Package 'PiC'

July 29, 2026

Type Package

Title Interactive Processing and Segmentation of Forest TLS Point-Cloud Data

Version 3.3.3

Description Tools for the processing, segmentation, and analysis of terrestrial laser scanning (TLS and MLS) forest point-cloud data. The package provides fast voxel-based processing, classification of point clouds into forest floor, understory, canopy, and woody components, and algorithms for single-tree analysis and structural characterization. Methods are designed to handle large and dense point-cloud datasets efficiently, supporting applications in forest structure assessment, connectivity analysis, and fire-risk evaluation. Input data are provided as '.xyz', '.txt', '.las', or '.laz' point-cloud files. The circle-fitting routines used for diameter estimation are adapted, in base R, from the 'conicfit' package (GPL-3) by Jose Gama, based on the original algorithms and code by Nikolai Chernov.
For methodological details, see Ferrara and Arrizza (2025) <https://hdl.handle.net/20.500.14243/533471> and Ferrara et al. (2018) <doi:10.1016/j.agrformet.2018.04.008>.

License GPL (>= 3)

Depends R (>= 4.3)

Imports collapse, data.table, dbscan, dplyr, magrittr, RANN, stats, tictoc, terra, tools, utils

Suggests DT, fs, ggplot2, lidR, grid, gridExtra, later, plotly, shiny, shinycssloaders, shinydashboard, shinydashboardPlus, shinyFeedback, shinyFiles, shinyjs, shinythemes, scales, shinyWidgets, testthat (>= 3.0.0), tidyr, viridis, withr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/rupppy/PiC

BugReports https://github.com/rupppy/PiC/issues

Additional_repositories https://r-lidar.r-universe.dev

NeedsCompilation no

Author Roberto Ferrara [aut, cre] (ORCID:
<https://orcid.org/0009-0000-3627-6867>),
Stefano Arrizza [ctb] (ORCID: <https://orcid.org/0009-0009-2290-3650>),
Nikolai Chernov [cph] (Author of the original circle-fitting algorithms
and code),
Jose Gama [ctb] (Author of the R implementation of the circle-fitting
code in the 'conicfit' package)

Maintainer Roberto Ferrara <roberto.ferrara@cnr.it>

Repository CRAN

Date/Publication 2026-07-29 13:40:09 UTC

Contents

`Floseg` *Forest floor segmentation with DTM validation*

Description

Segments the input point cloud into forest floor and above-ground biomass (AGB) using a two-stage DTM approach: 1. Create coarse DTM (e.g., 0.5m resolution) with outlier removal and gap filling 2. Interpolate fine DTM_small (0.1m resolution) from filled coarse DTM 3. Extract points within vertical tolerance of DTM_small surface

This approach is robust on sloped or irregular terrain.

Usage

```
Floseg(
  a,
  filename = "XXX",
  dtm_coarse_res = 0.5,
  dtm_fine_res = 0.1,
  tolerance = 0.4,
  clean_outliers = TRUE,
```

```
  outlier_k = 1,
  output_path = tempdir()
)
```

Arguments

a	Input point cloud data frame (x,y,z columns) or file path (.xyz)
filename	Output file prefix (default = "XXX")
dtm_coarse_res	Coarse DTM resolution in meters (default = 0.5)
dtm_fine_res	Fine DTM resolution in meters (default = 0.1)
tolerance	Vertical tolerance for floor extraction in meters (default = 0.4)
clean_outliers	Remove local outliers from coarse DTM (default = TRUE)
outlier_k	Z-score threshold for outlier removal (default = 1)
output_path	Directory where output files will be written (default = tempdir())

Value

List with floor_points, agb_points, dtm_coarse, dtm_fine, and file paths

---

Forest_seg *Forest component segmentation, plot and tree metrics*

Description

`Forest_seg()` is the main end-to-end routine of the package. It takes a raw terrestrial or mobile laser scanning (TLS/MLS) point cloud and runs a complete eight-stage pipeline that reconstructs the forest floor, recognises the woody structures, measures individual trees, separates crown from understory, and writes fully classified point clouds together with per-tree and per-plot reports and a reproducibility log.

All processing is performed on native float coordinates. A single global shift is applied at the start for numerical precision and reversed once at the end, so that output coordinates are returned in the original reference system. The wood and crown analysis relies on three dedicated algorithms, **LOR** (Ligneous Object Recognition), **DAV** (Directional Anisotropy of Vegetation) and **GAB** (Geometrical Aggregation of Biomass), described in the Details section.

