-assisted approaches.

## 1. Introduction

Information about forest resources is essential for sustainable forest management and development of forest policies, and forest inventories (FIs) are fundamental for estimating and monitoring the state and evolution of forest resources at both global and regional scales (Tomppo et al., 2010). FIs have improved since they were first introduced, owing to the continuous appearance of new technologies, especially in the last few decades since the emergence of remote and proximal sensing. In this technological context, light detection and ranging (LiDAR) systems provide 3-dimensional point clouds, which are suitable for estimating tree attributes and are very useful for many forestry applications (Dubayah and Drake, 2000). This technology has proved operationally viable for estimating essential FI variables at stand level, such as arithmetic mean height ($\bar{h}$, m), basal area ($G$, m$^2$ ha$^{-1}$) and volume ($V$, m$^3$ ha$^{-1}$), with airborne laser scanning (ALS) devices (Wulder et al., 2012; White et al., 2016). Terrestrial laser scanning (TLS) devices, such as LiDAR devices with millimetric precision, are considered to show great potential for enhancing FIs (Dassot et al., 2011; White et al., 2016) and also forest ecology research (Calders et al., 2020; Danson et al., 2018). Apart from much higher spatial resolution under canopy, the main advantages of using TLS data rather than ALS data are better observation of near-ground vegetation (White et al., 2016) and thus better trunk coverage for estimating the woody component, which is one of the most important components in FIs. In fact, TLS-based approaches can provide very accurate estimates of the diameter at breast height (*dbh,* measured at 1.3 m from the ground) and the stem curve; with the single-scan approach yielding values of 1–4 and 1.3–6 cm respectively,

* Corresponding author.
*E-mail addresses:* juanalberto.molina.valero@usc.es (J.A. Molina-Valero), adela.martinez.calvo@usc.es (A. Martínez-Calvo), mariajose.ginzo@usc.es (M.J. Ginzo Villamayor), manuel.antonio.novo.perez@udc.es (M.A. Novo Pérez), juangabriel.alvarez@usc.es (J.G. Álvarez-González), fmontes@inia.es (F. Montes), cesar.cruzado@usc.es (C. Pérez-Cruzado).

https://doi.org/10.1016/j.envsoft.2022.105337
Received 13 September 2021; Received in revised form 18 January 2022; Accepted 24 January 2022
Available online 28 January 2022

depending on the stand conditions (Liang et al., 2018a), which are close to the values required in practical applications such as FIs. However, TLS devices have not been yet adopted in FIs, for several reasons: (i) difficulties in the automation of data processing to provide reliable measurements of important forest variables, (ii) high acquisition costs; (iii) limited software; and (iv) lack of trained personnel (Liang et al., 2016). Many researchers agree that affordability is the main key challenge to overcome, emphasizing that automation of point cloud processing with attainable and easy-to-use software able to extract information related to important forest attributes is essential (Dassot et al., 2011; Newnham et al., 2015; White et al., 2016; Liang et al., 2016, 2018a).

As TLS data sets comprise millions of points, sophisticated methods for automatic processing are required. Many algorithms with a high level of automation and that are able to extract tree attributes, such as *dbh*, total height (*h*, m) and stem volume (*v*, m$^3$), have been developed in the last few decades (Cabo et al., 2018; Liang et al., 2012, 2018b; Olofsson et al., 2014; Olofsson and Holmgren, 2016; Zhang et al., 2019). Although most algorithms yield acceptable *dbh* and stem curve estimations according to FI requirements, stem detection and estimation of *h* cause bottlenecks in the process, especially with single scans (Krok et al., 2020; Liang et al., 2018a). Some of the algorithms developed have also been included in software applications, such as SimpleForest (Hackenberg et al., 2015), 3D Forest (Trochta et al., 2017) and AutoStem™ (Bienert et al., 2007), among others (Krok et al., 2020). However, these programs have some drawbacks for use in FIs: (i) they focus on single-tree rather than stand-level approaches (SimpleForest); (ii) they involve semi-automatic processing (3D Forest); and (iii) the software is only available commercially (AutoStem™) (i.e. it is not free or open source). Furthermore, the previous studies have mainly focused on replicating plot-based measurements, which does not extend conventional inventory approaches from a sampling perspective and thus limits the utility of TLS in FIs, the main purpose of which is to estimate important forest variables at larger scales (e.g. stand and regional) than tree and plot levels (Newnham et al., 2015; White et al., 2016). Methods that enable TLS to be used for FI purposes must therefore contemplate the use of different approaches (Newnham et al., 2015), and other procedures in which not all trees in the sample plots are measured may be feasible (Liang et al., 2018a). Thus, further research is required to address the challenges in the operational use of TLS (Liang et al., 2018a).

Here we present FORTLS (Molina-Valero et al., 2021), an R package developed with the objective of automating TLS point cloud data processing and estimating variables for forestry purposes. To fulfil this objective, FORTLS enables (i) detection of trees and estimation of *dbh* and other tree attributes, (ii) estimation of some stand variables (e.g. $\overline{h}$, $G$, $V$), (iii) computation of metrics related to important tree attributes estimated in FIs at stand-level, and (iv) optimization of plot design for combining TLS data and field measured data. The package also includes several features for correcting occlusion problems to improve the estimation of stand variables. The current version of FORTLS is based on single-scan TLS data, with the aim of facilitating operational use, and it has been designed as relatively easy-to-use open-source software aimed at use by both scientists and technical users. Relative to multi-scan and multi-single-scan approaches, the single-scan approach improves data acquisition, shortens the processing time and increases the sample size in a cost-efficient manner, mainly because it does not require pre-scanning tasks involving location of artificial reference objects (Holopainen et al., 2014) or automated post-processing matching methods (Liu et al., 2017). Finally, and as a case study, we have tested FORTLS for estimating common forestry variables in an experimental plot of 1 ha located in an even-aged *Pinus sylvestris* L. stand in northern Spain. These features of the FORTLS package may enable the operational use of TLS in FIs, in combination with model-based or model-assisted inference approaches.

## 2. Methods

### 2.1. Software design

FORTLS (Molina-Valero et al., 2021) has been developed as an R package (R Core Team, 2021) because R is free statistical software which is accessible to any user interested in this tool. The initial stages of development of this package were outlined in Molina-Valero et al. (2020), although the first version of FORTLS was not available until March 2021. Currently, both the most recent stable version of the package and the most up-to-date version can be downloaded free of charge, from respectively the CRAN (https://CRAN.R-project.org/package=FORTLS) and GitHub development (https://github.com/Molina-Valero/FORTLS/tree/devel) repositories.

The R package FORTLS has been optimized by implementing C++ code in the most demanding computing processes by means of the Rcpp package (Eddelbuettel, 2013; Eddelbuettel and Balamuta, 2018; Eddelbuettel and François, 2011) and the RcppEigen package (Bates and Eddelbuettel, 2013), which enables integration of the Eigen C++ library for specific matrix calculation. For operations with objects in spatial data classes, both the raster (Hijmans, 2020) and sp (Bivand et al., 2013; Pebesma and Bivand, 2005) packages have been used. For obtaining Voronoi polygons, the ggvoronoi package (Garrett et al., 2021) has been used in the simulations and metrics.variables functions. As TLS point clouds represent large data sets, FORTLS also imports the vroom package (Hester and Wickham, 2020) for accelerating loading and saving .txt files. We have used other important packages to generate and save interactive graphics, namely plotly (Sievert, 2020) and htmlwidgets (Vaidyanathan et al., 2020). Apart from the packages included in R base distribution and other accessory packages such as progress (Csárdi and FitzJohn, 2019), scales (Wickham and Seidel, 2020) and tidyr (Wickham, 2021), the other external R packages used for more specific functions are mentioned below, with their respective functions.

The functions and results compiled in this work are based on the stable version 1.0.6 of the FORTLS package available in CRAN. In the following sections, all steps involved in TLS point cloud data processing with FORTLS, as well as the most relevant algorithms, are described: (i) normalization; (ii) tree detection; and (iii) estimation of metrics and variables at stand-level.

#### 2.1.1. Normalization

The normalization process is a necessary first step in processing point cloud data, and it is implemented in the normalize function (Table 1), which for some processes uses the functions readLAS, clip_circle, classify_ground, grid_terrain and normalize_height included in the lidR package (Roussel et al., 2020; Roussel and Auty, 2020). Normalization involves obtaining the coordinates relative to plot centre for TLS point clouds supplied as .las or .laz files. The process includes the following steps: (i) classification of points as “ground”; (ii) generation of a digital terrain model (DTM); (iii) computation of coordinates relative to DTM (Cartesian, cylindrical and spherical); and (iv) reduction of point cloud density by the point cropping process (PCP).

In the initial step, points are classified as “ground” or “not ground” with the Cloth Simulation Filter (CSF) algorithm (Zhang et al., 2016). The DTM is then generated by spatial interpolation of “ground” points. Two methods are available for executing this process: (i) spatial interpolation based on Delaunay triangulation (by default); and (ii) spatial interpolation using a k-nearest neighbour approach with inverse-distance weighting. The point cloud is then normalized by subtracting the DTM created. Once the point cloud has been normalized, Cartesian, cylindrical and spherical coordinates are calculated relative to the sampling point (TLS device establishment point). Finally, the normalize function applies the PCP algorithm developed by Molina--Valero et al. (2019) to reduce the point density and thus produce a spatially homogeneous point cloud in which the distribution of points is proportional to the object size. During execution of the PCP, a selection

**Table 1**
Main functions included in the FORTLS package.

| Function (arguments) | Description |
|---|---|
| **correlations**(simulations, variables = c ("N","G","V","d","dg","d.0","h","h.0"), method = c("pearson","spearman”), save.result = TRUE, dir.result = NULL) | Correlation between field estimates and TLS metrics |
| **distance.sampling**(tree.list.tls, id.plots = NULL, strata.attributes = NULL) | Distance sampling methods for correcting occlusion effects |
| **estimation.plot.size**(tree.list.tls, plot.parameters = list( radius.max = 25, k.tree.max = 50, BAF.max = 4), average = FALSE, all.plot.designs = FALSE) | Assessment of consistency of metrics for simulated TLS plots |
| **metrics.variables**(tree.list.tls, distance.sampling = NULL, plot.parameters, dir.data = NULL, save.result = TRUE, dir. result = NULL) | Computation of metrics and variables for TLS plots |
| **normalize**(las, x.center = NULL, y.center = NULL, max.dist = NULL, min.height = NULL, max. height = NULL, algorithm.dtm = "tin”, res.dtm = 0.2, csf = list(cloth_resolution = 0.5), id = NULL, file = NULL, dir.data = NULL, save.result = TRUE, dir. result = NULL) | Production of relative coordinates and density reduction for TLS point clouds |
| **optimize.plot.design**(correlations, variables = c ("N","G","V","d","dg","d.0","h","h.0"), dir.result = NULL) | Optimization of plot design based on optimal correlations |
| **relative.bias**(simulations, variables = c ("N","G","V","d","dg","d.0","h","h.0"), save.result = TRUE, dir.result = NULL) | Relative bias between field estimates and TLS metrics |
| **simulations**( tree.list.tls, distance.sampling = NULL, tree. list.field, plot.parameters = list( radius.max = 25, k.tree.max = 50, BAF.max = 4), dir.data = NULL, save.result = TRUE, dir. result = NULL) | Computation of metrics and variables for simulated TLS and field plots |
| **tree.detection**(data, dbh.min = 7.5, dbh.max = 200, ncr.threshold = 0.1, tls.resolution = list(), breaks = c(1.0,1.3,1.6), plot.attributes = NULL, save.result = TRUE, dir.result = NULL) | Tree detection and cross section estimation |
| **tree.detection.multiple**(las.list, id = NULL, file = NULL, normalize.arguments = list( max.dist = NULL, min.height = NULL, max. height = NULL, algorithm.dtm = "tin”, res.dtm = 0.2), tree.detection.arguments = list( dbh.min = 7.5, dbh.max = 200, ncr.threshold = 0.1, tls.resolution = list(), breaks = c(1.0,1.3,1.6), plot.attributes = NULL), dir.data = NULL, save.result = TRUE, dir. result = NULL) | Tree detection and cross section estimation for multiple plots |

probability $prob(\boldsymbol{p})$ is assigned to each cloud point $\boldsymbol{p}$ according to Eq. (1) (Fig. 1):

$$prob(\boldsymbol{p}) = \frac{r_p}{r_{max}} \tag{1}$$

where $r_p$ is the radial distance of the point $\boldsymbol{p}$ from the plot centre, and $r_{max}$ is the radial distance of the farthest point from the plot centre. Finally, the point is selected if the selection probability is equal to or higher than a random value generated from the uniform distribution for the interval [0, 1].

### 2.1.2. *Tree detection*

Tree detection represents a very important and challenging step in estimating variables of interest in TLS-assisted FIs. This is partly due to occlusions, which are much more important when working with single scans. This process of tree detection is implemented in the tree.detection function (Table 1), which has been designed to detect as many trees as possible from point cloud data obtained as normalize function output.

In the tree detection process, one or several horizontal slices from the original point cloud are extracted. Slices at heights of 1.0, 1.3 and 1.6 m ± 5 cm over the terrain (all around 1.3 m as reference section to estimate *dbh*) are considered by default in tree.detection. The probability of detecting trees increases when more than one slice is considered. However, the tree.detection function can also extract (if specified in the arguments) slices at other heights when trees are not identifiable at these pre-established values. Each horizontal slice is processed by algorithms that are able to (i) remove branches and foliage points, (ii) detect point clusters corresponding to potential tree sections, and (iii) classify detected point clusters as tree sections, or not, according to several tests. The tree sections detected for all horizontal slices are then merged, and for each tree detected the tree.detection function estimates certain attributes by: (i) calculating the coordinates corresponding to tree normal section centre (1.3 m above ground level) and its horizontal distance from plot centre; (ii) estimating the *dbh*; (iii) classifying the tree as fully visible or partially occluded; and (iv) obtaining the number of points corresponding to normal section slice (1.3 m ± 5 cm) for both original and reduced (by applying PCP) point clouds.

The main steps involved in the previously mentioned algorithms for detecting tree sections for each horizontal slice and for estimating metrics and variables related to detected trees attributes are described in the following subsections and summarized in Fig. 2.

*2.1.2.1. Removing branches and foliage points.* For each horizontal slice, this first step aims to remove points corresponding to fine branches and foliage (e.g. leaves and shrubs) and mainly to retain stem points, for which we considered local surface variation, also known as the normal change rate (NCR). This is a quantitative measure of curvature feature useful for discerning some “noisy” points (Pauly et al., 2002) with higher values representing more curved surfaces (predictably fine branches and foliage).

