Slovak University of Technology in Bratislava
Faculty of Civil Engineering

Reg. No.: SvF-16646-104394

Utilization of aerial laser scanning point cloud data for the study of habitats

Master's thesis

2024 **Bc. Tomáš Homola**

Slovak University of Technology in Bratislava
Faculty of Civil Engineering

Utilization of aerial laser scanning point cloud data for the study of habitats

Master's thesis

Study programme:	Mathematical and Computer Modelling
Study field:	9.1.9. Applied Mathematics
Training workplace:	Department of Mathematics and Constructive Geometry
Supervisor:	Ing. Mgr. Lukáš Tomek, PhD.

Bratislava 2024 **Bc. Tomáš Homola**

Slovak University of Technology in Bratislava
Department of Mathematics and Descriptive Geometry

Faculty of Civil Engineering
Academic year: 2023/2024
Reg. No.: SvF-16646-104394

MASTER THESIS TOPIC

Student:	**Bc. Tomáš Homola**
Student's ID:	104394
Study programme:	Mathematical and Computational Modeling
Study field:	Mathematics
Thesis supervisor:	Ing. Mgr. Lukáš Tomek, PhD.
Head of department:	Ing. Marek Macák, PhD.

Topic: **Utilisation of aerial laser scanning point cloud data for the study of habitats**

Language of thesis: English

Specification of Assignment:

Štúdium biotopov má kľúčový význam pre ochranu a obnovu biodiverzity, pretože rozmanitosť aj početnosť zvierat a rastlín je úzko spätá so štruktúrou vegetácie a súvisiacou mikroklímou.

V rokoch 2017 – 2023 na Slovensku bežal projekt, v ktorom sa prostredníctvom leteckého laserového skenovania snímalo celé územie Slovenska. Jedným z výsledných produktov je klasifikované mračno bodov, ktoré popisuje povrch terénu a objektov na ňom (napríklad budov, či vegetácie). Ako toto mračno využiť na štúdium rôznych biotopov?

Keďže mračno bodov obsahuje trojrozmerné priestorové dáta o vegetácii, je možné získať z neho informácie, ktoré sa nedajú získať zo satelitných snímok či ortofotomozaiky, lebo v nich nie je výšková informácia.
V prvej fáze práce by išlo o zoznámenie sa s tým, ako z mračien bodov vypočítať rôzne metriky, napríklad výšku vegetácie, hustotu vegetácie, či rozptyl výšky vegetácie a podobne. V druhej fáze by sa skúmalo, ako je možné získané metriky využiť pri štúdiu biotopov, napríklad pri ich kategorizácii.

Deadline for submission of Master thesis:	09. 05. 2024
Approval of assignment of Master thesis:	04. 04. 2024
Assignment of Master thesis approved by:	prof. RNDr. Karol Mikula, DrSc. – study programme supervisor

Guidelines

for Diploma Thesis Writing

Preliminary provision

In accordance with the Act No. 131/2002 Coll. on University Education and on Amending and Supplementing Certain Acts as Amended by Later Regulations, a diploma thesis forms part of the studies in every study programme. The defence of a diploma thesis is part of the state exam. A diploma thesis is the final thesis in Master's study programmes. The basis for writing a diploma thesis is the assignment of a diploma thesis.

Thesis Structure

- Title page
- Assignment of the diploma thesis
- Guidelines for writing
- Author's declaration
- Title and abstract in English and Slovak (one page)
- Table of contents with chapter numbers
- List of appendices
- List of abbreviations and symbols
- Text of the thesis (recommended structuring),
  - Introduction
  - Current situation with respect to the thesis research problem
  - Goals of the thesis
  - Author's solution to the problem divided into chapters depending on the character of the thesis
  - Evaluation of the achieved results or of the proposed solutions
  - Conclusion
- Summary (concerns only a thesis written in a language other than Slovak)
- Bibliography
- Appendices (plans, tables, maps, sketches) including a poster of 1000x700 mm

Length and Form

1. The contents and the form of the graduation thesis must be handled in accordance with the regulation of the Ministry of Education, Science, Research and Sport of the Slovak Republic No. 233/2011 Coll., which is used to implement certain provisions of the Act No. 131/2002 Coll., and in accordance with the Methodical regulation No. 56/2011 on Essentials of Final Theses.
2. The required length of the thesis is 30-50 pages. The thesis must be submitted in two bound copies, at least one of them has to be in hardback (not in ring binding) so selected pages cannot be removed. With the approval of the supervisor it is possible to submit extensive graphic appendices in one copy.