Usage

```
Forest_seg(
  a,
  filename = "XXX",
  integer_precision = "mm",
  dtm_coarse_res = 0.5,
  tolerance = 0.4,
  dimVox = 2,
  th = 2,
```

```
  eps = 2,
  mpts = 9,
  h_trunk = 3,
  N = 3000,
  w_linear = 0.5,
  Vox_print = FALSE,
  Woodpoints_print = TRUE,
  h_rescue_min = 1,
  h_rescue_max = 5,
  dbh_tolerance = 0.05,
  dbh_max_rmse = 5,
  dbh_min_radius = 0.025,
  dbh_max_radius = 0.5,
  canopy_vox_dim = 0.15,
  canopy_min_density = 500,
  dav_understory_max_start = 1.3,
  dav_height_factor = 0.8,
  dav_median_height_factor = 0.5,
  dav_eps = 2,
  dav_minPts = 4,
  calculate_cbh = TRUE,
  cbh_hex_side = 0.15,
  cbh_min_branch_length = 2,
  cbh_save_points = TRUE,
  output_format = "las",
  output_path = tempdir(),
  generate_reports = TRUE,
  ...
)
```

Arguments

a	Input point cloud data frame (.xyz format) or file path
filename	Output file prefix (default = "XXX")
integer_precision	Coordinate decimal precision: "mm" (3 decimals) or "cm" (2 decimals)
dtm_coarse_res	Coarse DTM/DSM resolution in m (default = 0.5). Fine resolution is automatically set to half. DSM uses same resolution for height calculation.
tolerance	Vertical tolerance for floor extraction in m (default = 0.4)
dimVox	Voxel dimension in cm for LOR wood segmentation (default = 2)
th	Minimum points per voxel (default = 2)
eps	DBSCAN epsilon radius in voxel units (default = 2)
mpts	DBSCAN minimum points (default = 9)
h_trunk	Minimum trunk length in m (default = 3)
N	Minimum voxels in wood cluster (default = 3000)
w_linear	Wood cluster quality filter (default = 0.5)

`Vox_print` Save voxelization? (default = FALSE)

`Woodpoints_print`
Save wood points? (default = TRUE)

`h_rescue_min` Minimum normalized height (m above DTM) for RESCUE PASS wood recovery (default = 1)

`h_rescue_max` Maximum normalized height (m above DTM) for RESCUE PASS wood recovery (default = 5)

`dbh_tolerance` Vertical tolerance for DBH slice in m (default = 0.05)

`dbh_max_rmse` Maximum RMSE for DBH fit in cm (default = 5)

`dbh_min_radius` Minimum valid DBH radius in m (default = 0.025, i.e. 5 cm DBH)

`dbh_max_radius` Maximum valid DBH radius in m (default = 0.5, i.e. 100 cm DBH)

`canopy_vox_dim` Voxel size for DAV (Directional Anisotropy of Vegetation) analysis in m (default = 0.15)

`canopy_min_density`
Minimum canopy density in pts/m^3 (default = 500)

`dav_understory_max_start`
Maximum start height for understory in m (default = 1.3)

`dav_height_factor`
Internal DAV parameter. Upper fraction of median tree height for understory ceiling (default = 0.8, stable value).

`dav_median_height_factor`
Internal DAV parameter. Fraction of median tree height for mass distribution threshold (default = 0.50, stable value).

`dav_eps` DBSCAN epsilon for DAV (default = 2)

`dav_minPts` DBSCAN minPts for DAV (default = 4)

`calculate_cbh` Calculate crown base height? (default = TRUE)

`cbh_hex_side` Hexagon edge length in m (default = 0.15)

`cbh_min_branch_length`
Minimum horizontal extent of foliage in m (default = 2.0)

`cbh_save_points`
Save CBH points? (default = TRUE)

`output_format` Output format: "las" (single classified LAS file, default) or "xyz" (separate files per class). Any value other than "xyz" falls back to "las".

`output_path` Output directory (default = tempdir())

`generate_reports`
Generate CSV reports? (default = TRUE)

`...` Currently unused; reserved for future extensions

Details

Processing pipeline (8 stages)

1. *Load and shift* - the input (data frame or `.xyz` / `.txt` / `.las` / `.laz` file) is coerced to an `x, y, z` table and a global coordinate shift is applied.