The NCR index is estimated at point level considering a local neighbourhood, as follows. Given a fixed radius *r*, the set of local

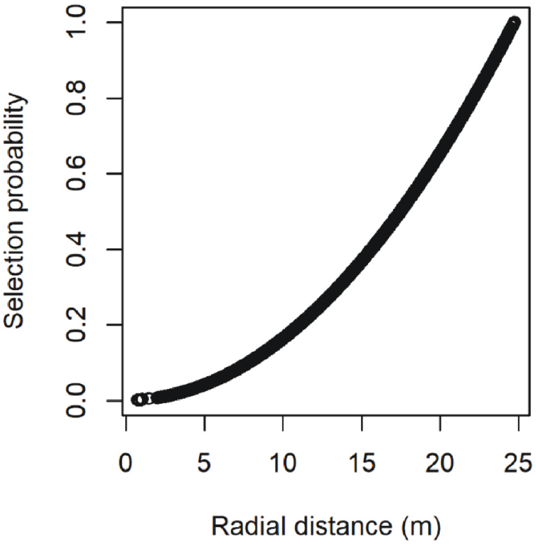


**Fig. 1.** Selection probability of points corresponding to a single-scan according to the radial distance from the plot centre, implemented in the normalize function.

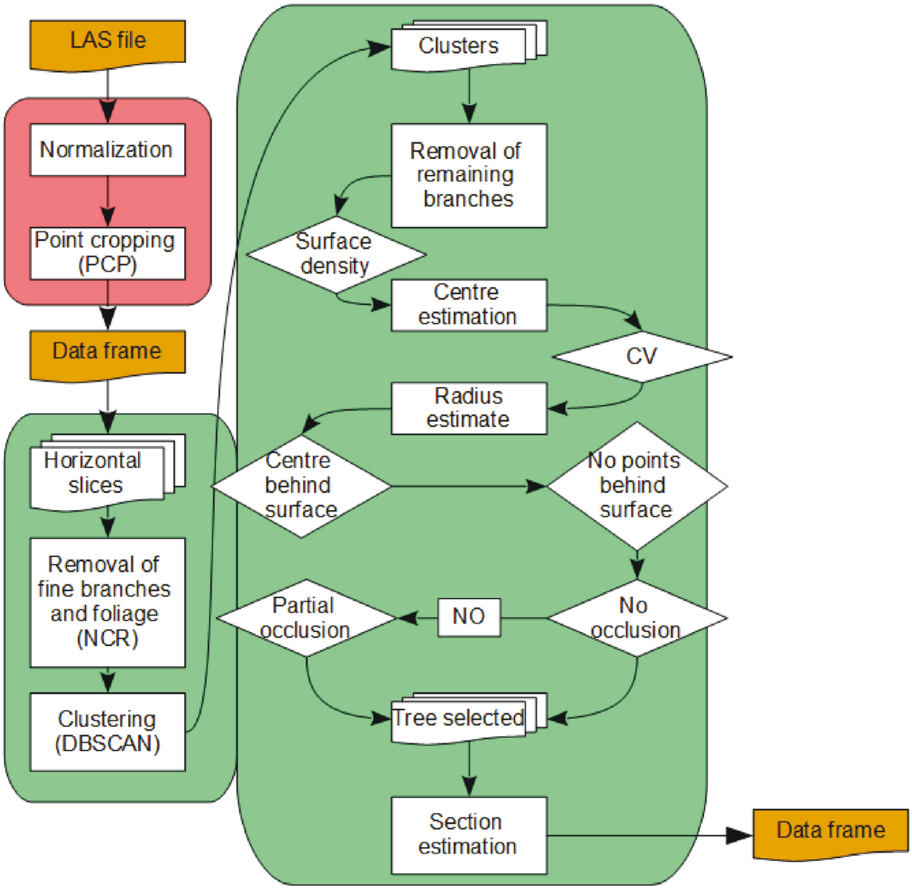


**Fig. 2.** FORTLS workflow for normalization (red) and tree detection (green) processes. All of the processes included here are described in sections 2.1.1 and 2.1.2.

neighbours of a point $\boldsymbol{p}$ is denoted by $\{\boldsymbol{p}_i\}_{i \in N_{p,r}}$, where $N_{p,r}$ is the index set of all cloud points satisfying the condition that $d(\boldsymbol{p}, \boldsymbol{p}_i) < r$ (where $d$ is the Euclidean distance). The NCR for point $\boldsymbol{p}$ is then estimated by eigenanalysis of the 3 × 3 covariance matrix $\boldsymbol{C}_p$ of its local neighbourhood (Eq. (2)):

$$\boldsymbol{C}_p = \frac{1}{N_{p,r}} \sum_{i \in N_{p,r}} (\boldsymbol{p}_i - \overline{\boldsymbol{p}})(\boldsymbol{p}_i - \overline{\boldsymbol{p}})^T \tag{2}$$

where $\overline{\boldsymbol{p}}$ is the centroid of the local neighbourhood of $\boldsymbol{p}$. Obtaining the eigenvalues $\{\lambda_i\}_{i=0}^{2}$ by singular value decomposition of $\boldsymbol{C}_p$ and assuming that $\lambda_0 \leq \lambda_1 \leq \lambda_2$, $\lambda_0$ describes the variation along the normal surface, the extent to which the points deviate from the tangent plane is estimated (Pauly et al., 2002). Hence, the NCR index for radius $r$ at point $\boldsymbol{p}$ is defined as follows (Eq. (3)):

$$NCR_r(\boldsymbol{p}) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{3}$$

Once NCR is computed, $\boldsymbol{p}$ is retained as a stem point if its NCR value is lower than a pre-established threshold (Fig. 3a). In accordance with other studies, in the tree.detection function, the neighbourhood was established by a radius of 5 cm as suitable for calculating NCR for the stem separation in forests (Ma et al., 2016; Xia et al., 2015). The threshold value of NCR used to remove branches and foliage points was established as 0.1 by default, also according to other studies in which the index has already been used with the same objective and obtaining good results (e.g. Jin et al., 2016; Zhang et al., 2019). Nevertheless, other NCR thresholds can be specified by users in the corresponding argument of the tree.detection function.

*2.1.2.2. Detection of point clusters.* Once branches and foliage points have been removed from the horizontal slice, the next step is to detect point clusters corresponding to potential tree sections. This process involves several steps: (i) a clustering process is first applied to the horizontal projection of Cartesian coordinates of points; (ii) points corresponding to possible branches are then removed using surface density approaches; and (iii) clusters receiving fewer points than expected for a full visible stem are discarded.

As mentioned above, potential tree sections are first detected through a clustering process applied to the horizontal projection of Cartesian coordinates. This clustering process is performed by the Density-Based Spatial Clustering of Applications based on the Noise (DBSCAN) method (Ester et al., 1996) and applied with the dbscan function in the R package dbscan (Hahsler et al., 2019). The size of the epsilon neighbourhood is established as the minimum distance between two consecutive points at the farthest distance from TLS in the respective horizontal slice, and a minimum of 5 points required in that epsilon neighbourhood. This algorithm detects as many as possible sections corresponding to trees (according to occlusion conditions), as well as other possible clusters generated by other items (branches, shrubs, etc.), and it has been used in previous studies of TLS with the same objective (Ferrara et al., 2018; Molina-Valero et al., 2019).

For refining extracted stem points, we used a similar approach to that proposed by Zhang et al. (2019) in order to remove any branches remaining in the clusters, based on the principle that stems should generate more points than other parts of the tree (due to the fractal-size distribution according to West et al., 1999); in addition, these points should have a predominantly vertical distribution. Thus, if the point cloud is vertically projected and rasterized in a grid, cells over stems usually include more points than those located over branches and foliage. Each cluster is thus rasterized on the horizontal plane (Cartesian coordinates) with an adapted grid step size of twice the distance

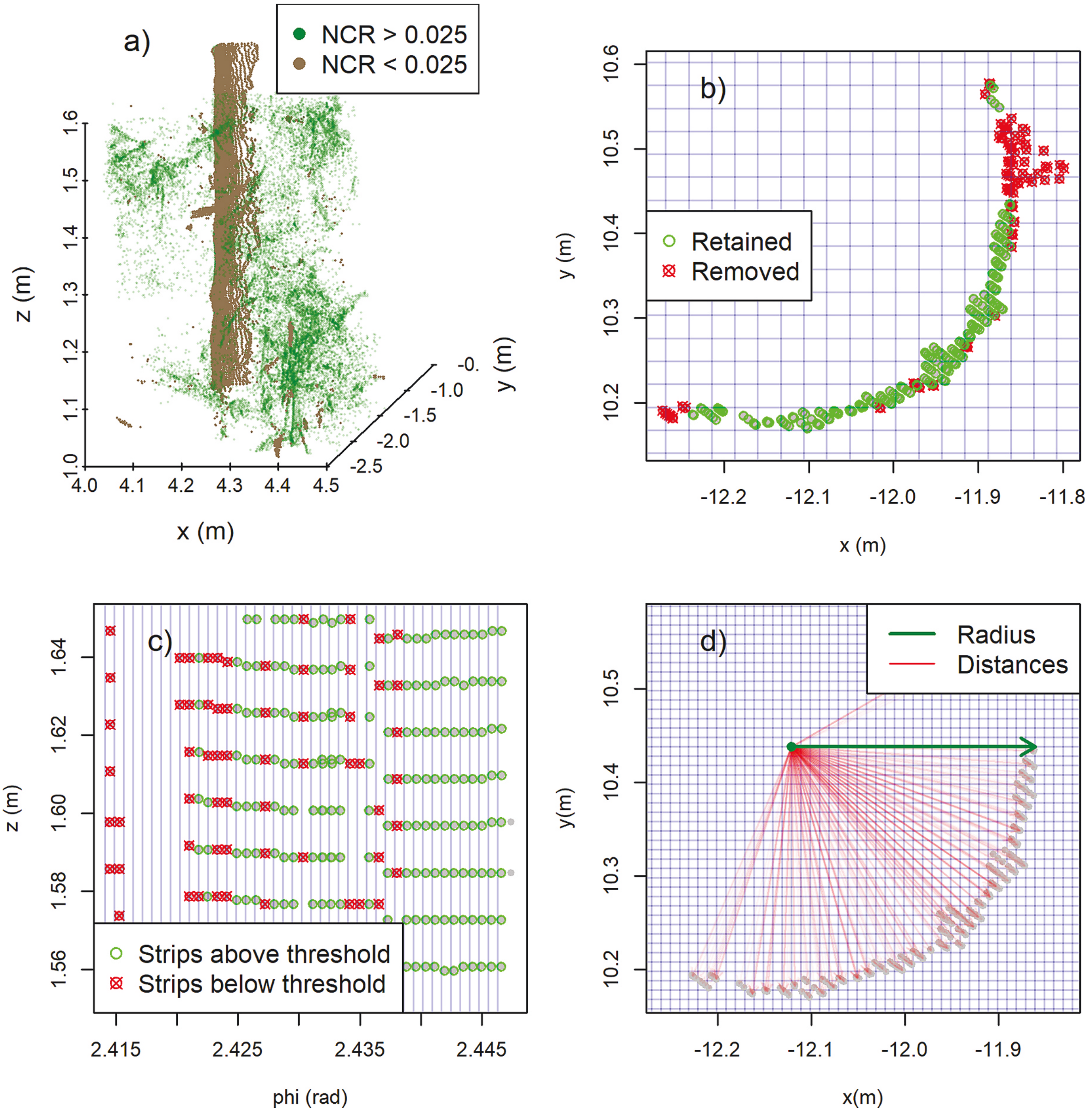


**Fig. 3.** Detection of potential tree sections implemented in the tree.detection function: a) points corresponding to branches and foliage (green) removed by applying the NCR index for a threshold of 0.025; b) refinement of extracted stem points where those points included in grid cells with fewer points than the median (red points) are removed; c) estimation of the approximate number of points (n) that each strip must contain if the section corresponds to a tree fully viewed by TLS, the green points represent those cells containing more or equal points than n; and d) estimated location of tree section centre (green point) and radius calculated (green wide arrow) as the average of all distances (red fine arrows) between cluster points and the grid intersection allocated as the tree centre.

between two consecutive points at mean cluster distance from TLS, and those points included in cells with fewer points than the median value of the number of points per cell will be removed (Fig. 3b).

At this point of the process, several tests were used to distinguish those clusters belonging to tree sections. Each cluster was first projected vertically with cylindrical coordinates ($\varphi$, $z$) and divided into regular strips bound by vertical scan resolution ($\alpha_v$) (Fig. 3c). As stems completely visibly from TLS must generate the maximum possible number of points according to scan resolution and distance from the TLS instrument, we estimated the approximate number of points that each strip must include if the section corresponds to a fully visible tree. As normalized coordinates are projected on a horizontal plane, the slope effect was first corrected to calculate the vertical resolution ($\Delta v$) in coordinate $z$ at cluster real distance from TLS (m), as follows (Eq. (4)):

$$\Delta v = 2 \times \frac{tan(\alpha_v/2)}{\bar{r}_{cluster}/\cos(\overline{slope})} \tag{4}$$

where $\alpha_v$ is the vertical scan resolution (rad), $\bar{r}_{cluster}$ is the mean radial distance in spherical coordinates from TLS to cluster (m), and $\overline{slope}$ is the mean slope of cluster according to DTM (rad). The number of points $n$ that each strip must contain in fully visibility conditions is then computed as follows (Eq. (5)):

$$n = \frac{\Delta z}{\Delta v} \tag{5}$$

where $\Delta z$ is the slice thickness (m), and $\Delta v$ is the previously defined vertical resolution (Eq. (4)). Finally, we reduced the predicted number of points by 30% in very stepped terrains (>0.5 rad). Only those clusters with at least one strip containing the number of points $n$ that every strip must contain in fully visibility conditions were selected.

Once clusters have fulfilled all of the previous checks, the next step involves obtaining the centre of the potential tree section. With this aim, regular square grids of 1 cm were overlapped on each cluster selected. The tree section centre was then considered as the intersection grid point where the variance of the distances between this intersection and all the cluster points reaches the lowest value (Fig. 3d), considered to occur when the coefficient of variation of distances is smaller than 0.1, otherwise the cluster is dismissed. Finally, the radius of the tree section was computed as the average of all the distances between the estimated centre and remaining cluster points at the moment of algorithm processing.

*2.1.2.3. Cluster classification.* This step consists of checking multiple geometrics features and indices to verify which clusters finally correspond to tree sections. Most criteria used are based on those determined by Molina-Valero et al. (2019), and they were applied to each cluster. The first test involves checking whether the centre is located behind the cluster points relative to TLS. This is considered fulfilled when at least 95% of the cluster points have a lower cylindrical coordinate $\rho$ (hence closer than plot centre) than the tree centre. The second test consists of checking for the absence of points behind the tree surface, which is considered true when at least 95% of the distances between cluster points and the tree centre are greater than half of the estimated radius. This may be visually checked by means of a distance histogram (Fig. 4a). The similarity between the cluster shape and the circumference arc is then assessed, checking that extreme points in cylindrical coordinate $\rho$ are farther from TLS than central points (Fig. 4b). This is possible when trees are largely visible from TLS, but otherwise will not be possible due to partial occlusions. In such cases, the clusters were checked to determine whether they form a smaller arc of a circle (Fig. 4c). For this purpose, we calculated the Pearson coefficient correlations for $\varphi$ values in increasing order and the correlative numbering (Fig. 4d). Those clusters with values below 0.995 were removed.

