Luke
NATURAL RESOURCES
INSTITUTE FINLAND

Constructing forest trees from laser scanning points - automatized analyses using Matlab and R

Timo P. Pitkänen
Research scientist, Forest inventory and planning
timo.p.pitkanen@luke.fi

Own background in a nutshell

- Worked for Natural Resources Institute Finland (LUKE) since 2016
- Background in geography
- Dealing with various spatial data for forest research
    - Satellite images, point clouds, GIS data…
- Most datasets are either big or HUGE
    - Need for server-based resources and long runs
    - Using both own servers as well as CSC/Puhti
- Gradually learnt to utilize server resources better
    - No other choices for huge datasets

How to make trees from points

- Step 1: scanning point cloud(s) in the forest
    - Using terrestrial laser scanner (TLS), combining several scans together
- Step 2: data preprocessing with own laptop
    - Initial proprietary data format need own processing → laz files → Puhti
- Step 3: converting points to trees (MATLAB)
    - Point cloud → searching for vertical and cylindrical surfaces → stem and branches
- Step 4: further analysis steps (R)
    - R allows a range of point cloud processing steps (and is the most familiar for me)

Luke
NATURAL RESOURCES INSTITUTE FINLAND

Point cloud processing with R

- Principally, point clouds are just “normal” data
    - Special case: x,y,z coordinates → representations of real objects, or parts of them
    - Challenges: how to iterate points into objects, how to make incomplete structures into complete
- Some while ago, ready-made solutions for R were few
    - Researchers developed own codes (a lot of work!)
    - However, things are getting gradually better
- A few currently available packages are presented in the next slides → using them makes the start easier!

R package: lidR (Nov 13, 2021)

Airborne LiDAR *(Light Detection and Ranging) interface for* ***data manipulation and visualization****.* ***Read/write*** *'las' and 'laz' files, computation of* ***metrics*** *in area-based approach,* ***point filtering****, artificial* ***point reduction****,* ***classification*** *from geographic data,* ***normalization****, individual* ***tree segmentation*** *and other manipulations.*

- `readLAS, writeLAS` : reader and writer functions for las data
- `cloud_metrics, voxel_metrics etc.` : basic metrics from data
- `decimate_points` : reduce full cloud into sparser version
- `voxelize_points` : make point cloud into voxels, i.e., evenly sized cubes
- `find_trees` : find locations of trees (several algorithms available)
- `segment_trees` : segmentation (separation) of single trees
- `delineate_crowns` : delineate the area (hull) of each tree

R package: lidaRtRee (Dec 9, 2021)

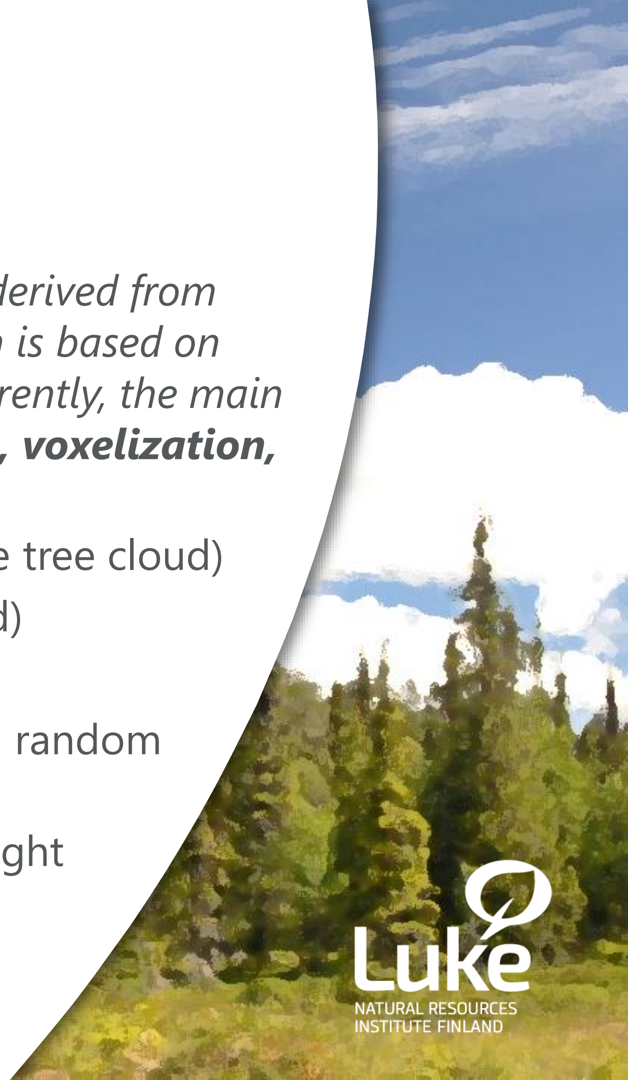
*Provides functions for **forest analysis using airborne laser scanning** (LiDAR remote sensing) data: **tree detection** and **segmentation**; forest **parameters estimation** and **mapping** with the area-based approach. It includes complementary steps for forest mapping: **co-registration of field plots** with LiDAR data; extraction of both **physical** (gaps, edges, trees) and **statistical features** from LiDAR data useful for e.g. habitat suitability modeling; **model calibration** with ground reference, and **maps export**.*