3. The author is required to submit the thesis electronically in the information system. The author is responsible for the compliance of the paper and electronic version.
4. After the submission of the thesis in the information system, the author is obligated to deliver a draft of the licence agreement to the faculty. The draft of the licence agreement is generated by academic information system.
5. The recommended font type is Times New Roman, font size 12, and it should stay the same throughout the thesis. The recommended page settings are: spacing – 1.5; left margin – 3.5 cm; right margin – 2 cm; top and bottom margin – 2.5 cm; vertical page orientation; A4 format.
6. Figures and equations should be numbered within individual chapters, e.g. figure No. 3.1 is the figure No. 1 in chapter 3. Equations should be numbered at the right end of the line in parentheses, e.g. (3.1).
7. All the calculations should be clearly arranged, so that their correctness can be verified.
8. In case of borrowed equations, tables or cited parts of another text, the source must be indicated.
9. Quotations of other texts, including electronic materials, must be presented according to STN ISO 690 (01 0197): 2012. *Information and Documentation. Guidelines for Bibliographic References and Citations to Information Resources*.
10. Examples of bibliographical entries:
    BOWLES, J: Foundation analysis and design. Singapore, The McGraw-Hill Companies, Inc. 1997, ISBN 0-07-912247-7.
    MICHALČÁK, O. – ADLER, E.: Výskum stability dunajských hrádzí. In: *Zborník vedeckých prác Stavebnej fakulty SVŠT*, Bratislava, Edičné stredisko SVŠT 1976, ISBN 0-3552-5214.
    ŠÜTTI, J.: Určovanie priestorových posunov stavebných objektov. *Geodetický kartografický obzor*, 35 (77), 1987, č. 2, ISSN 0811-6900.
    Article 18. Technical Cooperation. http://www.lac.uk/iso/tc456 (2013-09-28)
11. The author of the thesis is responsible for its linguistic and terminological correctness.
12. The form of the poster (electronic or printed) is determined by the guarantor of each study programme.
13. A template of the poster can be found on the document server in the university's information system.

....
study programme guarantor's signature

I have taken the provisions of the above-mentioned guidelines into consideration. I am aware of the fact that if my diploma thesis is not written in conformity with these guidelines, it will not be accepted.

In Bratislava ....
student's signature

Declaration

I declare that this thesis has been composed solely by myself under supervision of my supervisor and using the literature stated in Bibliography.

Bratislava 09. 05. 2024

Bc. Tomáš Homola

Acknowledgement

Firstly, I would like to express my gratitude to Ing. Aneta A. Ožvat, Ing. Michal Kollár, PhD., and prof. RNDr. Karol Mikula, DrSc. for their help regarding the natural numerical networks model and the segmented curves of protected Natura 2000 habitats.

I also want to extend a huge thanks to doc. Ing. Tomáš Bacigál, PhD. for his assistance in helping me better understand all the statistics-related topics.

Last but not least, I want to express my sincere gratitude to my supervisor Ing. Mgr. Lukáš Tomek, PhD. for providing me with his guidance, valuable feedback and dedicating his time to me. I am also grateful for the great opportunity to gain a new knowledge outside my regular studies. It has been an honor to work with him once again.

Bratislava 09. 05. 2024

Bc. Tomáš Homola

Abstract

Title: Utilization of aerial laser scanning point cloud data for the study of habitats
Abstract: In this thesis we adopted a workflow for extracting various statistical metrics from classified point cloud data describing vertical vegetation structure. These LiDAR metrics were subsequently exported into multi-band geo-referenced TIFF images for further analysis. Additionally, we used this data to create a multi-dimensional dataset which served as an input for habitat classification using natural numerical networks and linear discriminant analysis. All computations were implemented in Matlab and C++.