*2.1.2.4. Estimating tree attributes.* When several sections are identified at different heights, those corresponding to the same trees are joined using the DBSCAN algorithm on the horizontal projection, and some tree

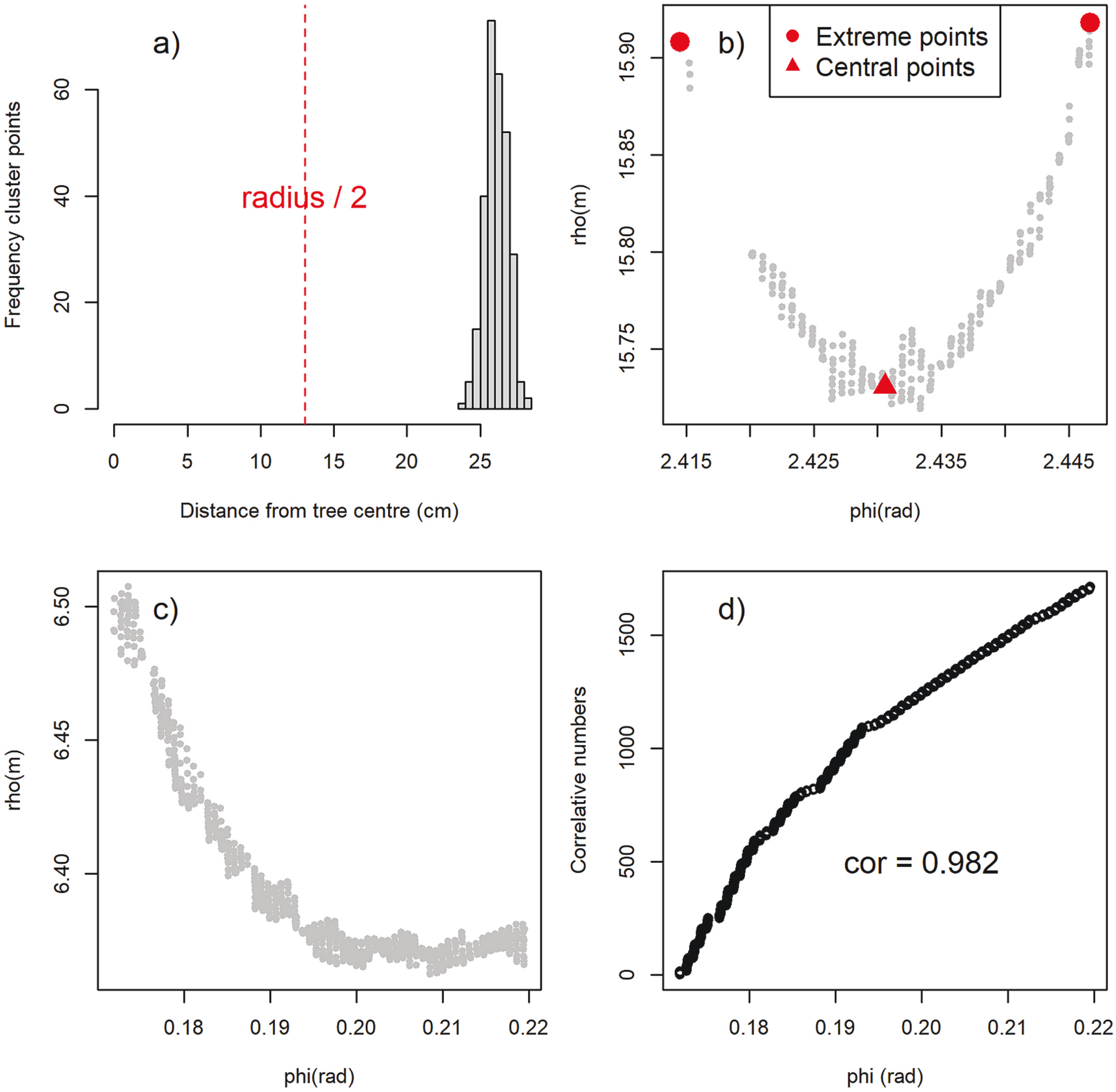


**Fig. 4.** Representation of some of the tests implemented in the tree.detection function to distinguish those clusters corresponding to tree sections: a) histogram of distances between cluster points and estimated tree centre to check for the absence of points behind tree surface; b) mean coordinates of points included in first and last $\varphi$ coordinate percentiles (red points) and in the middle cluster $\varphi$ ± TLS angle aperture (red triangle); c) tree partially occluded; and d) assessment of the Pearson's correlation between $\varphi$ values in increasing order and their corresponding correlative numbers.

attributes are obtained: coordinates of normal section centre and horizontal distance from plot centre, estimated *dbh*, indicator of partial occlusion and number of points corresponding to normal section for original and reduced point clouds.

When trees are exclusively detected at 1.3 m, the *dbh* is estimated directly as twice the radius estimated at this height. Conversely, when trees are detected from other section(s) (including 1.3 m or not), a linear taper equation is fitted with radius as the response variable *radius* and section height ($h_{sec}h_{sec}$) as the explanatory variable (Eq. (6)):

$$radius = \beta_0 + \beta_1 \cdot h_{sec} \tag{6}$$

The radius at 1.3 m is then predicted, as follows (Eq. (7)):

$$\widehat{radius}_{1.3} = radius_i + \widehat{\beta}_1 \cdot (1.3 - h_{sec_i}) \tag{7}$$

where $radius_i$ and $h_{sec_i}$ are the estimated radius and the height corresponding to section $i$, and $\widehat{\beta}_1$ is the slope parameter fitted in the linear regression (Eq. (6)). Hence, *dbh* is computed as twice the averaged predicted radius.

Finally, the number of points (*num.points*) corresponding to a normal section (+/- 5 cm) in the original point cloud is computed and also estimated for each detected tree *j* by using the *dbh* values previously computed as follows (Eq. (8)):

$$num.points.est_j = dbh_j \frac{\sum_{i \in I} \frac{num.points_i}{dbh_i}}{\#I} \tag{8}$$

where the index set $I$ corresponds to all detected trees fully visible at 1.3 m, and $\#I$ is the number of trees fully visible at 1.3 m. The number of points and the estimated number of points for the point cloud reduced by PCP are obtained in a similar way.

### 2.1.3. *Computing TLS metrics and variables at stand level*

Once normalization and tree detection processes for each point cloud are completed, TLS metrics and variables can be estimated at stand level for three different plot designs, all of which are included in the metrics. variables function (Table 1). These plot designs are circular fixed area, k-tree and angle-count (Bitterlich, 1948) plots (Fig. 5), each of which is defined by a unique design parameter (radius, k and basal area factor (BAF), respectively) that must be specified in the function arguments.

The metrics and variables computed for each plot design are summarized in Table 2, and more details are compiled below. The two approaches for optimizing plot design by means of FORTLS functions are also described.

*2.1.3.1. Stand-level metrics.* Metrics are computed using points directly from normalized point cloud in a relatively similar way as in the FUSION/LDV software for LiDAR data analysis and visualization (McGaughey, 2009). These are statistical descriptive measures such as percentiles or only number of points belonging to specific sections of point cloud. The following metrics are available:

- Total number of points corresponding to the normal section (+/- 5 cm) of trees detected after removing some noisy points with NCR and refinement of extracted stem points processes (see section 2.1.2). These can be computed for raw point clouds (num.points) and reduced point clouds (num.points.hom).
- Number of estimated points corresponding to the normal section (+/- 5 cm) of trees detected. The number of points corresponding to each tree can be estimated according to Eq. (8) for raw point clouds (num.points.est) and, analogously, for reduced point clouds (num.points.hom.est).
- Percentiles of z coordinate (m). Computed percentiles are P01, P05, P10, P20, P25, P30, P40, P50, P60, P70, P75, P80, P90, P95 and P99.

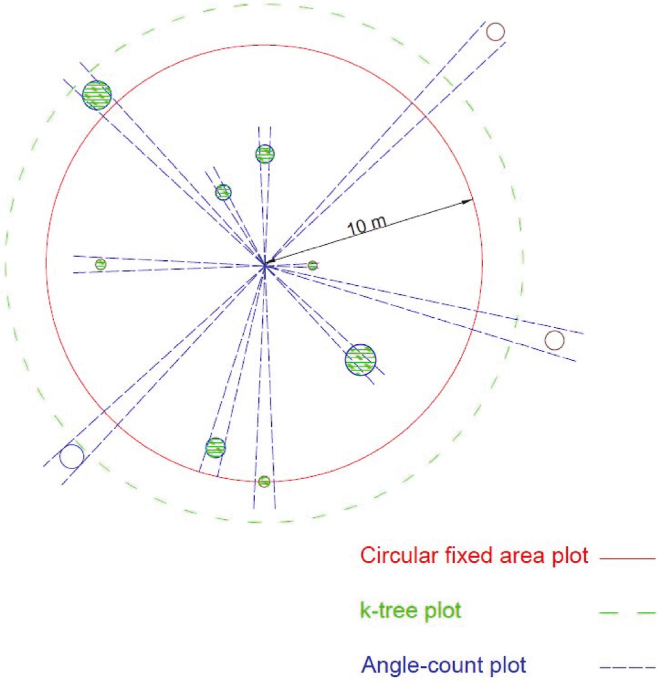


**Fig. 5.** Example of the three plot designs considered in FORTLS for the same sampling point. Circular fixed area plot of 10 m radius (red) includes 7 trees; k-tree plot for k = 8 (green) yields a larger radius than circular fixed area; and angle-count plot (blue) with a particular BAF includes 7 trees.

- Descriptive statistics of z coordinate distribution: mean, maximum (max), minimum (min), standard deviation (sd), variance (var), mode, kurtosis and skewness.
- Percentage of points above mode (perc_on_mode) and mean (perc_on_mean) values of z coordinates.
- Scale (weibull_b) and shape (weibull_c) parameters of a Weibull distribution fitted to z coordinates distribution.

*2.1.3.2. Stand-level variables.* Variables represent estimates based on the attributes of trees detected from TLS point cloud data, further aggregated at stand-level, and finally expanded to unit area (ha). The following variables are available:

- Apparent stand density (N.tls, trees ha$^{-1}$), which is estimated for trees detected in a similar procedure to that used in conventional inventories for circular fixed area and k-tree plots (Eq. (9)) and angle-count plots (Eq. (10)):

$$N.tls = \frac{10000}{\pi R^2} \cdot n \tag{9}$$

$$N.tls = \sum_{i=1}^{n} \frac{\text{BAF}}{g_i} \tag{10}$$

where $R$ is the plot radius (m), $n$ is the number of trees detected in the corresponding plot design, BAF is the basal area factor (m$^2$ ha$^{-1}$), and $g_i$ is the basal area of the tree $i$ (m$^2$).

- Apparent stand basal area (G.tls, m$^2$ ha$^{-1}$), which is estimated for trees detected in a similar procedure as that used in conventional inventories for circular fixed area and k-tree plots (Eq. (11)) and angle-count plots (Eq. (12)):

$$G.tls = \frac{10000}{\pi R^2} \sum_{i=1}^{n} g_i \tag{11}$$

**Table 2**
FORTLS metrics and variables.

| Metrics and variables | Description |
|---|---|
| N[a]/N.tls | Stand density ($N$, trees ha$^{-1}$) |
| N.hn[b], N.hr[b], N.hn.cov[b], N.hr.cov[b] | Stand density ($N$, trees ha$^{-1}$) with occlusion corrections based on distance sampling methodologies |
| N.sh[b] | Stand density ($N$, trees ha$^{-1}$) with correction of the shadowing effect |
| N.pam[c] | Stand density ($N$, trees ha$^{-1}$) with occlusion correction based on a Poisson attenuation model |
| G[a]/G.tls | Stand basal area ($G$, m$^2$ ha$^{-1}$) |
| G.hn[b], G.hr[b], G.hn.cov[b], G.hr.cov[b] | Stand basal area ($G$, m$^2$ ha$^{-1}$) with occlusion corrections based on distance sampling methodologies |
| G.sh[b] | Stand basal area ($G$, m$^2$ ha$^{-1}$) with correction of the shadowing effect |
| G.pam[c] | Stand basal area ($G$, m$^2$ ha$^{-1}$) with occlusion correction based on a Poisson attenuation model |
| V[a]/V.tls | Stand volume ($V$, m$^3$ ha$^{-1}$) |
| V.hn[b], V.hr[b], V.hn.cov[b], V.hr.cov[b] | Stand volume ($V$, m$^3$ ha$^{-1}$) with occlusion corrections based on distance sampling methodologies |
| V.sh[b] | Stand volume ($V$, m$^3$ ha$^{-1}$) with correction of the shadowing effect |
| V.pam[c] | Stand volume ($V$, m$^3$ ha$^{-1}$) with occlusion correction based on a Poisson attenuation model |
| num.points, num.points.est, num.points.hom, num.points.hom.est | Number of points and estimated number of points corresponding to normal sections (+/- 5 cm) of trees in the original point cloud (num.points and num.points.est, respectively); and number of points and estimated number of points (points) corresponding to normal sections (+/- 5 cm) of trees in the reduced point cloud (num.points.hom and num.points.hom.est, respectively). |
| d[a], dg[a], dgeom[a], dharm[a]/d.tls, dg.tls, dgeom.tls, dharm.tls | Stand mean *dbh*, using arithmetic mean ($\overline{d}$, cm), quadratic mean ($d_g$, cm), geometric mean (cm), and harmonic mean (cm), respectively. |
| h[a], hg[a], hgeom[a], hharm[a]/h.tls, hg.tls, hgeom.tls, hharm.tls | Stand mean *h*, using arithmetic mean ($\overline{h}$, m), quadratic mean (m), geometric mean (m), and harmonic mean (m), respectively. |
| d.0[a], dg.0[a], dgeom.0[a], dharm.0[a]/d.0.tls, dg.0.tls, dgeom.0.tls, dharm.0.tls | Stand dominant mean *dbh*, using arithmetic mean ($d_0$, cm), quadratic mean (cm), geometric mean (cm), and harmonic mean (cm), respectively. |
| h.0[a], hg.0[a], hgeom.0[a], hharm.0[a]/h.0.tls, hg.0.tls, hgeom.0.tls, hharm.0.tls | Stand dominant mean *h*, using arithmetic mean ($H_0$, m), quadratic mean (m), geometric mean (m), and harmonic mean (m), respectively. |
| P01, P05, P10, P20, P25, P30, P40, P50, P60, P70, P75, P80, P90, P95, P99 | Height percentiles (m) derived from z coordinates of TLS point clouds relative to ground level. |
| mean[d], max[d], min[d], sd[d], var[d], mode[d], kurtosis[d], skewness[d] | Descriptive statistics of z coordinates distribution of TLS point clouds relative to ground level: mean, maximum, minimum, standard deviation, variance, mode, kurtosis and skewness, respectively. |
| perc.on.mode[d], perc.on.mean[d] | Percentage of points above mode and mean values of z coordinates of TLS point clouds relative to ground level. |
| weibull_b[d], weibull_c[d] | Scale and shape parameters, respectively, for Weibull distribution fitted for z coordinates of TLS point clouds relative to ground level. |

[a] Variables estimated from field measurement data.
[b] Variables only estimated for circular fixed area and k-tree plot design.
[c] Variables only estimated for angle-count plot design.
[d] Variables not included in FORTLS example data because they were included in the previous version (1.0.3).

$$G.tls = \mathrm{BAF} \cdot n \tag{12}$$

- Apparent stand stem volume (V.tls, m$^3$ ha$^{-1}$), which is estimated for trees detected by modelling stem profile as a paraboloid and calculating the volumes of revolution for fixed area and k-tree plots (Eq. (13)) and angle-count plots (Eq. (14)):

$$V.tls = \frac{10000}{\pi R^2} \sum_{i=1}^{n} \pi \cdot \frac{h^2_{P_{99}} i}{2} \cdot \frac{\left(dbh_i/2\right)^2}{\left(h_{P_{99_i}} - 1.3\right)^2} \tag{13}$$

$$V.tls = \sum_{i=1}^{n} \frac{\mathrm{BAF}}{g_i} \cdot \pi \cdot \frac{h^2_{P_{99}} i}{2} \cdot \frac{\left(dbh_i/2\right)^2}{\left(h_{P_{99_i}} - 1.3\right)^2} \tag{14}$$

where $h_{P_{99}\,i}$ and $dbh_i$ are the 99th percentile of points delimited by Voronoi polygons (m) (i.e. estimates of $h$) and $dbh$ (m) for tree $i$, respectively.