2. *Forest floor extraction* - a coarse/fine ground model (DTM) separates the forest floor from low vegetation and above-ground biomass (AGB).
3. *LOR (Ligneous Object Recognition)* - woody points (trunks and branches) are identified.
4. *Tree detection and metrics* - tree positions and heights are derived from the woody clusters.
5. *DBH* - diameter at breast height is fitted at multiple heights.
6. *DAV (Directional Anisotropy of Vegetation)* - foliage is split into tree crown and understory.
7. *GAB (Geometrical Aggregation of Biomass)* - crown base height (CBH) is estimated.
8. *Output* - the classified point cloud, the reports and the log are written.

Point classification scheme

Every point is assigned an integer class, consistent with the codes stored in the classified LAS file:

- 2 Forest floor (ground / vegetal soil)
- 3 Understory (low vegetation and shrubs below the crown)
- 4 Wood (trunks and branches)
- 5 Crown / foliage
- 6 Non-valid trees (clusters that failed DBH validation)
- 7 Noise (isolated or non-classifiable points)

LOR - Ligneous Object Recognition (stage 3)

AGB points are voxelised (`dimVox`, `th`) and grouped with a density-based clustering (DBSCAN, parameters `eps` / `mpts`). Clusters are kept as woody structures when they exceed a minimum size (`N`), satisfy a linearity/quality filter (`w_linear`) and reach a minimum trunk length (`h_trunk`). A dedicated rescue pass (`h_rescue_min`-`h_rescue_max`) recovers woody points in the height band where trunks and low branches are most often missed.

DAV - Directional Anisotropy of Vegetation (stage 6)

The remaining foliage is voxelised at `canopy_vox_dim` and filtered by a minimum density (`canopy_min_density`). Voxels are clustered (`dav_eps` / `dav_minPts`) and split into crown and understory using an adaptive height threshold derived from the median tree height (`dav_height_factor`, `dav_median_height_factor`, capped by `dav_understory_max_start`). Low vegetation from the forest floor stage is assigned directly to the understory. The voxel classification is then mapped back to the original points, and crown and understory volumes are accumulated for the plot report.

GAB - Geometrical Aggregation of Biomass / CBH (stage 7)

When `calculate_cbh = TRUE`, crown and wood voxels are merged for spatial continuity and projected onto a hexagonal tessellation (`cbh_hex_side`). Each foliage cell is assigned to the nearest tree by Voronoi partitioning on voxel coordinates, and a breadth-first connectivity trace (trunk to branches to foliage) identifies the lowest continuously foliated branch, whose height is reported as the crown base height. The step runs in parallel across the available cores. A minimum foliage extent (`cbh_min_branch_length`) guards against spurious detections.

DBH measurement (stage 5)

Diameter is estimated by circle fitting (Pratt/Landau) on horizontal slices at 1.3 m, with fallbacks at 1.8 m and 2.3 m. A fit is accepted only within the radius range (`dbh_min_radius`-`dbh_max_radius`) and below the maximum RMSE (`dbh_max_rmse`); trees failing all heights are flagged as non-valid (class 6).

Outputs

- *Classified point cloud* - by default a single classified LAS file (`output_format = "las"`) carrying the class codes above; with `output_format = "xyz"` one plain-text file per class is written instead.
- *Tree report* (`<filename>_tree_report.csv`) - one row per tree: identifier, class, position (X, Y, Z), height, DBH and its RMSE, validity flag, and CBH when computed.
- *Plot report* (`<filename>_plot_report.csv`) - stand-level statistics including tree density, basal area, crown and understory volume, and canopy coverage.
- *CBH diagnostics* - per-tree CBH details, written when `calculate_cbh = TRUE`.
- *Parameters log* - a complete record of every parameter (exposed and internal), the input summary, the coordinate shift, the software/system versions and the timing, for full reproducibility.

Report and log writing is controlled by `generate_reports`.

Fixed internal parameters (not exposed in the interface)

- `dtm_fine_res`: automatically set to `dtm_coarse_res / 2`
- `dbh_heights`: fixed at `c(1.3, 1.8, 2.3)` metres
- `dbh_min_points`: fixed at 8 (works well for both Pratt and Landau fitting)

Value

A named list with the in-memory `tree_metrics` table, the paths of the files written (`tree_report_file`, `plot_report_file`, `las_file` or, in XYZ mode, the per-class `crown_file`, `understory_file`, `wood_file`, `noiseL_file`, `noiseH_file`, `non_valid_trees_file`), the `parameters_log` path, and `shift_info` (the applied coordinate shift).