- `clouds_metrics, clouds_tree_metrics` : cloud and tree metrics
- `points2DTM, point2DSM` : computes a digital terrain model (ground) and digital surface model (canopy) → subtraction canopy height model
- `tree_segmentation, tree_extraction` : searching trees and calculating statistics related to them
- `coregistration` : co-registration of field data on trees vs. CHM
- `aba_build_model, aba_predict` : area-based modelling

R package: rTLS (Dec 11, 2021)

*A set of tools to **process and calculate metrics** on point clouds derived from **terrestrial LiDAR** (Light Detection and Ranging; TLS). Its creation is based on key aspects of the TLS **application in forestry and ecology**. Currently, the main routines are based on **filtering, neighboring features of points, voxelization, canopy structure, and the creation of artificial stands**.*

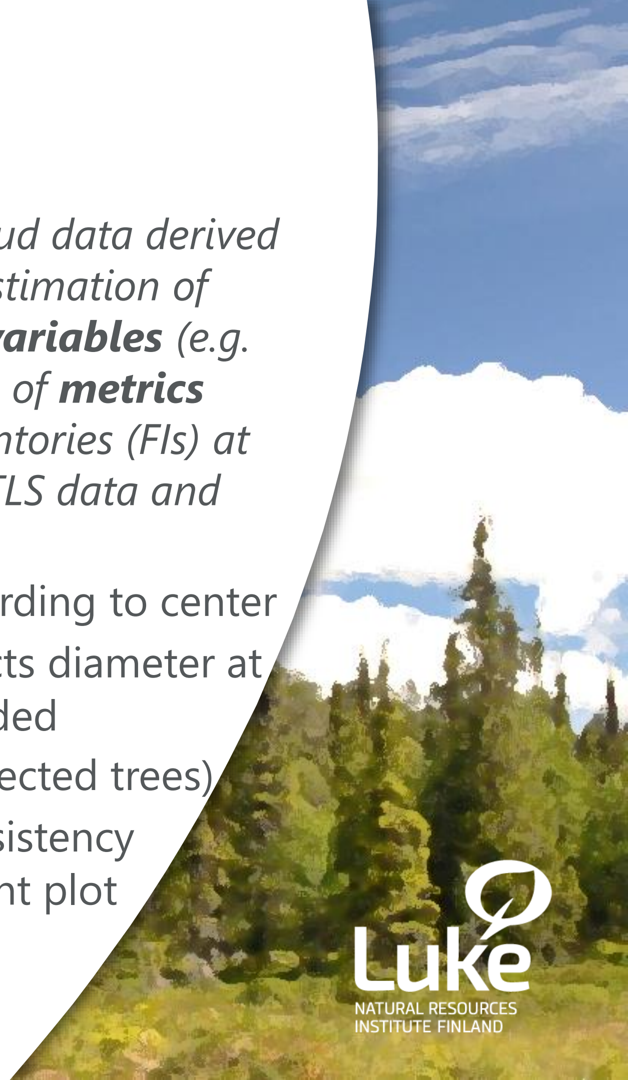
- `tree_metrics` : tree height, crown area and DBH (of a single tree cloud)
- `trunk_volume` : predicts trunk volume (of a single tree cloud)
- `voxels` : voxelization of a point cloud
- `circleRANSAC` : circle fitting (for a slice of a tree stem) using random sample consensus algorithm
- `canopy_structure` : estimation of the canopy structure (height profiles, gap probabilities etc.)