- Mean and dominant diameters (cm), which are estimated for arithmetic, quadratic, geometric and harmonic means. In the case of dominant diameters, only the *n* largest trees per ha (according to *dbh*) are considered. Although it can be specified in the arguments, the 100 largest trees per ha are considered by default.
- Mean and dominant heights (m), which are estimated for arithmetic, quadratic, geometric and harmonic means. In the case of dominant heights, only the *n* largest trees per ha (according to *dbh*) are considered. Although the number of trees can be specified in the arguments, the 100 largest trees per ha are considered by default.

In the previous calculations, for the k-tree design, the plot radius *R* is defined as the mean of horizontal distances of trees k and k+1 (Kleinn and Vilčko, 2006).

#### 2.1.3.3. *Dealing with occlusions.*

All FORTLS functions used for estimating stand variables also include correction of occlusions approaches. In the case of angle-count plots, occlusion corrections are based on gap probability attenuation with distance from TLS depending on a Poisson distribution. In the case of circular fixed area and k-tree plots, distance sampling methods and shadowing effect correction are considered. In order to obtain occlusion corrections based on distance sampling methods, the distance.sampling function must be executed previously and the values obtained must be incorporated as an argument for functions which compute stand variables, whereas the other corrections are computed by default. A brief description of the implemented occlusion corrections is given below.

#### 2.1.3.4. *Poisson attenuation model.*

This method has been used in measurements with TLS (Strahler et al., 2008; Lovell et al., 2011) and optical (Montes et al., 2019) instruments to reduce the device-related bias in the relascope-based approach. It is based on geometric gap probability ($P_{gap}$), which decreases exponentially following a Poisson distribution (Eq. (15)):

$$P_{gap}(\lambda, D_E, R) = e^{(-\lambda D_E R)} \tag{15}$$

where $\lambda$ is the number of trees per m$^2$, $D_E$ is the effective *dbh* and *R* is the horizontal distance to the farthest tree. The corrected stand density (N.pam, trees ha$^{-1}$) for angle-count plots is then obtained as follows (Eq. (16)):

$$N.pam = N.tls \,/\, F(\lambda D_E R) \tag{16}$$

where $F$ is a function defined as $F(t) = \frac{2}{t^2}(1 - e^{-t}(1+t))$, with $t = \frac{\lambda D_E D}{2\sqrt{\mathrm{BAF}}}$: see Strahler et al. (2008) and Lovell et al. (2011) for further details. Similarly, corrected stand basal area (G.pam, m$^2$ ha$^{-1}$) and volume (V.pam, m$^3$ ha$^{-1}$) are computed.

*2.1.3.5. Point transect sampling.* This approach is based on the *point transects* method from distance sampling methods (Buckland et al., 2001). These methods use detection functions $g(r,\theta)$ with variable $r$ (distance from sampling point) and parameter $\theta$, which describe how the probability of detection decreases as distance increases. We used Half-Normal (Eq. (17)) and Hazard-Rate (Eq. (18)) functions as these have been successfully used in measurements with TLS (Astrup et al., 2014) and optical (Montes et al., 2019) devices:

$$g(r,\theta) = e^{\left(\frac{-r^2}{2\sigma^2}\right)} \tag{17}$$

$$g(r,\theta) = 1 - e^{-\left(\frac{r}{\sigma}\right)^{-b}} \tag{18}$$

Parameter $\theta$ includes the shape ($b$) (only in the Hazard-Rate function) and the scale ($\sigma$), which was also expanded with *dbh* as a covariate into an exponential function (Eq. (19)) according to Ducey and Astrup (2013) and Astrup et al. (2014):

$$\sigma = \alpha_0 e^{(\alpha_1 dbh)} \tag{19}$$

Parameter $\theta$ is estimated by maximum likelihood (Marques and Buckland, 2003; Miller and Thomas, 2015; Clark, 2016) with data left-truncated at 1 m according to Astrup et al. (2014). The fitting process is carried out by means of the ds function included in the R package Distance (Miller et al., 2019). Once the parameters of detection functions are estimated, the probability of tree detection ($P_i$) is estimated with Eq. (20), which is implemented in the expansion factor of circular fixed area and k-tree plots as in Eq. (21):

$$P_i = \frac{2}{R^2}\int_0^R rg(r,\widehat{\theta}) \tag{20}$$

$$EF = \sum_{i=1}^{n} \frac{10000}{(P_i \pi R^2)} \tag{21}$$

where $R$ is the plot radius (m), and *EF* is the expansion factor. Multiplying N.tls by these *EFs* yields corrected stand densities (N.hn, N.hr, N.hn.cov, N.hr.cov, trees ha$^{-1}$). Similarly, corrected stand basal area (G.hn, G.hr, G.hn.cov, G.hr.cov, m$^2$ ha$^{-1}$) and volume (V.hn, V.hr, V.hn.cov, V.hr.cov, m$^3$ ha$^{-1}$) are computed.

All of these estimates are obtained by previously executing the distance.sampling function (Table 1), which returns $P_i$ and also values of parameter estimates of detection functions and the corresponding Akaike information criterion (AIC) of these fits.

*2.1.3.6. Correcting the shadowing effect.* This approach was developed by Seidel and Ammer (2014) for single-scan mode. These authors used the approach to correct the shadowing effect, which generates shaded unsampled areas (Eq. (22)), according to the shaded area percentage related to the total area sampled:

$$A_{shadow} = \left[\frac{(\pi R^2) - (\pi r_{tree}^2)}{360^\circ}\left(\frac{dbh}{r_{tree}}\right)\right] - \left[\frac{\pi\left(dbh/2\right)^2}{2}\right] \tag{22}$$

where $R$ is the radius of the plot (m), and $r_{tree}$ is the distance between the TLS instrument and the tree centre. This method is implemented for circular fixed area and k-tree plots and yields an expansion factor used to compute corrected estimates for stand density (N.sh, trees ha$^{-1}$), basal area (G.sh, m$^2$ ha$^{-1}$) and volume (V.sh, m$^3$ ha$^{-1}$).

*2.1.3.7. Optimizing the plot design.* Two different approaches can be used to find the best possible plot design depending on whether validation field data are available or not. The approaches are represented in the FORTLS workflow (Fig. 6) and detailed below.

*2.1.3.8. Analysis of estimation stability.* The function estimation.plot.size estimates both apparent tree density (N.tls, trees ha$^{-1}$) and apparent basal area (G.tls, m$^2$ ha$^{-1}$) for all of the aforementioned plot designs. In the case of circular fixed area plot design, concentric plots, in regular increments of 0.1 m radius (by default) to the maximum radius specified in the arguments, are simulated for computing N.tls and G.tls. As a result, line charts with estimates through plot size are obtained. For k-tree design, all possible plots are defined by $k = \{1, 2, \ldots, n\}$, where 1 is the nearest tree and $n$ the farthest tree considered in the argument k.tree.max (or the farthest detected/existing tree if the argument is not specified). Finally, for the angle-count design, variables will be estimated for regular BAF increments comprised from 0.1 to the BAF.max specified in the arguments. All of these line charts were inspired by Fig. 3 in Brunner and Gizachew (2014).

*2.1.3.9. Validation with field measurements.* For cases when field data are available, we designed a set of interconnected functions able to assess the performance of processed TLS data relative to the corresponding field data: simulations, relative.bias and correlations.

The field data necessary to conduct the analysis described hereinafter are tree dimensions (*dbh* and *h*) and positions relative to the TLS scanner. Analysis of the performance is based on comparisons between these two data sources for the different plot designs and sizes. The first function is simulations, which computes (in a similar way as estimation.plot.size) all of the metrics and variables aforementioned for TLS data 2.1.3and the corresponding variables based on field data (Table 2).

The relative.bias function was designed for direct comparison of TLS-based estimates and field based measurements, by means of relative bias (Eq. (23)).

$$Relative\ bias = \frac{\frac{1}{n}\sum_{i=1}^{n} y_i - \frac{1}{n}\sum_{i=1}^{n} x_i}{\frac{1}{n}\sum_{i=1}^{n} x_i} \tag{23}$$

where $x_i$ and $y_i$ are the values of the field estimate and its TLS counterpart, respectively, corresponding to plot $i$ for $i = 1, \ldots, n$. Relative bias is assessed for all the simulations to find the best possible plot design for each variable of interest.

For other possible approaches apart from direct variables estimations, the package has other functions that assess the best possible plot designs according to the correlations between variable estimates from field data and metrics/variables derived from TLS. The correlations function computes both Pearson and Spearman correlation coefficients for common set of plots and all simulations and plot designs considered. For each variable of interest, this function produces the optimum correlations for all simulations. The optimize.plot.design function then produces a graphical representation of the strongest correlations for all variables of interest.

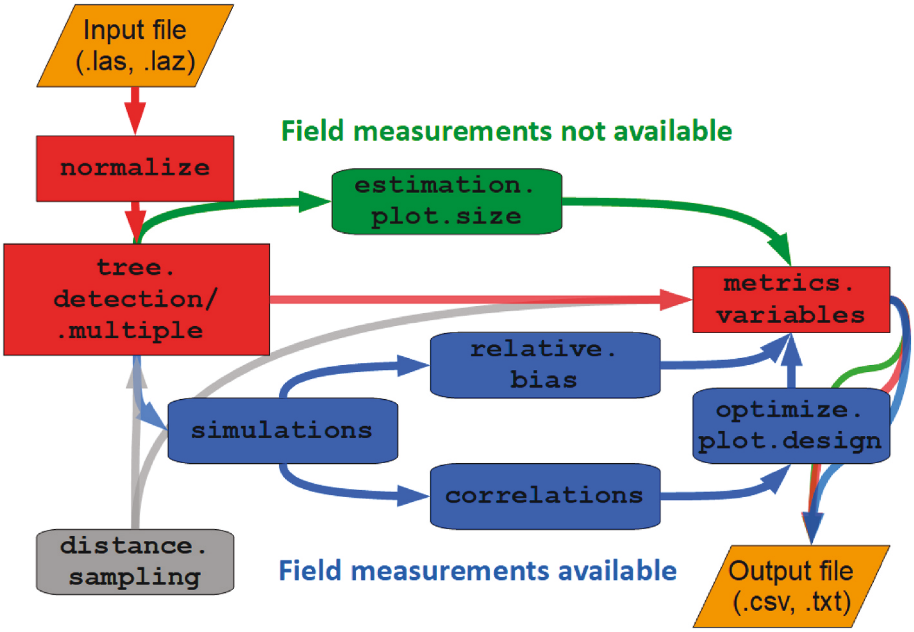


**Fig. 6.** Workflow of FORTLS, representing (i) tree detection (red), estimation of variables related to forest attributes (ii.a) with no field data available (green) and (ii.b) with field data available (blue); optional functions (grey), and initial input files and final output files (orange).

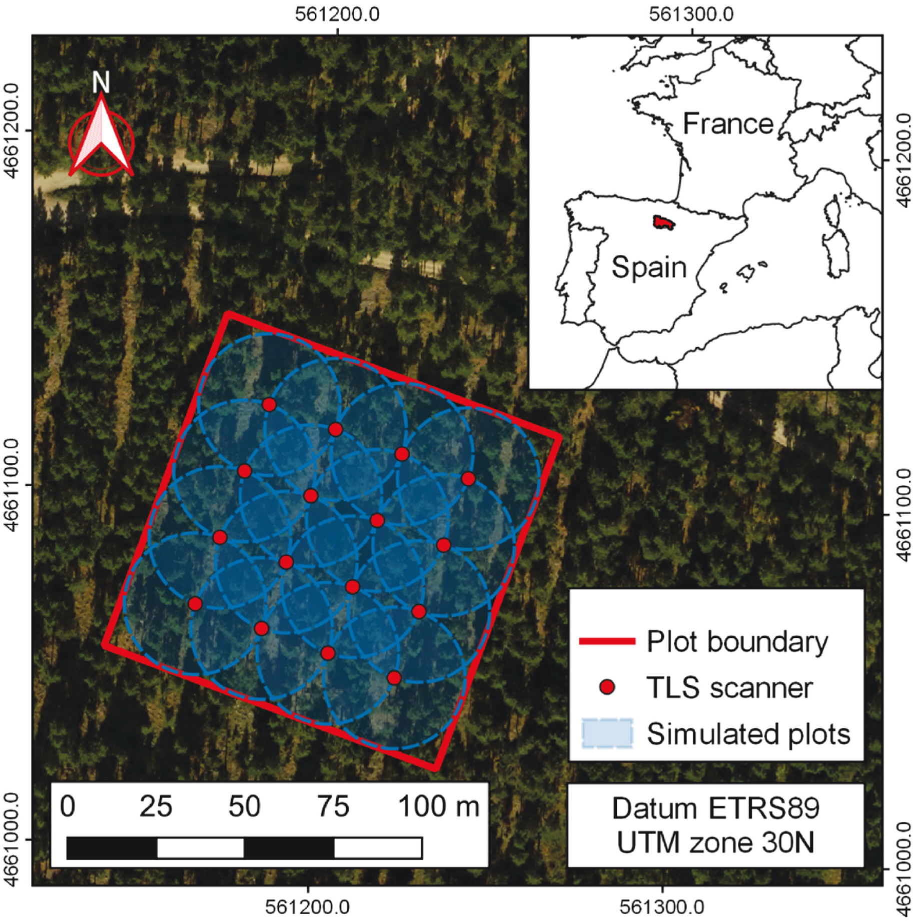


**Fig. 7.** Experimental sample plot used for software testing. The plot, of area 1 ha (100 × 100 m), is located in a pure, even-aged *Pinus sylvestris* stand located in La Rioja (Spain). Blue circles represent the 16 simulated subplots for both field data and processed TLS data.