Keywords: Natura 2000 habitats, classified point cloud, LiDAR metrics, GeoTIFF, habitat classification

Abstrakt

Názov práce: Využitie mračien bodov z leteckého laserového skenovania na štúdium biotopov
Abstrakt: V tejto práci sme využili postupy na extrakciu rôznych štatistických metrík z klasifikovaných mračien bodov, kde dané metriky popisujú vertikálnu štruktúru vegetácie. Tieto LiDAR-ové metriky boli následne exportované do multikanálových georeferencovaných TIFF obrázkov určených pre ďalšiu analýzu. Tieto údaje sme navyše použili na vytvorenie multidimenzionálneho datasetu, ktorý slúžil ako vstup pre klasifikáciu biotopov pomocou prirodzených numerických sietí a lineárnej diskriminačnej analýzy. Všetky výpočty sme implementovali v Matlabe a v jazyku C++.

Kľúčové slová: Natura 2000 biotopy, klasifikované mračno bodov, LiDAR-ové metriky, GeoTIFF, klasifikácia biotopov

Preface

The focus of this thesis is to explore the potential of Point Cloud (PC) data obtained from areal laser scanning to contribute knowledge about the vertical structure of vegetation, mainly in the context of protected Natura 2000 habitats. To achieve this, we will be utilizing the methods presented in the recent paper [15]. The authors developed a workflow called *Laserfarm* that allows for the extraction of various statistical metrics (LiDAR metrics) calculated from the height of vegetation points. These LiDAR metrics will serve as the foundation for further analysis and exploration of the vegetation structure.

This thesis is structured into 5 chapters. In chapter 1 we provide a motivation on how remote sensing methods can be useful for locating and preserving Natura 2000 habitats as well as a simple outline of our goals we plan to accomplish.

In chapter 2 we provide an overview of the input data we used. Specifically, the classified PC [26] in section 2.1 and the segmented Natura 2000 habitat curves in section 2.2. Next, we outline the steps of the Laserfarm workflow in section 2.5. We provide details on the computational grid construction in section 2.3. Furthermore, we describe the steps involved in the PC processing in section 2.4.

In chapter 3, in which we describe the methods we employed for further analysis of the representative LiDAR metrics. In section 3.1 we provide the basics behind the natural numerical networks model which is currently being used for habitat classification [18]. Additionally, we have chosen a second classification model, namely the linear discriminant analysis, explained in section 3.3. For dimensionality reduction we employed the principal component analysis (section 3.2) and for projection into a lower dimensional space we used the canonical discriminant analysis (section 3.3.1).

In chapter 4 we provide description of our workflow implementations in Matlab and in C++. Additionally, in Figure 4.1 we showcase the UI application we made for visualizing the PC data along line segments specified on the LiDAR metrics raster. This tool was helpful in gaining a better insight into the vegetation structure based on the LiDAR metrics values. To visually analyze the representative metrics, we developed a second UI application, depicted in Figure 4.2.

Our achieved results are presented in chapter 5. In section 5.1 we analyze the LiDAR metrics by plotting the corresponding PC data and showcase how it can help us better understand the LiDAR metrics. In section 5.2 we discuss the advantages and disadvantages of having a finer computational grid. Furthermore, in section 5.3 we present the results obtained from large-scale computations performed on the territorial extent of lots 02 and 03. All the GeoTIFF images from the large-scale computations are available to download at `https://bit.ly/tiff_lots_2_3`. Lastly, in section 5.4, we present the outcomes of the statistical analysis.

Contents

Chapter 1

Introduction

Natura 2000 is a network of protected areas set up by the European Union that plays a crucial role in preserving Europe's most valuable species and habitats [4]. Therefore it is important to map and monitor these regions. However, it can often be challenging to accurately define and keep track of these habitats as it requires specialists to inspect the areas in person.

With the emergence of advanced remote sensing methods such as the Sentinel 2 mission [5], we have access to high-resolution multi-spectral satellite images. Naturally, methods were developed using these images for automated tracking of Natura 2000 habitats, for example the NaturaSat software [19, 18], which provides tools for both segmentation and classification.