`metrics_from_las` *Calculate tree and plot metrics from a classified LAS file*

Description

Recalculates tree-level and plot-level metrics starting from a classified LAS file. Intended for use after manual correction of an automatic classification produced by `Forest_seg`.

Wood points (class 4) are re-clustered into individual trees using DBSCAN with fixed, permissive parameters (5 cm voxels, eps = 2, minPts = 5). These parameters are intentionally loose because the classification is already trusted; DBSCAN here only separates spatially distinct trunks.

Tree height is computed as the difference between the tree base elevation (minimum z of the wood cluster) and the maximum z of any point in the full cloud within a 0.5 m XY buffer around the tree base, consistent with the `Forest_seg` approach.

Crown Base Height (CBH) is calculated with the GAB (Geometrical Aggregation of Biomass) Voronoi method, identical to `Forest_seg`.

Output CSV files use the same column layout as `Forest_seg` reports.

Usage

```
metrics_from_las(
  las_file,
  output_path = NULL,
  filename = NULL,
  dbh_tolerance = 0.05,
  dbh_max_rmse = 5,
  dbh_min_radius = 0.025,
  dbh_max_radius = 0.8,
  calculate_cbh = TRUE,
  cbh_hex_side = 0.15,
  cbh_min_branch_length = 2,
  canopy_vox_dim = 0.15,
  canopy_min_density = 100,
  generate_reports = TRUE
)
```

Arguments

`las_file`	Character. Path to the classified LAS file.
`output_path`	Character. Directory for output files. Default: same directory as `las_file`.
`filename`	Character. Prefix for output file names. Default: derived from `las_file` (stripping `_classified`).
`dbh_tolerance`	Vertical half-width of the DBH slice in metres (default = 0.05).
`dbh_max_rmse`	Maximum acceptable RMSE for circle fit in cm (default = 5).
`dbh_min_radius`	Minimum valid stem radius in metres (default = 0.025).
`dbh_max_radius`	Maximum valid stem radius in metres (default = 0.8).
`calculate_cbh`	Logical. If TRUE, calculate Crown Base Height using the GAB method (default = TRUE). Requires crown points (class 5).
`cbh_hex_side`	Hexagonal cell side length in metres for GAB (default = 0.15).
`cbh_min_branch_length`	Minimum horizontal foliage extent in metres to consider a branch valid for CBH (default = 2.0).
`canopy_vox_dim`	Voxel size in metres for crown and wood voxelisation used in GAB CBH (default = 0.15).
`canopy_min_density`	Minimum point density in pts/m^3 used to derive the minimum points per hex cell for GAB (default = 100).
`generate_reports`	Logical. If TRUE (default), saves `<filename>_tree_report.csv` and `<filename>_plot_report.csv`.

Value

Invisibly returns a named list:

tree_metrics data.table with one row per detected tree.

plot_stats data.frame with plot-level aggregate metrics.

tree_report Path to the tree-level CSV, or NULL.

plot_report Path to the plot-level CSV, or NULL.

LAS classification codes expected

- 2 = Ground / forest floor (convex hull for plot area)
- 3 = Understory
- 4 = Wood (re-clustered into individual trees)
- 5 = Crown / foliage (used for height buffer and CBH)
- 6 = Non-valid trees (ignored)
- 7 = Noise (ignored)

See Also

Forest_seg, SegOne

---

pic_analyze_cloud *Point cloud diagnostic analysis*

Description

Analyzes point cloud quality and structure with voxel density analysis. Generates density maps and distribution curves. Results are always returned as plots (for Shiny) and optionally exported as PDF.

Usage

```
pic_analyze_cloud(
  points,
  voxel_sizes = 0.1,
  generate_pdf = TRUE,
  pdf_output = NULL,
  output_path = tempdir()
)
```

Arguments

points	Data frame with columns x, y, z (or will be coerced)
voxel_sizes	Numeric vector of voxel sizes in meters for analysis
generate_pdf	Logical, whether to generate PDF report (default = TRUE)
pdf_output	Character, PDF filename (if NULL and generate_pdf=TRUE, auto-generated)
output_path	Character, directory for PDF output (default = tempdir())

Value

List containing: - summary: Basic statistics (n_points, extent, area) - density_analysis: Density per m^2 statistics and raster - voxel_analysis: Multi-resolution voxel statistics - plots: Named list of ggplot2 objects (density_raster, density_curve) - text_output: Character vector with formatted statistics - pdf_file: Path to PDF if generated, NULL otherwise

---

`run_PiC` *Launch PiC Shiny App*

Description

Launch the Shiny app for interactive 3D point cloud processing.