**Table 3**
Main stand variables in the study plot.

| Variable | Stand value |
|---|---|
| $N$ live trees (trees ha$^{-1}$) | 322.00 |
| $N$ dead trees (trees ha$^{-1}$) | 4.00 |
| $G$ live trees (m$^2$ ha$^{-1}$) | 25.29 |
| $\bar{d}$ live trees (cm) | 31.11 |
| $d_g$ live trees (cm) | 31.62 |
| $\bar{h}$ live trees (m) | 15.61 |
| $d_0$ live trees (cm) | 36.69 |
| $H_0$ live trees (m) | 16.42 |

### 2.2. Case study

The functionality of the FORTLS package was tested in a fully mapped case study plot of 1 ha (100 × 100 m) in a pure, even-aged *P. sylvestris* stand located in La Rioja (Spain) (Fig. 7). All trees in the plot (live and standing dead) with *dbh* greater than 7.5 cm were measured. UTM coordinates and elevation of the plot corners were measured with a high accuracy GNSS receiver (Trimble R2), and trees were located in the plot with a total station Nikon DTM-332. The *dbh* of all trees was measured with a diameter tape to the nearest 0.1 cm, and for live trees only, *h* was measured with a digital hypsometer (Vertex IV, Haglöf Sweden) to the nearest 0.1 m. Main stand variables estimated from field data are shown in Table 3. After conventional inventory, we scanned each intersection point defined by a regular 20 m squared grid (16 points sampling) with a TLS *FARO Laser Scanner Focus$^{3D}$ X 130*, covering the full horizontal (0-360°) and vertical ranges (−60-90°) with a resolution of 7.67 mm at 10 m in both, horizontal and vertical angular apertures. All of the single scans were then processed with FORTLS, clipping the point clouds at 20 m radius from the plot centre. Finally, to compare stand variables derived from TLS data and from field inventory, we also extracted the trees measured at the same 16 points sampling scanned considering 20 m radius. All of these data can be found as example data in FORTLS as a list named "Rioja.data", in which the first element corresponds to the list of trees detected from TLS in each plot ("tree.list.tls"), and the second element corresponds to the list of trees measured in each plot in the field ("tree.list.field").

## 3. Results

The steps involved in processing the TLS data with FORTLS are described below, along with the different operations and arguments of the functions. Finally, the results of TLS data processing with FORTLS are presented for the aforementioned case study.

### 3.1. FORTLS implementation

Users can install the released version of FORTLS from CRAN (https://CRAN.R-project.org/package=FORTLS) or they can install the most currently developed version from Github (https://github.com/Molina-Valero/FORTLS/tree/devel) using the install_github function of the devtools package (Wickham et al., 2021). A list with a brief description of the 10 main functions available in FORTLS 1.0.6, together with their default argument values, is given in Table 1. The functions were designed to obtain two main uses from TLS data: (1) conservative tree detection (in which the algorithm gives preference to accuracy), and (2) computation of metrics and variables related to forest attributes (2.1) with no field data available and (2.2) with the corresponding field data available. The workflow of functions involved in these approaches is illustrated schematically in Fig. 6.

Although the workflow can be processed directly in the current R session, most functions are designed to import data and save results to a specific working directory, that should be specified in dir.data and dir.result arguments to facilitate efficient operation. In this respect, it is important to highlight that outputs from previous functions in the workflow (Fig. 6) are usually inputs in the following functions. Thus, systematization of the work will be enhanced when the data working directory corresponds to the results working directory. Script with a full example of workflow is supplied as supplementary material (S1), which should be consulted while reading the following sections.

#### 3.1.1. Tree detection

This approach is encompassed by point cloud normalization (section 2.1.1) and tree detection processes (section 2.1.2), which are executed by the normalize and tree.detection (or tree.detection.multiple for the cases with several scans) functions respectively.

TLS data must be supplied as .las or .laz files in normalize. Note that point cloud centre Cartesian ($x$, $y$) coordinates (relative to TLS sampling point) can be defined, otherwise the function will use the file centre coordinates as default. The maximum radial distance from the plot centre, and the minimum and maximum heights of the z coordinate can also be defined in the arguments. This enables possible outliers and unnecessary information to be discarded and decreases the computing time. The normalize function will return a data frame with a normalized point cloud, which will be saved in dir.result as a .txt file (otherwise specified save.result = FALSE) but with point density reduced by applying PCP.

This normalized point cloud is necessary as data input in tree.detection. Note that supplying the original (not the reduced) point cloud is highly recommended for detection of a greater number of trees. Parameters of TLS resolution are necessary in the tree.detection function, defined according to either angle resolution (rad) or distance between two consecutive points (mm) at a determinate distance from TLS (m). Some inventory parameters such as minimum and maximum *dbh* can then be specified in the arguments to prevent detection of smaller and/or larger desirable trees. Other parameters of the algorithm, such as NCR (see section 2.1.2) and plot level attributes, can be specified in the respective arguments (ncr.threshold and plot.attributes respectively). This function will return a data frame with the tree centre location in Cartesian ($x$, $y$) and cylindrical ($\rho$ (as horizontal distance), $\varphi$) coordinates, number of points corresponding to normal section (raw and

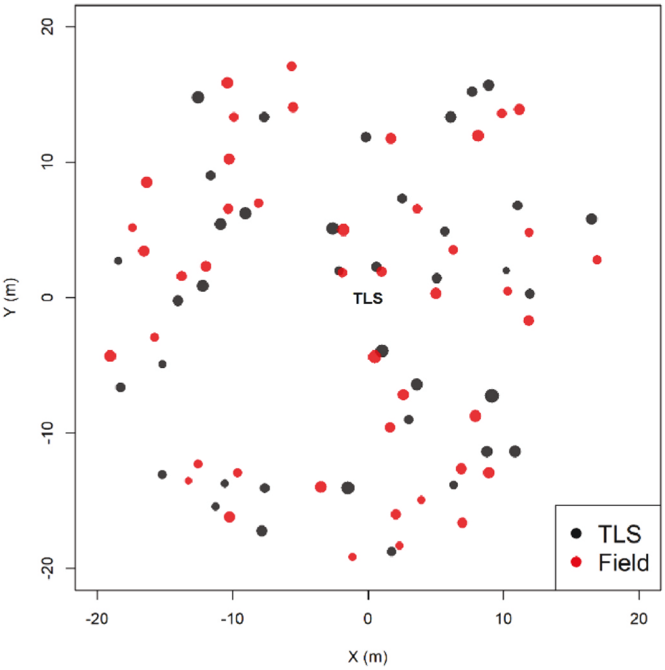


**Fig. 8.** Trees detected from point cloud data with the tree.detection function (black) and trees measured in the field (red). Tree sizes are proportional to the estimated/measured *dbh*.

estimated) and partial occlusion, for all detected trees. The data frame will be saved as a .csv file in the working directory provided in the dir. result argument (otherwise specified save.result = FALSE). In the first plot of the example (top left in Fig. 7) and considering a radius of 20 m, 38 out of 44 trees were detected. However, some angular deviation due to the internal compass of TLS can be observed (Fig. 8). Considering all of the simulated plots, our algorithm detected 598 out 659 trees, i.e. 91% of detected trees.

When sequential analysis of several scans is required, tree.detection. multiple will be the most appropriate function. This function includes both normalize and tree detection processes, enabling detection of trees for a set of plots in a single function. These must be included in the same working directory (specified in dir.data) as .las or .laz files. The output will be a similar data frame as the aforementioned data frame but that includes all of the plots processed. As before, a .csv file containing the data frame information will be saved by default in the working directory specified in dir.result, and the normalized point clouds reduced by PCP will be saved as .txt files. An example of a reduced point cloud, named "pcd", corresponding to the first plot in the case study can be found in the supplementary material (S5) as a .txt file. All trees detected with tree.detection.multiple for the 16 TLS sampling points (up to 20 m radius) in the case study (Fig. 7) and the list of trees measured in the field for these sampling points are included in supplementary material (S5) as .csv files and named "tree.list.tls" and "tree.list.field" respectively.

### 3.1.2. Estimating metrics and variables related to forest attributes

The metrics.variables function performs this process by computing a set of TLS metrics and variables from point cloud data. However, other complementary functions play an important role in improving these estimates based on correction of the occlusion effect and choice of a more appropriate plot design. In the first case, distance sampling approaches based on point transect sampling methods (Buckland et al., 2001) can be applied by means of the distance.sampling function. These approaches are used for the list of trees detected, which must be provided to the function as a data frame with the same format and include the tree.detection value. The function works for all the plot designs included by default (if no arguments are specified), but can also only consider a set of specified plots (argument id.plots must contain this information), or even plots distinguished by strata (reachable by using the argument strata.attributes). The function will generate a list with 3 elements: (i) probability of tree detection for the different detection functions and strata (if specified); (ii) parameters estimated for detection functions; and (iii) AIC estimator obtained for each detection probability function as the criteria for selecting the best fit. Regarding the second case, i.e. the optimization of plot design, more details can be found below.

At this stage, the input data sets for metrics.variables will be as follows: (i) list of trees detected; (ii) reduced normalized point clouds (loaded from dir.data); and optionally (iii) the list with probabilities of tree detection according to distance sampling methods. Plot parameters considered for estimating metrics and variables are implemented in the plot.parameters argument. This is a data frame containing the columns named radius, k.tree and BAF in reference to circular fixed area, k-tree and angle-count plot design parameters. The absence of parameter values rules out the corresponding plot design. It is also possible to include a column named num.trees that specifies the number of dominant trees per ha considered to estimate dominant *dbh* and *h* (otherwise it will be considered as 100 largest trees ha$^{-1}$ in terms of *dbh*). Any plots that are grouped by strata in the tree list input can be identified in a column named stratum to consider different plot design parameters for each stratum. In this case, another column, also named stratum and coinciding with its homologue from the tree list data in the strata coding, can be included in the plot.parameters argument. After execution of metrics.variables, a list with as many elements as plot designs considered will be returned. The elements will include the TLS metrics and variables computed for each plot design indicated. By default, this list will be saved as separate .csv files for each specified plot design in dir. result (otherwise specified save.result = FALSE). An example corresponding to circular fixed area, k-tree and angle-count plots of 15 m radius, 12 trees and BAF = 1 respectively, from the case study, can be found in supplementary material (S5) as .csv files named "metrics.variables.fixed.area.plot", "metrics.variables.k.tree.plot" and "metrics.variables.angle.count.plot".

#### 3.1.2.1. Field data not available.

When field data are not available, it is not possible to determine how well the TLS metrics and variables perform in estimating the corresponding variables. In this case, the most reliable estimates can be obtained with those plot designs in which estimates are stable for different plot design sizes. The estimation.plot.size function can be used to assess whether the plot designs considered will reach stable values and can thus use the corresponding parameters in metrics.variables. This function contemplates a fixed circular area, k-tree and angle-count plot designs, represented by line charts with estimated values of N.tls and G.tls (y axes) through plot sizes (x axes). The maximum sizes considered for each plot design must be specified in the plot.parameters argument. As in other functions, the absence of any of these arguments rules out the corresponding plot design. These charts can be generated for all the plots individually (no arguments specified) or for mean values across all plots or strata, including the standard deviation area (argument mean = TRUE). It is also possible to distinguish strata if they are included in a column named stratum, which must be given in the tree list input. The last function enables representation in the same chart of estimates of mean values for the different plot designs considered (Fig. 9), which can be executed by setting the argument all. plot.designs as TRUE.

#### 3.1.2.2. Field data available.

When field data are available, FORTLS also includes functions for determining the best possible plot design for yielding maximum correspondence between TLS-based and field estimates. These functions are based on comparison between TLS and their corresponding field data, and they form one of the main branches in the workflow (Fig. 6). The simulations function represents the first, compulsory step, which generates estimates of metrics and variables for a set of field plots and their TLS counterparts for circular fixed area, k-tree and angle-count plots. Thus, the list of trees detected in the point cloud and list of trees measured in the field are necessary as input data. In addition, probability based on distance sampling methods can be included as an optional argument if these methods are applied. Simulations are computed for continuous plot size increments of 0.1 m, 1 tree and 0.1 BAF by default, although other increments can be specified in the plot.parameters argument. The maximum value of plot parameter simulation must be specified for the respective elements radius.max, k. tree.max and BAF.max of the same argument, and, as in other functions, the absence of any of these arguments rules out the corresponding plot design. The number of largest trees per ha implemented in dominant variables can thus be specified in the num.trees element of the plot. parameters argument (by default 100 trees per ha). The simulations function will return a list with many elements (data.frame objects) as plot designs considered, in which each row will correspond to a simulated pair (plot, radius/k/BAF), and the columns will include all the metrics and variables estimates based on field and TLS data (Table 2). By default, these elements will be saved as separate .csv files in dir.result (otherwise specifying save.result = FALSE).

After generating simulations, two possible processes can be conducted (Fig. 6). One process assesses bias among variables estimated from TLS data and their counterpart estimated from field data and is carried out by the relative.bias function, which computes relative bias between field and TLS estimates (Eq. (23)). This will be executed with the simulated data generated previously in simulations, which are the input data of this function. The objective variables can be indicated in the variables argument, but by default those most conventionally used in

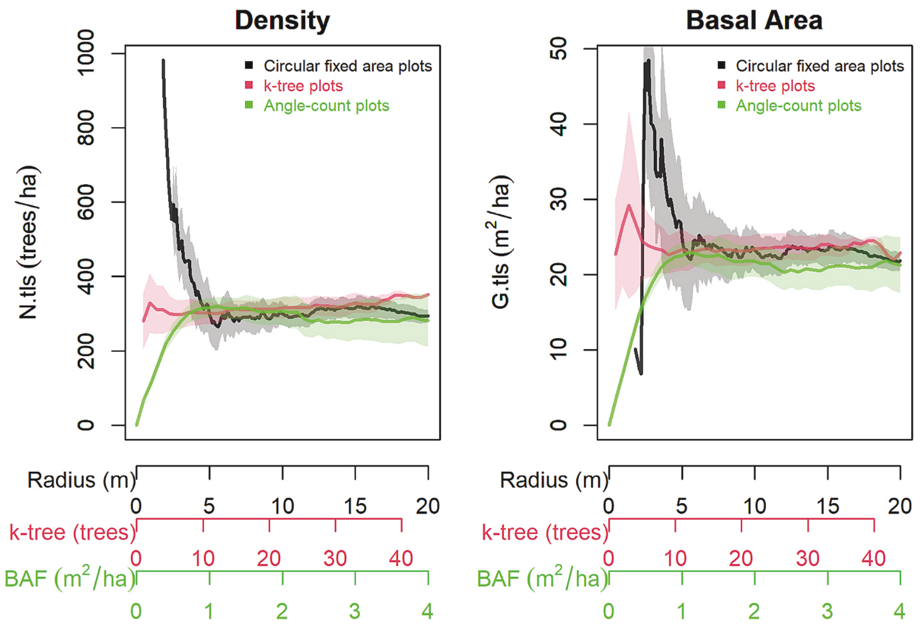


**Fig. 9.** Line charts output obtained with the estimation.plot.size function, with mean values of density (N.tls, trees $ha^{-1}$) and basal area (G.tls, $m^2$ $ha^{-1}$) estimates, derived from simulated TLS plots, for comparing circular fixed area, k-tree and angle-count plot designs.

forestry will be considered (N, G, V, d, dg, d.0, h and h.0). This function will return a list with as many elements (data frame objects) as plot designs considered, with rows corresponding to simulated pairs (plot, radius/k/BAF), and columns including relative bias between field and TLS variables estimates. By default, these elements will be saved as separate .csv files in dir.result (otherwise specified save.result = FALSE). In addition, interactive line charts with relative bias through size plot design will be generated by groups of variables (*N*, *G*, *V*, etc.) and plot design (supplementary material S2). These charts are very intuitive for assessing the best plot design and size for estimating variables directly from TLS data.