Despite its numerous advantages, Sentinel 2 satellite images lack height information. Fortunately, thanks to nation-wide aerial laser scanning surveys using LiDAR technology [27], we have also access to 3D point cloud data describing the surface of Earth and objects on it in great detail. The authors of [15] introduced a workflow for processing large-scale point cloud data into multi-band geo-referenced images containing information on vegetation structure described by several statistical metrics (LiDAR metrics). Utilizing height information from LiDAR data should therefore have the potential to contribute new knowledge for the study of Natura 2000 habitats.

In this thesis we aim to implement parts of the workflow from [15] using Matlab and C++. We plan to create GeoTIFF images containing LiDAR metrics for parts of Western Slovakia. Additionally, we will use some of the methods from [18] to form a multi-dimensional training dataset for classification models. The goal is to test if LiDAR data can provide relevant information about the protected habitats.

Chapter 2

Data description and processing

2.1 Classified Point Cloud

The areal laser scanning (ALS) using a LiDAR[1] sensor mounted on an airplane/drone is a powerful active remote sensing tool that enables us to capture detailed and accurate 3D information about the surface of Earth. It works on the principle of emitting light pulse directed toward the ground and measuring the time it takes to return. Using basic principles of physics, the onboard computer can then calculate the distance d_{GP} between the plane and the spot hit by the light pulse. To calculate the elevation z at the point of impact, the distance d_{GP} is subtracted from the plane's altitude. This information is stored along with the corresponding GPS coordinates as a triplet (x, y, z) representing one point. Additional data can be added to each point such as RGB values or return intensity of the light pulse.

In reality, to scan a wider area, multiple pulses are emitted at various angles. This must be considered when calculating the distance d_{GP}. Additionally, the computer must also factor in any changes in the plane's inertial measurement unit, such as yaw, roll, and pitch. The resulting data set of points is called a *Point Cloud* (PC). Visual representation of ALS is depicted in Figure 2.1.

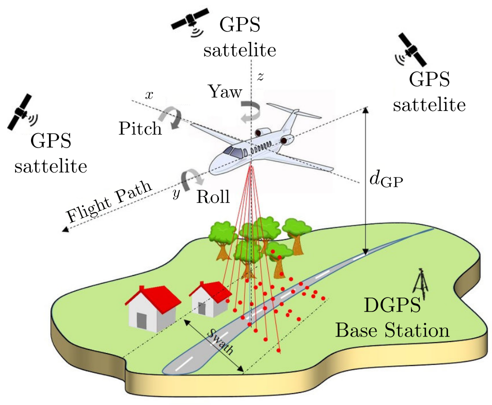


Figure 2.1: Airborne laser scanning, source of the image: [11].

Between the years 2017 and 2023, a large-scale ALS survey of the entire territory of Slovakia was carried out in the project titled ”Airborne Laser Scanning and DTM 5.0” by ÚGKK SR[2]. Its objective was to gather data from ALS conducted by the private sector and create a new Digital Terrain Model (DTM) of Slovakia [26].

[1]An acronym for Light Detection And Ranging.

[2]Geodesy, Cartography and Cadastre Authority of Slovak Republic.