Usage

```
run_PiC()
```

Details

This function launches an interactive web application for analyzing forest point cloud data. The app requires additional packages that are not installed by default. If these packages are missing, you will be prompted to install them.

To stop the server completely: - Use the "Exit Application" button (cleanest method) - Or press Ctrl+C in the R console / click Stop in RStudio

Note: Simply closing the browser tab will disconnect the interface but the server will continue running in R until you stop it.

Value

No return value, called for side effects (launches Shiny app)

Examples

```
## Not run:
# Launch the interactive app
run_PiC()

# To stop the server: click "Exit Application" button
# Or: press Ctrl+C / click Stop button in RStudio

## End(Not run)
```

SegOne *Single Tree Wood-Leaf Segmentation and Comprehensive Metrics Calculation*

Description

Performs wood-leaf segmentation and calculates comprehensive structural metrics for individual trees from terrestrial laser scanning (TLS) point cloud data.

Version 4.2 uses shared_utils.R functions for code reuse and consistency with the Forest_seg pipeline.

The analysis follows a four-stage processing pipeline:

1. Voxelization and wood component identification using DBSCAN clustering
2. Foliage separation through voxel-based subtraction
3. Tree structural metrics calculation (height, DBH, crown base)
4. Canopy volume quantification using density-weighted voxel analysis

Usage

```
SegOne(a, filename = "Elab_single_tree", dimVox = 2, th = 2,
       eps = 2, mpts = 6, N = 1000, R = 30, output_path = tempdir(),
       calculate_metrics = TRUE, voxel_size_canopy = 0.1,
       coverage_method = "linear")
```

Arguments

a	Input point cloud data. Can be either: (1) a data frame with x, y, z coordinates, or (2) a file path to a .txt or .xyz file containing point cloud data
filename	Character string specifying the output file prefix (default: "Elab_single_tree")
dimVox	Numeric value specifying voxel dimension in centimeters for wood segmentation. Typical range: 1-5 cm. Smaller values increase computational cost but improve spatial resolution (default: 2)
th	Integer specifying minimum number of points required to generate a voxel. Used to filter noise and low-density regions (default: 2)
eps	Numeric value specifying the epsilon neighborhood radius for DBSCAN clustering in voxel units. Determines spatial connectivity of wood clusters (default: 2)
mpts	Integer specifying minimum number of points required in epsilon neighborhood for a voxel to be considered a core point in DBSCAN algorithm (default: 6)
N	Integer specifying minimum number of voxels required to form a valid wood cluster. Filters small non-wood clusters (default: 1000)
R	Numeric threshold for cluster shape parameter (standard deviation proportion of variance from PCA). Used to identify cylindrical/linear wood structures. Higher values are more restrictive (default: 30)

`output_path` Character string specifying directory path for output files. If directory does not exist, it will be created (default: tempdir())

`calculate_metrics`
Logical flag indicating whether to calculate comprehensive tree metrics. If FALSE, only wood-leaf segmentation is performed (default: TRUE)

`voxel_size_canopy`
Numeric value specifying voxel size in meters for canopy volume calculation. Typical range: 0.05-0.2 m (default: 0.1)

`coverage_method`
Character string specifying method for calculating coverage degree in canopy volume analysis. Options: "linear", "mean_normalized", "exponential", "threshold", "mediterranean" (default: "linear")

Details

Processing Pipeline

Stage 1: Wood Segmentation

Wood components are identified through a multi-step process:

1. Point cloud voxelization at resolution specified by `dimVox`
2. DBSCAN clustering applied to voxel centroids using `eps` and `mpts`
3. Principal Component Analysis (PCA) filtering to identify cylindrical structures
4. Retention of clusters with shape parameter R exceeding threshold (cylindrical wood)

The PCA-based filtering exploits the geometric properties of tree stems and branches, which exhibit high first principal component values due to their elongated structure.