The other approach consists of evaluating the correlations between all metrics and variables estimated from TLS data and variables of interest estimated from field data: both Pearson and Spearman correlation coefficients are available. This is achieved with the correlations function, by means of the simulations with different previously generated plot design sizes. The target variables can be indicated in the variables argument, but by default those most commonly used in conventional forest inventories will be considered (N, G, V, d, dg, d.0, h and h.0). This function will return a list including the following: (i) correlation values for each method and plot design; (ii) another element with the same structure including the p-values of test for association corresponding to these correlations; and (iii) the strongest correlations for each set of simulations and the names of the TLS metrics or variables to which they correspond. The first and third elements will be saved as independent .csv files, one per correlation method and plot design (otherwise specified save.result = FALSE). This function will also generate interactive line charts, which will show the correlations between field variables of interest and all the TLS metrics and variables computed through regular continuous increase of plot size (supplementary material S3).

Finally, all the variables of interest can be assessed together according to the highest correlations achieved by means of interactive heatmaps using the optimize.plot.design function (Supplementary material S4). The input data will be the list including the strongest correlations obtained previously (third element of the list returned by correlations). The objective variables can be indicated in the variables argument, but by default those most commonly used in forest inventories will be considered (N, G, V, d, dg, d.0, h and h.0). The optimize.plot.design function is very useful for determining the most suitable plot design and size for all the variables of interest considered.

### 3.2. Experimental test case

We used the study plot to explore the potential of FORTLS to estimate forestry variables at stand level. We present some of the most important results here, but all of the analyses and outputs are reported in more detail in supplementary material (S1–S5).

#### 3.2.1. Estimation without field data available

As TLS plots are established on the basis of a systematic design sampling inventory and covering the same stand conditions, we considered assessing mean values across all TLS sampling points using the estimation.plot.size function and comparing all the plot designs in the same chart. Regarding N.tls, all of the plot designs yielded stable estimates from a given plot design size onwards. These estimates were 12–16 m for circular fixed area, 3–16 trees for k-tree, and 0.8–2 BAF in angle-count plots (Fig. 9). According to the standard deviation, different trends were observed in plot parameter values, which were quite stable in a circular fixed area, decreasing in k-tree, and increasing in angle-count plots. The patterns were very similar for G.tls estimates. In this case study, estimates in the stable zones reached very similar values for the three different designs. The stable zones showed similar pattern estimates for both N.tls and G.tls. For N.tls estimates, the estimated patterns almost completely coincided for a fixed radius of 13–17 m, 12–36 for k with the k-tree approach, and 1.2–1.6 for BAF, reaching approximately the value of 315 trees $ha^{-1}$ in all of these plot design, which is very close to 322 live trees $ha^{-1}$ measured in the field (Table 3). Regarding G.tls, estimates were similar for fixed radius of 13–17 m, 12–36 for k and 0.8–1.2 for BAF, reaching a slightly higher value than 23 $m^2$ $ha^{-1}$, which is lower than 25.29 $m^2$ $ha^{-1}$ estimated from field measurements (Table 3). Thus, in both cases the estimates slightly underestimated the stable sections of the curves relative to stand variables obtained from field measures.

#### 3.2.2. Estimation with field data available

Although all the variables and plot designs are assessed in supplementary material (S2), we focused here on *G* and circular fixed area plots to study relative bias in more detail. All of the variables estimated from TLS data showed positive values of relative bias for smaller plot sizes and negative values as plot size increased, with peaks reaching even higher than 50% for very small plots (Fig. 10). These peaks were

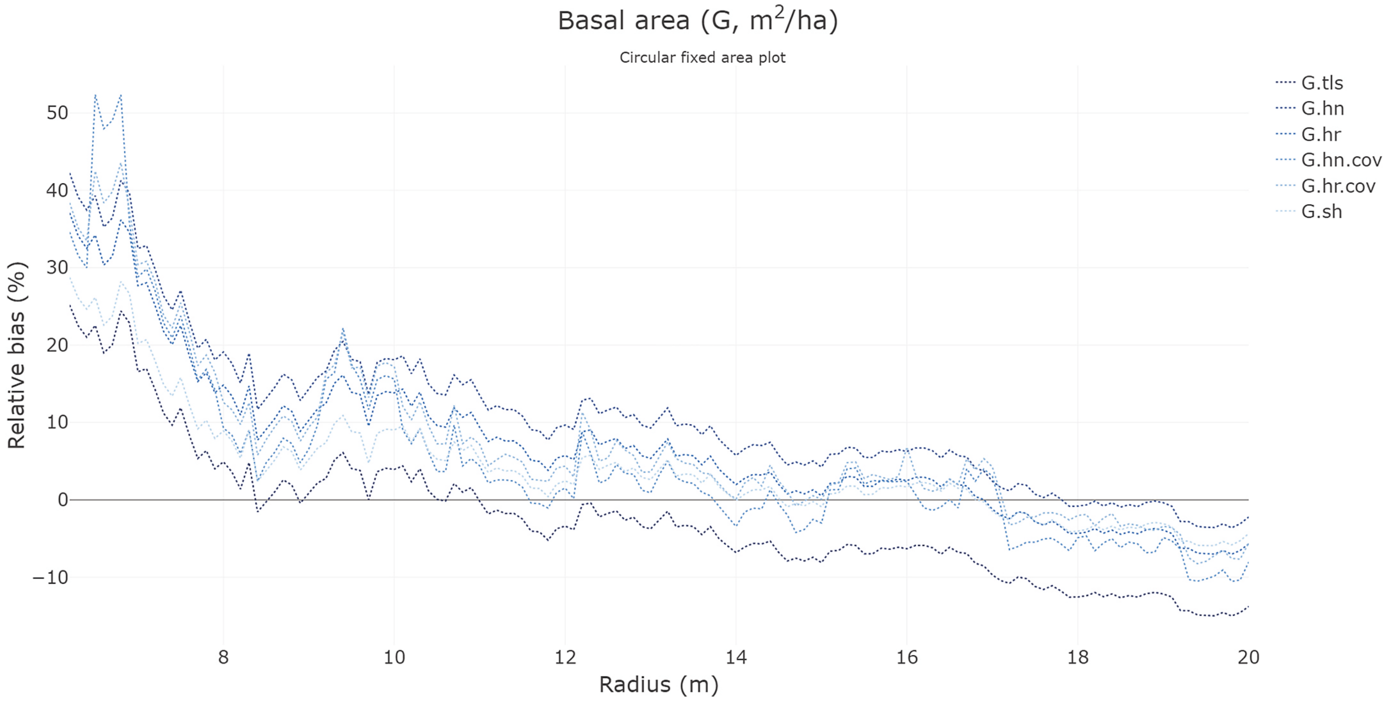


**Fig. 10.** Line chart output obtained with the relative.bias function for basal area ($G$, $m^2$ $ha^{-1}$) estimates, derived from simulated TLS and field plots, for circular fixed area plot design according to 0.1 m radius regular increments. The TLS variables assessed (legend) are explained in Table 2.

followed by a first sharp decrease until approximately 8 m radius and then a less steep decrease until the largest plot size. The best results were achieved with variables estimated with occlusion corrections methods, with most of them reaching low constant values of relative bias of radius between 15 and 17 m. However, the most stable estimates and lowest relative bias were based on half-normal detection function between 17.5 and 19 m radius, with relative bias below 1%.

Although all of the variables, plot designs and correlations methods (Pearson and Spearman) are assessed in supplementary material (S3), here we evaluated the Pearson correlation between arithmetic mean dominant height estimates based on field data (h.0) and the other metrics and variables obtained for TLS data for angle-count plots. In this case, the highest correlations were obtained with metrics corresponding to z coordinate percentiles, especially P95, which yielded correlations

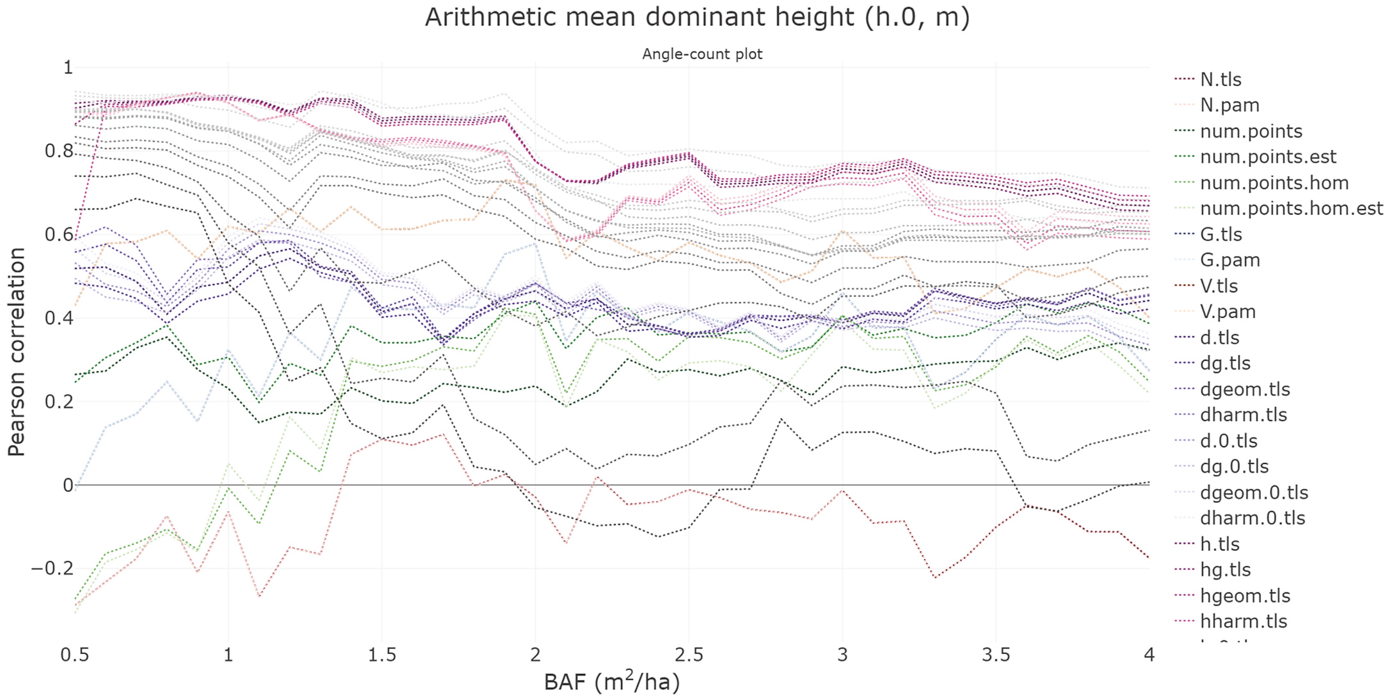


**Fig. 11.** Line chart output obtained with the correlations function for arithmetic mean dominant height ($H_0$, m) estimates, derived from simulated TLS and field plots, for angle-count plot design according to regular 0.1 $m^2$ $ha^{-1}$ increments in BAF (between 0.5 and 4, as the most usual values in forest inventory). The TLS variables and metrics assessed (legend) are explained in Table 2.

above 0.94, and h.0 estimates based on TLS data (Fig. 11). The general trend for all the metrics and variables was a slow decrease until a value of 1.9 BAF, with the highest correlations achieved in this BAF range. The decrease then became steeper until reaching values below 0.8. Interestingly, P95 performed better than the other metrics, with a less pronounced decrease.

In terms of correlation, the most suitable plot design considering performance of all the variables of interest can be assessed with optimize.plot.design. Again, this is evaluated for all plot design and correlation in supplementary information (S4), but here we only assessed the case of k-tree plot and Pearson correlation. Different trends differentiated groups of field variables. First, both h.0 and h yielded higher correlations when more trees were included and were reached for h earlier than for h.0 (Fig. 12). Once high correlations were reached, they remained stable above 0.9 until the largest k-tree plot. The diameter variables (d.0, dg and d) remained fairly stable across all plot sizes, with slightly higher values in smaller plots. However, the correlations were not very high, with values of between 0.6 and 0.8. For the other variables (*V*, *G* and *N*), the strongest correlations were obtained for smaller plots, with the highest values reached for plots of 6–10 trees, and they then decreased slightly before remaining stable. In the case of *N,* good correlations were also achieved for between 13 and 17 trees.

## 4. Discussion

Although some algorithms and applications have been developed for processing TLS point clouds for forestry purposes (Liang et al., 2018a; Krok et al., 2020), TLS has not yet been established as an operational device in FIs (Liang et al., 2016). To overcome this challenge, most researchers agree that automation of point cloud processing with attainable and easy-to-use software able to extract information related to important forest attributes is essential (Dassot et al., 2011; Newnham et al., 2015; White et al., 2016; Liang et al., 2016, 2018a). FORTLS contributes to making progress in this challenge by automating the processing of TLS point clouds for estimating FI variables. This is achieved by detecting trees and estimating some of their attributes and also generating metrics and/or variables related to conventional forestry variables at stand level. In addition, some of the functions enable assessment of the performance of metrics and variables estimates for different plot designs based on comparison with corresponding field data, thus allowing the best possible plot design to be established in each situation. This functionality confers flexibility to FORTLS in statistical inference for FIs, representing the main difference relative to similar applications (Bienert et al., 2007; Trochta et al., 2017).