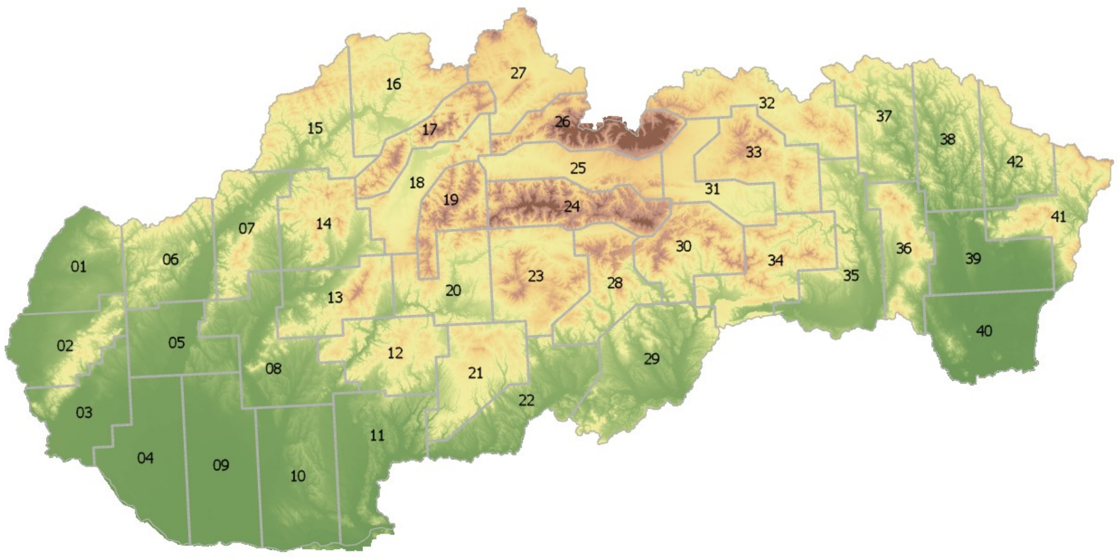


Figure 2.2: Division of Slovak territory into lots, source of the image: [26].

As stated in [26], Slovakia's territory was partitioned into 42 regions called *lots*, as shown in Figure 2.2. The lots' borders (lots' footprints) are described in ESRI Shapefiles (SHP, see [6]), available for download at [28]. The ALS was carried out step-by-step on individual lots, starting in the western regions and gradually moving to the east. The raw PC from ALS was then further processed into a classified PC according to:

1. *Compulsory classification:* 01-Unclassified, and 02-Ground
2. *Optional classification:* 01-Unclassified, 02-Ground, 03-Low vegetation, 04-Medium vegetation, 05-High vegetation, 06-Buildings, 07-Low noise, 09-Water, 11-Road surface, 17-Bridge deck, and 18-High noise

For the purposes of this thesis, we focused on the points classified as ground and vegetation. As described in [26], the DTM and classified PC must meet the following compulsory criteria:

- scanning density at least 5 points per m^2,
- overlap between swaths min. 20% on 95% of their mutual coincidence,
- horizontal and vertical reference systems:
  - S-JTSK(JTSK03)[3] + $\mathrm{H_{Bpv}}$[4],
  - ETRS89-TM34[5] + $\mathsf{h}_{\mathrm{ETRS89}}$,
- absolute vertical accuracy of point cloud at ellipsoidal heights: $|\mathrm{ETRS89} - m_{\mathsf{h}}| \leq 0.15\,\mathrm{m}$,
- absolute positional accuracy of point cloud: $|\mathrm{ETRS89\text{-}TM34} - m_{\mathrm{XY}}| \leq 0.30\,\mathrm{m}$,
- absolute vertical accuracy of DTM: $|\mathsf{h}_{\mathrm{ETRS89}} - m_{\mathrm{H}}| \leq 0.20\,\mathrm{m}$,
- absolute vertical accuracy of DTM: $|\mathrm{Bpv} - m_{\mathrm{H}}| \leq 0.25\,\mathrm{m}$,

[3]System of the Unified Trigonometrical Cadastral Network. In Slovak: Systém Jednotnej Trigonometrickej Siete Katastrálnej.

[4]Baltic vertical reference frame after adjustment. In Slovak: Baltský výškový systém po vyrovnaní.

[5]European Terrestrial Reference System 1989 zone 34.

where m_{h} and m_{H} stand for the measured elevation and m_{XY} for the location. The coordinates of all data utilized in this thesis will be expressed in S-JTSK(JTSK03) - Krovak East-North and $\mathrm{H_{Bpv}}$ reference systems.

Both DTM and classified PC are freely available for the public [28]. To download smaller volume of data, one can use the data export feature of the web application Map Client ZBGIS. The user can specify the export region, for example, by outlining it with a polygon. Once done, the data download link will be sent to the provided user's email address. In order to obtain a larger amount of PC data, it is necessary to fill out the order form and deliver it together with an external drive to the ÚGKK SR offices located either in Bratislava or Prešov [28].