R package: FORTLS (Nov 6, 2021)

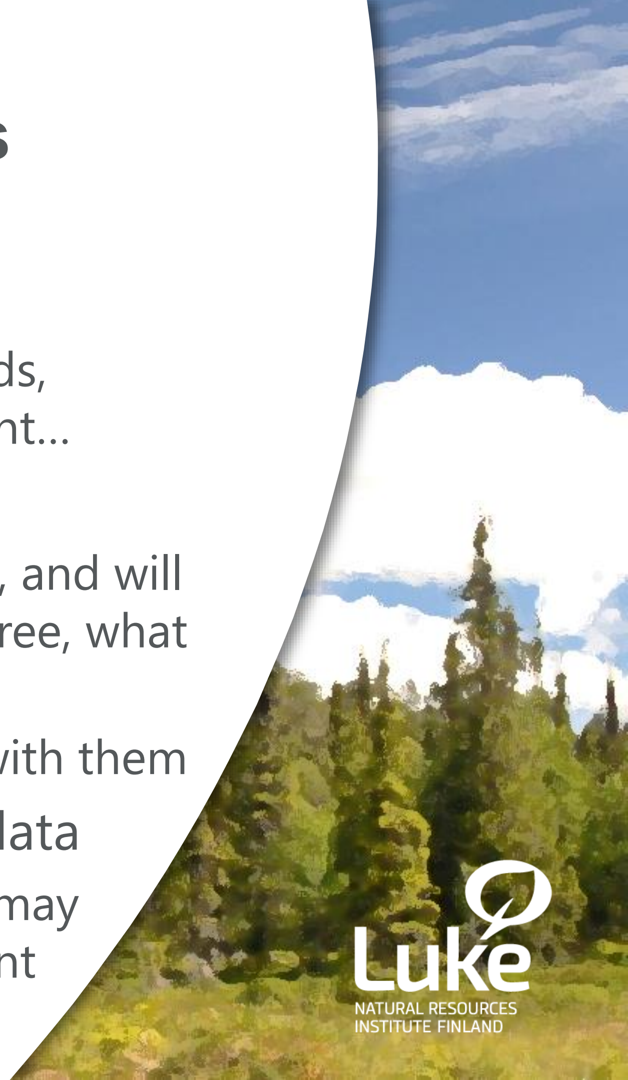
*Process automation of **Terrestrial Laser Scanner (TLS)** point cloud data derived from **single scans**. 'FORTLS' enables (i) **detection of trees** and estimation of **diameter at breast height** (dbh), (ii) estimation of some **stand variables** (e.g. density, basal area, mean and dominant height), (iii) computation of **metrics related to important forest attributes** estimated in Forest Inventories (FIs) at stand level and (iv) **optimization of plot design** for combining TLS data and field measured data.*

- `normalize` : normalizes and decimates the point cloud according to center
- `tree.detection` : detects trees from the point cloud, predicts diameter at h=1.3 m, classifies if the tree is fully visible or partially occluded
- `metrics.variables` : calculates tree-wise metrics (from detected trees)
- `estimation.plot.size` : provides data to evaluate the consistency of stand variables (tree density and basal area /ha) at different plot designs (circular, n trees, angle count) based on simulations



Things to consider with point clouds

- Data origin matters
    - ALS and TLS need different processing methods, single-scan TLS and multi-scan TLS are different…
- Point density and completeness matters
    - Sparse and dense data are ultimately different, and will answer to different questions (e.g. where is a tree, what is stem volume of a tree)
    - Point clouds have occlusions → have to deal with them
- Nice ready-made functions may not fit to your data
    - Initial work for building and testing functions may have been done in totally different environment



Some hints to survive with large point cloud data

- Do not use initial accuracy, but decimate
    - This will only rarely make your results less accurate
- Split the cloud into smaller chunks for easier processing
    - Could these chunks or other parts can also be used for multiprocessing, and combine results in the end?
- Would voxels be enough instead of initial points?
    - Voxels may also help to deal with occlusions
- Consider carefully how to use and organize your data
    - For example, use data.table in R instead of data.frame
    - For storing the data, .laz is a good option

Luke

NATURAL RESOURCES
INSTITUTE FINLAND