Unlike other existing applications which extract tree attributes in high detail, such as Computree (Othmani et al., 2013), SimpleForest (Hackenberg et al., 2015), AdTree (Du et al., 2019) and TreeGSM (Raumonen et al., 2013), FORTLS mainly focuses on providing metrics and variables related to forestry attributes at stand-level. This approach, together with field implementation based on single scans, enables larger areas to be covered, although at the expense of lower detail at tree-level. We therefore consider that our approach perceives TLS devices as sampling rather than measurement instruments, and applications with similar perspective to ours are very scarce, e.g. 3D Forest (mainly developed for describing forest 3D structure) and AutoStem™ (mainly focused on timber production). Although use of these tools cannot be directly compared with our case study, because the methods and stands conditions are different, some useful information can be obtained. Regarding 3D Forest, the case study was framed in a stand characterized by highly variable canopy openness and dominated by sessile oak (*Quercus petraea* Matt.) with a mixture of other broadleaf species (Trochta et al., 2017). The authors sampled a plot of 2.4 ha in which the *dbh* of all the trees with *dbh* ≥ 10 cm (824) and *h* of 181 trees were measured. The plot was then scanned with a resolution of 2 mm at 10 m by multiple scans set at 44 × 44 m, which were co-registered before processing data in 3D Forest. The best results yielded a relative bias of 1% (positive) and 0.8% (negative) for $\overline{d}$ and $\overline{h}$ respectively. In our case, the relative bias for $\overline{d}$ was negative and slightly higher than in Trochta et al. (2017), with the best performance obtained for angle-count plot design, reaching peaks with lower relative bias than 1% for low values of BAF (S2). However, although the quadratic mean *dbh* ($d_g$) yielded the

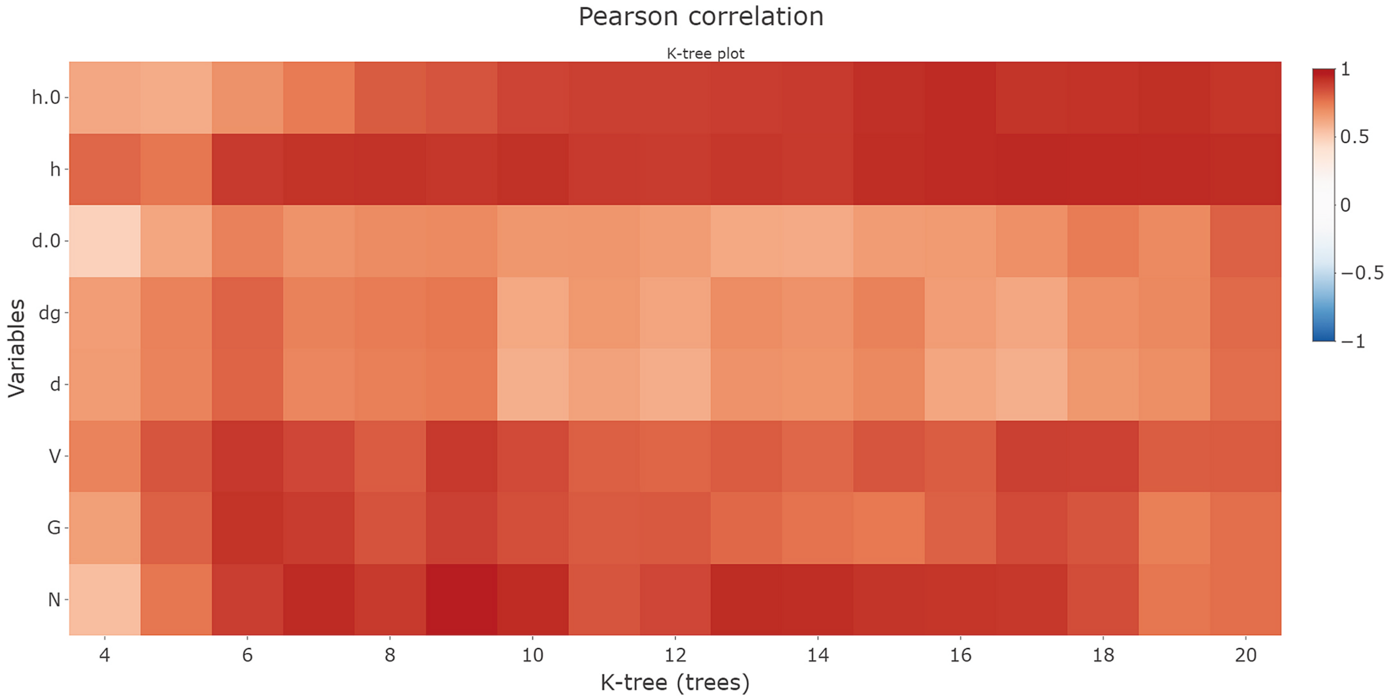


**Fig. 12.** Heat map output obtained with the optimize.plot.design function for estimates of density (*N*, trees ha$^{-1}$), basal area (*G*, m$^2$ ha$^{-1}$), volume (*V*, m$^3$ ha$^{-1}$), arithmetic mean *dbh* ($\overline{d}$, cm), quadratic mean *dbh* ($d_g$, cm), arithmetic mean dominant *dbh* ($d_0$, cm), arithmetic mean height ($\overline{h}$, m), and arithmetic mean dominant height ($H_0$, m), derived from simulated TLS and field plots, for k-tree plot design.

best performance in all cases, $\overline{h}$ was always underestimated, reaching similar relative bias only for circular fixed area plots of around 18 m radius (S2). This again shows that height variables are systematically underestimated (Liang et al., 2016, 2018a; Krok et al., 2020). On the other hand, AutoStem™ was assessed for a stand planted with *Picea sitchensis* (Bong.) Carr. with a current density of 600 trees per ha (Mengesha et al., 2015). This was measured in nine randomly located plots of 15 m radius, in which all trees were measured by conventional methods and single TLS scans were made from the plot centre with a resolution of 6.28 mm at 10 m. The overall difference between TLS-derived and conventional volume estimates was 5.6% when occluded trees (not detected by TLS) were excluded from the analysis of both sources, i.e. TLS and field data, and 10.2% when estimates based on TLS were corrected by simple correction factors. FORTLS yielded much lower relative bias in volume estimates, especially for the k-tree plot design and estimates corrected with distance sampling methods (G.hr.cov), which yielded values of around 0% for 12–14 trees (S2). In any case, comparison with these findings should be done with caution, as we used paraboloid function for estimating volume, in contrast to Mengesha et al. (2015), who applied methods based on retrieved stem profiles. Nevertheless, this is the most interesting comparison because the aforementioned authors used a very similar sampling methodology, with single randomly located scans (systematic in our case) and the corresponding field plot measurements.

All plot designs considered (circular fixed area, k-tree and angle-count plots) yielded stable estimates in N.tls and G.tls for certain plot size ranges (Fig. 9), as occurred for *G* estimates using a 2D TLS device (Fig. 3, Brunner and Gizachew, 2014). However, the variables were slightly underestimated, unlike in Brunner and Gizachew (2014), in which unbiased *G* estimates between real and scan basal area were observed for 5–10 m radius plot size (only for one of the studied stands). However, our findings are generally consistent with the most recent findings, which indicate underestimation of *N* and *G* due to deficit in tree detection caused by occlusions, especially for single scans (Liang et al., 2016, 2018a; Krok et al., 2020). In our study, uncorrected estimates (G.tls) yielded lower bias for smaller plot sizes, due to lower occlusion rates, which is consistent with the findings of Corona et al. (2019), who concluded that under easy-to-measure stand conditions, plots of 10 m radius in which occlusion corrections are not considered may be good enough for estimating *G*.

To overcome underestimates derived from occlusion effects, we incorporated several methods applied in TLS single scans in other studies, which improved estimations in larger plot sizes, where G.hn yielded the best estimates for 17.5–19 m radius (Fig. 10). This finding is consistent with those of Astrup et al. (2014) who reported that a larger detection radius seems to improve estimates when distance sampling methods are applied. The results presented here indicate that correction occlusion methods can improve estimates as observed for *G* in angle-count (Strahler et al., 2008; Lovell et al., 2011) and circular fixed area plots (Seidel and Ammer, 2014; Astrup et al., 2014). Because these methods can be assessed in FORTLS in terms of relative bias through continuous plot size increment and different plot designs, this represents an advantage for determining the best possible plot design in execution of a single workflow.

FORTLS also evaluates correlations between variables of interest and TLS-derived metrics and variables. This contribution implies a new perspective, enabling selection of the best possible plot design according to statistical correlation measures instead of measurement accuracy. This approach may be considered for estimating forestry variables assisted by or based on models, in a similar way as ABA inference developed for ALS devices (Næsset, 2002). In some cases, simple linear regressions may be fitted when strong relationships are observed for only one TLS metrics and/or variables, as between h.0 and P95 for the study case, with correlations above 0.94 (Fig. 11). In this case the plot design considered was an angle-count plot for a BAF value of 1.3. In addition, this concept provides an opportunity to solve the systematic problem regarding underestimation of height variables derived from TLS measurements (Liang et al., 2016, 2018a; Krok et al., 2020).

Different groups of variables yielded the highest correlations at different plot sizes (Fig. 12); when the correlations for height variables were strongest in larger plots, diameter variables retained approximate stable correlations, and the highest correlations for other variables (*N*, *G* and *V*) were reached in smaller plots. Thus, the plot design could be adapted to our stand conditions and target variables for more efficient sampling.

Here we have demonstrated the utility of the R package FORTLS in FIs in a case study. As FORTLS works with single scan data, co-registration of point clouds in specific software and placement of targets for field measurements are not required. This improves data acquisition and shortens the processing time, as well as enabling the sample size to be increased in a cost-efficient manner, which is one of the most desirable features of TLS in FIs (Liang et al., 2016). Further research to consolidate FORTLS for the approaches mentioned here should encompass the following: (i) larger and more complex study cases; (ii) consideration of more metrics and variables with high potential for correlation with other forest attributes (leaf area index, species, etc.); (iii) exploration of the possibility of making inferences assisted by models, by developing an adequate sampling methodology; and (iv) improvement of the computation process as much as possible, in relation to both algorithms and computing time.

## 5. Conclusions

The R package FORTLS is useful software for processing TLS data for forestry purposes. It has the advantage of working with single scans and conducting automatic data processing, which may overcome the major challenge of affordability in data acquisition and data processing. It has yielded good results for conventional variables, based on a preliminary case study with direct estimates, as well as good correlations between field-derived variables and TLS-derived metrics and variables. However, its potential for producing model-assisted inferences from metrics and/or variables has not yet been demonstrated. In addition, one of the most valuable features of the software is its flexibility to adapt to the best possible plot design for each variable, enabling multiple plot designs to be used in a single sampling design. Further research considering larger and more complex case studies is necessary to consolidate FORTLS as an operational tool in FIs, as well as to develop new metrics and variables.

## 6. Software availability

Name of software: FORTLS 1.0.6

Developers: Juan Alberto Molina-Valero, María José Ginzo Villamayor, Manuel Antonio Novo Pérez, Adela Martínez-Calvo, Juan Gabriel Álvarez-González, Fernando Montes, César Pérez-Cruzado

Contact Address: Unit for Sustainable Environmental and Forest Management (UXAFORES), Department of Agroforestry Engineering, Higher Polytechnic Engineering School, Universidade de Santiago de Compostela, Benigno Ledo s/n, Campus Terra, 27002 Lugo, Spain. Email: juanalberto.molina.valero@usc.es

Software required: R $\geq$ 3.5.0

First available: March 2, 2021

Availability: https://CRAN.R-project.org/package=FORTLS

Installation in R: install.packages('FORTLS')

Program languages: R and C++

License: GPL 3

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

This work was supported by the Spanish Ministry of Science and Innovation [AGL2016-76769-C2-2-R; PID2020-119204RB-C22] and Galician Regional Government [2020-CP031; ED431F 2020/02]; JAMV was supported by the Spanish Ministry of Science, Innovation and Universities through the FPU program [FPU16/03057]; AMC was supported by Galician Regional Government within the framework of the agreement "Development of the Galician continuous forest inventory" [2020-CP031]; CPC was supported by the Spanish Ministry of Science and Innovation [RYC2018-024939-I].

The authors thank Diego Lombardero Barrera, Joel Rodríguez Ruiz, Mario López Fernández and Óscar López Álvarez for help with fieldwork. We are also grateful for the comments made by anonymous reviewers, which helped us to improve the quality of the paper.

## Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.envsoft.2022.105337.

## References

Astrup, R., Ducey, M.J., Granhus, A., Ritter, T., von Lüpke, N., 2014. Approaches for estimating stand-level volume using terrestrial laser scanning in a single-scan mode. Canadian J. For. Res. 44 (6), 666–676. https://doiorg/10.1139/cjfr-2013-0535.

Bates, D., Eddelbuettel, D., 2013. Fast and elegant numerical linear algebra using the RcppEigen package. J. Stat. Software 52 (5), 1–24. https://doi.org/10.18637/jss.v052.i05.

Bienert, A., Scheller, S., Keane, E., Mohan, F., Nugent, C., 2007. Tree detection and diameter estimations by analysis of forest terrestrial laserscanner point clouds. In: ISPRS Workshop on Laser Scanning, vol. 36, pp. 50–55 (Espoo, Finland).

Bitterlich, W., 1948. Die winkelzählprobe. Allg. Forst-und holzwirt. Zeitung Wien., 59, pp. 4–5 (Winkelzählprobe = angle count sampling).

Bivand, R.S., Pebesma, E., Gomez-Rubio, V., 2013. Applied Spatial Data Analysis with R, vol. 2. Springer, New York.

Brunner, A., Gizachew, B., 2014. Rapid detection of stand density, tree positions, and tree diameter with a 2D terrestrial laser scanner. Eur. J. For. Res. 133, 819–831. https://doi.org/10.1007/s10342-014-0799-1.

Buckland, S.T., Anderson, D.R., Burnham, K.P., Laake, J.L., Borchers, D.L., Thomas, L., 2001. Introduction to Distance Sampling: Estimating Abundance of Biological Populations. Oxford University Press.

Cabo, C., Ordóñez, C., López-Sánchez, C.A., Armesto, J., 2018. Automatic dendrometry: tree detection, tree height and diameter estimation using terrestrial laser scanning. Int. J. Appl. Earth Obs. Geoinf. 69, 164–174. https://doi.org/10.1016/j.jag.2018.01.011.

Calders, K., Adams, J., Armston, J., Bartholomeus, H., Bauwens, S., Bentley, L.P., Chave, J., Danson, F.M., Demol, M., Disney, M., Gaulton, R., Krishna Moorthy, S.M., Levick, S.R., Saarinen, N., Schaaf, C., Stovall, A., Terryn, L., Wilkes, P., Verbeeck, H., 2020. Terrestrial laser scanning in forest ecology: expanding the horizon. Remote Sens. Environ. 251, 112102 https://doi.org/10.1016/j.rse.2020.112102.

Clark, R.G., 2016. Statistical efficiency in distance sampling. PLoS One 11 (3), e0149298. https://doi.org/10.1371/journal.pone.0149298.

Corona, P., Di Biase, R.M., Fattorini, L., D'Amati, M., 2019. A Monte Carlo appraisal of tree abundance and stand basal area estimation in forest inventories based on terrestrial laser scanning. Canadian J. For. Res. 49 (1), 41–52. https://doi.org/10.1139/cjfr-2017-0462.

Csárdi, G., FitzJohn, R., 2019. Progress: Terminal Progress Bars. R package version 1.2.2. https://CRAN.R-project.org/package=progress.

Danson, F.M., Disney, M.I., Gaulton, R., Schaaf, C., Strahler, A., 2018. The terrestrial laser scanning revolution in forest ecology. Interface Focus 8 (2), 20180001. https://doi.org/10.1098/rsfs.2018.0001.

Dassot, M., Constant, T., Fournier, M., 2011. The use of terrestrial LiDAR technology in forest science: application fields, benefits and challenges. Annals For. Sci. 68, 959–974. https://doi.org/10.1007/s13595-011-0102-2.

Du, S., Lindenbergh, R., Ledoux, H., Stoter, J., Nan, L., 2019. AdTree: accurate, detailed, and automatic modelling of laser-scanned trees. Rem. Sens. 11 (18), 2074. https://doi.org/10.3390/rs11182074.

Dubayah, R.O., Drake, J.B., 2000. Lidar remote sensing for forestry. J. For. 98 (6), 44–46. https://doi.org/10.1093/jof/98.6.44.