PC data can be stored in LAS (LASer) file format or its compressed version LAZ. When downloading via the Map Client ZBGIS, both file types are available. However, larger PC data are given out only in LAZ format due to their substantial size (almost 10.5 TB in the compressed version). The territorial extent of each LAZ file, also called a *footprint*, is given as a polygon described by X and Y coordinates of its nodal points, see Figure 2.3. All footprints from the same lot are grouped and stored in the corresponding SHP file. These files are available at [28].

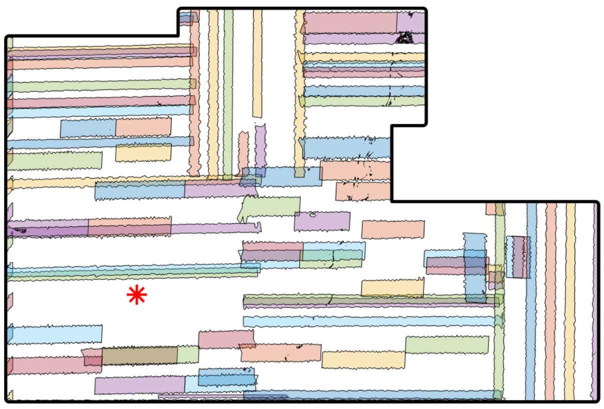


Figure 2.3: Example of some footprints from LOT05, the red point marking the city of Trnava.

The SHP files provided by ÚGKK SR can be loaded using the function `shaperead()` from the Matlab Mapping Toolbox [25]. As the input argument, we provide the path to the selected SHP file. For convenience, the X and Y coordinates of nodal points for each footprint were used to create `polyshape`[6] objects. This significantly simplified the workflow for the PC processing, which is described in section 2.4. Furthermore, we also need to save the LAZ file name of each footprint to access the correct PC data.

To read PC data from a LAZ file, we can use the function `readPointCloud()` from the Matlab Lidar Toolbox [24]. As the first argument, we input the `lasFileReader` object containing metadata for a given LAZ file. Then we can specify what attributes to include, for instance, the points' classification. After the reading is completed, retrieved information is saved in 2 objects of type `pointCloud` and `lidarPointAttributes`, respectively.

Listing 2.1: Code example for handling the SHP and LAZ files in Matlab.

```
% Read SHP file
shapefile = shaperead(shapefileName);
footprints = cell(numel(shapefile), 1);
for i = 1:numel(shapefile)
    footprints{i}.poly = polyshape(shapefile(i).X, shapefile(i).Y);
    footprints{i}.lazFileName = shapefile.filename;
```

[6]Matlab class for handling 2D polygonal shapes.

```
end

% Read LAZ file
lasReader = lasFileReader(lazFileName);
[ptCloud, ptAttributes] = readPointCloud(lasReader,...
                           'Attributes', 'Classification');
xyz = ptCloud.Location; % points' coordinates
class = ptAttributes.Classification;
```

2.2 Habitat curves

In total, we were provided with 111 habitat curves segmented in NaturaSat software which were checked in the field by botanists [18]. These include the following 4 protected Natura 2000 habitats:

(91E0) Alluvial forests with Alnus glutinosa and Fraxinus excelsior (22 curves)

(91F0) Riparian mixed forests of Quercus robur, Ulmus laevis and Ulmus minor, Fraxinus excelsior or Fraxinus angustifolia, along the great rivers (27 curves)

(91G0) Pannonic woods with Quercus petraea and Carpinus betulus (31 curves)

(9110) Luzulo-Fagetum beech forests (31 curves)