Stage 2: Foliage Separation

Foliage points are extracted using voxel-based subtraction:

1. Both wood and total point cloud are voxelized at 0.2 m resolution
2. Wood-occupied voxels are identified
3. Non-wood voxels are retained and mapped back to original points

This approach ensures complete spatial separation between wood and foliage components.

Stage 3: Structural Metrics Calculation

When `calculate_metrics = TRUE`, the following metrics are computed:

- **Tree Base Location (X, Y, Z_min)**: Coordinates of lowest wood point
- **Tree Height**: Vertical distance from base to highest point within 1 m buffer
- **DBH (Diameter at Breast Height)**: Calculated at 1.3 m using Pratt circle fitting algorithm with +/- 5 cm tolerance. Valid range: 5-300 cm. DBH value is always reported in CSV output. DBH_RMSE_cm column provides fit quality (max 5 cm for validation). DBH_valido flag indicates whether measurement meets quality standards.
- **Crown Base Height**: Detected using vertical density analysis to filter noise, followed by gap analysis. Uses 0.5 m bins with minimum 3 points per bin. Valid range: 0.5 m to 90

Stage 4: Canopy Volume Quantification

Canopy volume metrics are calculated using density-weighted voxel analysis:

1. Foliage points voxelized at resolution `voxel_size_canopy`
2. Point density calculated per voxel
3. Coverage degree computed using specified `coverage_method`
4. Two volume metrics calculated:
   - **Canopy Volume**: Total occupied voxel volume (m^3)
   - **Occupied Volume**: Density-weighted volume accounting for point distribution
5. Coverage area computed from ground projection of occupied voxels (mq)

Improvements in Version 2.0

Enhanced Crown Base Calculation:

- Vertical density analysis filters noise points (isolated points near trunk)
- Uses 0.5 m vertical bins with minimum 3 points per bin threshold
- Combines density filtering with gap analysis for robust detection
- Validates results with multiple criteria

Improved DBH Validation:

- Updated maximum diameter to 3.0 m (300 cm) for large/monumental trees
- RMSE quality check (max 5 cm) used to flag poor circle fits
- DBH value always reported (even if RMSE threshold exceeded)
- DBH_RMSE_cm column provides fit quality indicator
- DBH_valido flag indicates whether measurement meets all quality standards
- Enhanced diagnostic messages showing fit quality

Coverage Degree Methods

The `coverage_method` parameter determines how point density is translated to coverage degree:

- **linear**: Linear normalization by column maximum: $CD = N/N_{max}$
- **mean_normalized**: Normalization by mean density: $CD = N/\bar{N}$
- **exponential**: Exponential saturation: $CD = 1 - exp(-N/\bar{N})$
- **threshold**: Binary classification at 50th percentile
- **mediterranean**: Power-law scaling optimized for sparse Mediterranean canopies: $CD = (N/N_{max})^{0.7}$

For single trees, "linear" is recommended as it provides intuitive interpretation of point density relative to maximum observed density.

Quality Assurance

The function implements several validation checks:

- DBH validation: radius 2.5-150 cm, minimum 5 points, RMSE reported (< 5 cm for valid flag)
- Crown base validation: density filtering, minimum 0.5 m, maximum 90
- Point count validation: sufficient points in measurement zones
- Comprehensive error handling with diagnostic messages

Output Files

Two files are generated in `output_path`:

1. **Classified LAS**: `<filename>_eps<eps>_mpts<mpts>.las`
   - Class 4 = wood points, Class 5 = foliage points
   - Single file replaces the two legacy .txt outputs
2. **Metrics**: `<filename>_metrics.csv` (if calculate_metrics = TRUE)
   - Contains all calculated structural metrics
   - DBH_cm: Always populated when calculation succeeds
   - DBH_RMSE_cm: Fit quality indicator (lower is better, <5 cm is valid)
   - Semicolon-delimited format

Value

Invisibly returns a named list containing:

las_file Character string with full path to classified LAS file (class 4 = wood, class 5 = foliage)

metrics_file Character string with full path to metrics CSV file (NULL if calculate_metrics = FALSE)

metrics data.table containing calculated tree structural metrics (NULL if calculate_metrics = FALSE)

Parameter Selection Guidelines

Voxel Size (dimVox):

- Small trees or fine branches: 1-2 cm
- Medium trees: 2-3 cm
- Large trees: 3-5 cm

DBSCAN Parameters (eps, mpts):

- Densely scanned trees: eps = 1-2, mpts = 4-6
- Sparsely scanned trees: eps = 2-3, mpts = 3-4
- Complex branch structures: lower eps, higher mpts

Cluster Size (N):

- Small trees: N = 500-1000
- Medium trees: N = 1000-2000
- Large trees: N = 2000-5000

Note

Version 2.0 addresses two critical issues identified in field testing:

1. Crown base calculation now robust against noise points near trunk
2. DBH validation extended to 3 m diameter with quality checks

This implementation is fully consistent with the Forest_seg approach, ensuring methodological coherence across the PiC package.