Ducey, M.J., Astrup, R., 2013. Adjusting for nondetection in forest inventories derived from terrestrial laser scanning. Can. J. Rem. Sens. 39 (5), 410–425.

Eddelbuettel, D., 2013. Seamless R and C++ Integration with Rcpp. Springer, New York, ISBN 978-1-4614-6867-7.

Eddelbuettel, D., Balamuta, J.J., 2018. Extending R with C++: a brief introduction to Rcpp. Am. Statistician 72 (1), 28–36. https://doi.org/10.1080/00031305.2017.1375990.

Eddelbuettel, D., François, R., 2011. Rcpp: seamless R and C++ integration. J. Stat. Software 40 (8), 1–18. https://doi.org/10.18637/jss.v040.i08.

Ester, M., Kriegel, H.P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Kdd 96 (34), 226–231.

Ferrara, R., Virdis, S.G.P., Ventura, A., Ghisu, T., Duce, P., Pellizzaro, G., 2018. An automated approach for wood-leaf separation from terrestrial LIDAR point clouds using the density based clustering algorithm DBSCAN. Agric. For. Meteorology 262, 434–444. https://doi.org/10.1016/j.agrformet.2018.04.008.

Garrett, R.C., Nar, A., Fisher, T.J., 2021. Ggvoronoi: Voronoi Diagrams and Heatmaps with 'ggplot2. R package version 0.8.4. https://CRAN.R-project.org/package=ggvoronoi.

Hackenberg, J., Spiecker, H., Calders, K., Disney, M., Raumonen, P., 2015. SimpleTree—an efficient open source tool to build tree models from TLS clouds. Forests 6 (11), 4245–4294. https://doi.org/10.3390/f6114245.

Hahsler, M., Piekenbrock, M., Doran, D., 2019. Dbscan: fast density-based clustering with R. J. Stat. Software 91 (1), 1–30. https://doi.org/10.18637/jss.v091.i01.

Hester, J., Wickham, H., 2020. Vroom: Read and Write Rectangular Text Data Quickly. R package version 1.3.2. https://CRAN.R-project.org/package=vroom.

Hijmans, R.J., 2020. Raster: Geographic Data Analysis and Modeling. R package version 3.4-5. https://CRAN.R-project.org/package=raster.

Holopainen, M., Vastaranta, M., Hyyppä, J., 2014. Outlook for the next generation's precision forestry in Finland. Forests 5 (7), 1682–1694. https://doi.org/10.3390/f5071682.

Jin, S., Tamura, M., Susaki, J., 2016. A new approach to retrieve leaf normal distribution using terrestrial laser scanners. J. For. Res. 27, 631–638. https://doi.org/10.1007/s11676-015-0204-z.

Kleinn, C., Vilčko, F., 2006. A new empirical approach for estimation in k-tree sampling. For. Ecol. Manag. 237 (1), 522–533. https://doi.org/10.1016/j.foreco.2006.09.072.

Krok, G., Kraszewski, B., Stereńczak, K., 2020. Application of terrestrial laser scanning in forest inventory–an overview of selected issues. For. Res. Pap. 81 (4), 175–194. https://doi.org/10.2478/frp-2020-0021.

Liang, X., Litkey, P., Hyyppa, J., Kaartinen, H., Vastaranta, M., Holopainen, M., 2012. Automatic stem mapping using single-scan terrestrial laser scanning. IEEE Trans. Geosci. Rem. Sens. 50 (2), 661–670. https://doi.org/10.1109/TGRS.2011.2161613.

Liang, X., Kankare, V., Hyyppä, J., Wang, Y., Kukko, A., Haggrén, H., Yu, X., Kaartinen, H., Jaakkola, A., Guan, F., Holopainen, M., Vastaranta, M., 2016. Terrestrial laser scanning in forest inventories. ISPRS J. Photogrammetry Remote Sens. 115, 63–77. https://doi.org/10.1016/j.isprsjprs.2016.01.006.

Liang, X., Hyyppä, J., Kaartinen, H., Lehtomäki, M., Pyörälä, J., Pfeifer, N., Holopainen, M., Brolly, G., Francesco, P., Hackenberg, J., Huang, H., Jo, H., Katoh, M., Liu, L., Makros, M., Morel, J., Olofsson, K., Poveda-Lopez, J., Trochta, J., Wang, D., Wang, J., Xi, Z., Yang, B., Zheng, G., Kankare, V., Louma, V., Yu, X., Chen, L., Vastaranta, M., Saarinen, N., Wang, Y., 2018a. International benchmarking of terrestrial laser scanning approaches for forest inventories. ISPRS J. Photogrammetry Remote Sens. 144, 137–179. https://doi.org/10.1016/j.isprsjprs.2018.06.021.

Liang, X., Kukko, A., Hyyppä, J., Lehtomäki, M., Pyörälä, J., Yu, X., Kaartinen, H., Jaakkola, A., Wang, Y., 2018b. In-situ measurements from mobile platforms: an emerging approach to address the old challenges associated with forest inventories. ISPRS J. Photogrammetry Remote Sens. 143, 97–107. https://doi.org/10.1016/j.isprsjprs.2018.04.019.

Liu, J., Liang, X., Hyyppä, J., Yu, X., Lehtomäki, M., Pyörälä, J., Zhu, L., Wang, Y., Chen, R., 2017. Automated matching of multiple terrestrial laser scans for stem mapping without the use of artificial references. Int. J. Appl. Earth Obs. Geoinf. 56, 13–23. https://doi.org/10.1016/j.jag.2016.11.003.

Lovell, J.L., Jupp, D.L.B., Newnham, G.J., Culvenor, D.S., 2011. Measuring tree stem diameters using intensity profiles from ground-based scanning lidar from a fixed viewpoint. ISPRS J. Photogrammetry Remote Sens. 66 (1), 46–55. https://doi.org/10.1016/j.isprsjprs.2010.08.006.

Ma, L., Zheng, G., Eitel, J.U., Moskal, L.M., He, W., Huang, H., 2016. Improved salient feature-based approach for automatically separating photosynthetic and non photosynthetic components within terrestrial lidar point cloud data of forest canopies. IEEE Trans. Geosci. Rem. Sens. 54 (2), 679–696. https://doi.org/10.1109/TGRS.2015.2459716.

Marques, F.F., Buckland, S.T., 2003. Incorporating covariates into standard line transect analyses. Biometrics 59 (4), 924–935. https://doi.org/10.1111/j.0006-341X.2003.00107.x.

McGaughey, R.J., 2009. FUSION/LDV: software for LIDAR data analysis and visualization. In: US Department of Agriculture, Forest Service, Pacific Northwest Research Station: Seattle, WA, USA, vol. 123, 2.

Mengesha, T., Hawkins, M., Nieuwenhuis, M., 2015. Validation of terrestrial laser scanning data using conventional forest inventory methods. Eur. J. For. Res. 134 (2), 211–222. https://doi.org/10.1007/s10342-014-0844-0.

Miller, D.L., Thomas, L., 2015. Mixture models for distance sampling detection functions. PLoS One 10 (3), e0118726. https://doi.org/10.1371/journal.pone.0118726.

Miller, D.L., Rexstad, E., Thomas, L., Marshall, L., Laake, J.L., 2019. Distance sampling in R. J. Stat. Software 89 (1), 1–28. https://doi.org/10.18637/jss.v089.i01.

Molina Valero, J.A., Ginzo Villamayor, M.J., Novo Pérez, M.A., Álvarez-González, J.G., Pérez-Cruzado, C., 2019. Estimación del área basimétrica en masas maduras de *Pinus sylvestris* en base a una única medición del escáner láser terrestre (TLS). Cuad. Soc. Esp. Cienc. For. 45 (3), 97–116. https://doi.org/10.31167/csecfv0i45.19887.

Molina-Valero, J.A., Ginzo Villamayor, M.J., Novo Pérez, M.A., Álvarez-González, J.G., Montes, F., Martínez-Calvo, A., Pérez-Cruzado, C., 2020. FORTLS: an R package for processing TLS data and estimating stand variables in forest inventories. Environ. Sci. Proc. 3, 38. https://doi.org/10.3390/IECF2020-08066.

Molina-Valero, J.A., Ginzo Villamayor, M.J., Novo Pérez, A.M., Martínez-Calvo, A., Álvarez-González, J.G., Montes, F., Pérez-Cruzado, C., 2021. FORTLS: Automatic

Processing of TLS Point Cloud Data for Forestry Purposes. R package version 1.0.6. https://CRAN.R-project.org/package=FORTLS.

Montes, F., Rubio-Cuadrado, A., Sánchez-González, M., Aulló-Maestro, I., Cabrera, M., Gómez, C., 2019. Occlusion probability in operational forest inventory field sampling with ForeStereo. Photogramm. Eng. Rem. Sens. 85 (7), 493–508. https://doi.org/10.14358/PERS.85.7.493.

Næsset, E., 2002. Predicting forest stand characteristics with airborne scanning laser using a practical two-stage procedure and field data. Remote Sens. Environ. 80 (1), 88–99. https://doi.org/10.1016/S0034-4257(01)00290-5.

Newnham, G.J., Armston, J.D., Calders, K., Disney, M.I., Lovell, J.L., Schaaf, C.B., Sthrahler, A.H., Danson, F.M., 2015. Terrestrial laser scanning for plot-scale forest measurement. Curr. Forestry Rep. 1, 239–251. https://doi.org/10.1007/s40725-015-0025-5.

Olofsson, K., Holmgren, J., 2016. Single tree stem profile detection using terrestrial laser scanner data, flatness saliency features and curvature properties. Forests 7 (9), 207. https://doi.org/10.3390/f7090207.

Olofsson, K., Holmgren, J., Olsson, H., 2014. Tree stem and height measurements using terrestrial laser scanning and the RANSAC algorithm. Rem. Sens. 6 (5), 4323–4344. https://doi.org/10.3390/rs6054323.

Othmani, A., Voon, L.F.C.L.Y., Stolz, C., Piboule, A., 2013. Single tree species classification from terrestrial laser scanning data for forest inventory. Pattern Recogn. Lett. 34 (16), 2144–2150. https://doi.org/10.1016/j.patrec.2013.08.004.

Pauly, M., Gross, M., Kobbelt, L.P., 2002. Efficient simplification of point-sampled surfaces. In: IEEE Conference on Visualization, pp. 163–170. https://doi.org/10.1109/VISUAL.2002.1183771. Boston, USA.

Pebesma, E.J., Bivand, R.S., 2005. Classes and methods for spatial data in R. R. News 5 (2), 9–13. https://CRAN.R-project.org/doc/Rnews/.

R Core Team, 2021. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Raumonen, P., Kaasalainen, M., Åkerblom, M., Kaasalainen, S., Kaartinen, H., Vastaranta, M., Holopainen, M., Disney, M., Lewis, P., 2013. Fast automatic precision tree models from terrestrial laser scanner data. Rem. Sens. 5 (2), 491–520. https://doi.org/10.3390/rs5020491.

Roussel, J.R., Auty, D., 2020. Airborne LiDAR Data Manipulation and Visualization for Forestry Applications. R package version 3.0.4. https://cran.r-project.org/package=lidR.

Roussel, J.R., Auty, D., Coops, N.C., Tompalski, P., Goodbody, T.R.H., Sanchez Meador, A., Bourdon, J.F., de Boissieu, F., Achim, A., 2020. lidR: an R package for analysis of Airborne Laser Scanning (ALS) data. Remote Sens. Environ. 251, 112061 https://doi.org/10.1016/j.rse.2020.112061.

Seidel, D., Ammer, C., 2014. Efficient measurements of basal area in short rotation forests based on terrestrial laser scanning under special consideration of shadowing. iFor. Biogeosci. For. 7 (4), 227–232. https://doi.org/10.3832/ifor1084-007.

Sievert, C., 2020. Interactive Web-Based Data Visualization with R, Plotly, and Shiny. CRC Press, ISBN 9781138331457. https://plotly-r.com.

Strahler, A.H., Jupp, D.L.B., Woodcock, C.E., Schaaf, C.B., Yao, T., Zhao, F., Yang, X., Lovell, J., Culvenor, D., Newnham, G., Ni-Miester, W., Boykin-Morris, W., 2008. Retrieval of forest structural parameters using a ground-based lidar instrument (Echidna®). Can. J. Rem. Sens. 34 (Suppl. 2), S426–S440. https://doi.org/10.5589/m08-046.

Tomppo, E., Gschwantner, T., Lawrence, M., McRoberts, R.E., Gabler, K., Schadauer, K., Cienciala, E., 2010. National forest inventories. Pathways for common reporting. European Science Foundation 1, 541–553. https://doi.org/10.1007/978-90-481-3233-1.

Trochta, J., Krůček, M., Vrška, T., Král, K., 2017. 3D Forest: an application for descriptions of three-dimensional forest structures using terrestrial LiDAR. PLoS One 12 (5), e0176871. https://doi.org/10.1371/journal.pone.0176871.

Vaidyanathan, R., Xie, Y., Allaire, J.J., Cheng, J., Sievert, C., Russell, K., 2020. Htmlwidgets: HTML Widgets for R. R package version 1.5.3. https://CRAN.R-project.org/package=htmlwidgets.

West, G.B., Brown, J.H., Enquist, B.J., 1999. The fourth dimension of life: fractal geometry and allometric scaling of organisms. Sci 284 (5420), 1677–1679. https://doi.org/10.1126/science.284.5420.1677.

White, J.C., Coops, N.C., Wulder, M.A., Vastaranta, M., Hilker, T., Tompalski, P., 2016. Remote sensing technologies for enhancing forest inventories: a review. Can. J. Rem. Sens. 42 (5), 619–641. https://doi.org/10.1080/07038992.2016.1207484.

Wickham, H., 2021. Tidyr: Tidy Messy Data. R package version 1.1.3. https://CRAN.R-project.org/package=tidyr.

Wickham, H., Seidel, D., 2020. Scales: Scale Functions for Visualization. R package version 1.1.1. https://CRAN.R-project.org/package=scales.

Wickham, H., Hester, J., Chang, W., 2021. Devtools: Tools to Make Developing R Packages Easier. R package version 2.4.2. https://CRAN.R-project.org/package=devtools.

Wulder, M.A., White, J.C., Nelson, R.F., Næsset, E., Ørka, H.O., Coops, N.C., Hilker, T., Bater, C.W., Gobakken, T., 2012. Lidar sampling for large-area forest characterization: a review. Remote Sens. Environ. 121, 196–209. https://doi.org/10.1016/j.rse.2012.02.001.

Xia, S., Wang, C., Pan, F., Xi, X., Zeng, H., Liu, H., 2015. Detecting stems in dense and homogeneous forest using single-scan TLS. Forests 6 (11), 3923–3945. https://doi.org/10.3390/f6113923.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., Yan, G., 2016. An easy-to-use airborne LiDAR data filtering method based on Cloth simulation. Rem. Sens. 8 (6), 501. https://doi.org/10.3390/rs8060501.

Zhang, W., Wan, P., Wang, T., Cai, S., Chen, Y., Jin, X., Yan, G., 2019. A novel approach for the detection of standing tree stems from plot-level terrestrial laser scanning data. Rem. Sens. 11 (2), 211. https://doi.org/10.3390/rs11020211.