References

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. Agricultural and Forest Meteorology, 262, 434-444. doi:10.1016/j.agrformet.2018.04.008

Pratt, V. (1987). Direct least-squares fitting of algebraic surfaces. ACM SIGGRAPH Computer Graphics, 21(4), 145-152.

Examples

```
## Not run:
# Basic usage with default parameters
result <- SegOne(
  a = "single_tree.xyz",
  filename = "tree_analysis",
  output_path = "~/results"
)

# View calculated metrics
print(result$metrics)

# Advanced usage for large tree
result <- SegOne(
  a = my_point_cloud,
  filename = "large_oak",
  dimVox = 3,           # Larger voxels for large tree
  eps = 2,              # Increased connectivity
  mpts = 6,             # More stringent clustering
  N = 2000,             # Larger minimum cluster size
  R = 35,               # Stricter cylindrical filter
  output_path = "~/tree_metrics",
  voxel_size_canopy = 0.15,
  coverage_method = "linear"
)

## End(Not run)
```

`Voxels` *Voxelize Point Cloud (Optimized)*

Description

Transforms a 3D point cloud into voxel representation with density filtering. Version 3.0 uses optimized data.table operations and native float coordinates for maximum performance and consistency with Forest_seg pipeline.

Key Features:

- Pure data.table operations (10-20x faster than dplyr)
- Native coordinate system (no confusing shifts)
- Memory efficient (no intermediate copies)
- Progress reporting for large datasets
- Consistent with Forest_seg v2.7 architecture

Performance:

- 1M points: ~0.5 seconds
- 10M points: ~3 seconds
- 100M points: ~30 seconds

Usage

```
Voxels(
  a,
  filename = "XXX",
  dimVox = 2,
  th = 2,
  output_path = tempdir(),
  coordinate_precision = "mm"
)
```

Arguments

`a`	Input point cloud. Can be: • File path (.xyz, .txt) • Data frame with x,y,z columns • Matrix with 3+ columns
`filename`	Output file prefix (default = "XXX")
`dimVox`	Voxel dimension in centimeters (default = 2). Typical range: 1-5 cm for forest applications
`th`	Minimum points per voxel for retention (default = 2). Higher values filter more noise but may remove valid sparse regions. Suggested: 1-2 for high density scans, 2-4 for sparse scans

`output_path` Output directory (default = tempdir())

`coordinate_precision`
Decimal places for coordinate rounding. "mm" (3 decimals) or "cm" (2 decimals). Default = "mm"

Details

Voxelization Process:

1. Input validation and coercion to data.table 2. Calculate voxel indices (u, v, w) from coordinates 3. Aggregate points per voxel (fast data.table grouping) 4. Filter by minimum point threshold 5. Export to file

Coordinate System:

Unlike v2.0, this version preserves native coordinates: - No forced positive transformation - No confusing min/max shifts - Voxel indices calculated directly: floor(coordinate / voxel_size) + 1 - Consistent with Forest_seg coordinate handling

Memory Usage:

Approximate memory requirements: - Input points: ~40 bytes/point (x,y,z + overhead) - Voxelized: ~20 bytes/voxel + indices - Peak usage during aggregation: ~2x input size

Value

Invisibly returns list with:

- voxels: data.table with columns u, v, w, N (voxel indices + point count)
- output_file: Path to saved voxel file
- stats: Voxelization statistics

Examples

```
## Not run:
# Basic usage
Voxels(
  a = "forest_scan.xyz",
  filename = "plot_A",
  dimVox = 2,
  th = 2,
  output_path = "results/"
)

# High precision voxelization
result <- Voxels(
  a = point_cloud,
  dimVox = 1,  # 1 cm voxels
  th = 3,      # Aggressive noise filtering
  coordinate_precision = "mm"
)

# Access statistics
print(result$stats)
```

```
## End(Not run)
```

Index