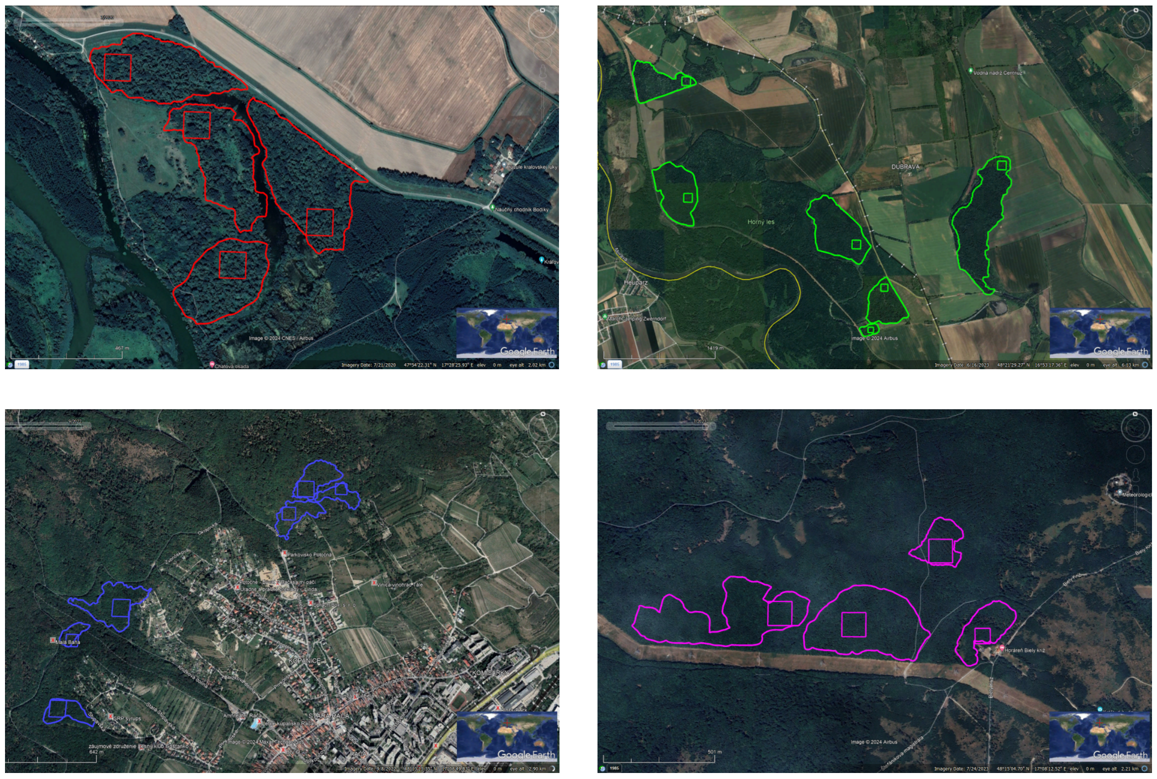


Figure 2.4: Segmented curves of the protected Natura 2000 habitats together with the representative squares visualized in Google Earth. The red curves represent mixed alder-ash alluvial forests located in Bodíky municipal region around the Danube river. The green curves represent riparian mixed oak-elm-ash forests north-west of the village Vysoká pri Morave. The blue curves show the Pannonic oak-hornbeam woods above Bratislava-Rača. Lastly, the magenta curves indicate the acidophilous beech forests near Biely kríž cottage.

For classification purposes, there are also 111 so-called *representative squares* corresponding to each segmented area. These play an important role in the formation of the learning data-set. Aside from the habitat curves, we were provided with 22 additional representative squares for monoculture tree plantations in the vicinity of the 91E0 habitats around the Danube river.

Both the habitats' curves and the representative squares were supplied as KML files. It is an XML file format commonly used to display geographic data, for instance, in Google Earth [10]. Similarly to the footprint curves mentioned in section 2.1, we also stored the habitats' curves as `polyshape` objects.

2.2.1 Conversion of coordinates

In order to work with habitat curves and classified PC, both must use the same coordinate system. The nodal points of each habitat curve are expressed in GPS coordinates, meaning as longitude λ and latitude ϕ. It is equivalent to saying they are described using the geodetic coordinates $(\lambda, \phi, \mathsf{h})$ within the global WGS84[7] datum. The ellipsoidal height h is assumed to be zero.

Points from classified PCs are expressed in the projected grid coordinates within the local S-JTSK(JTSK03) datum. Because we have tens of millions of these points, it is more practical to convert the habitat curves to these grid coordinates. This conversion of coordinates is carried out in 5 steps, as shown in Figure 2.5. The steps (T1) and (T4) involve changes between geodetic and geocentric coordinates. The steps (T2) and (T3) encompass datum transformations using the Helmert transformation [21]. The final step (T5) is a forward projection from geodetic to grid coordinates of our choice. For more details see section 2.3 *Coordinate system transformation* in [12].

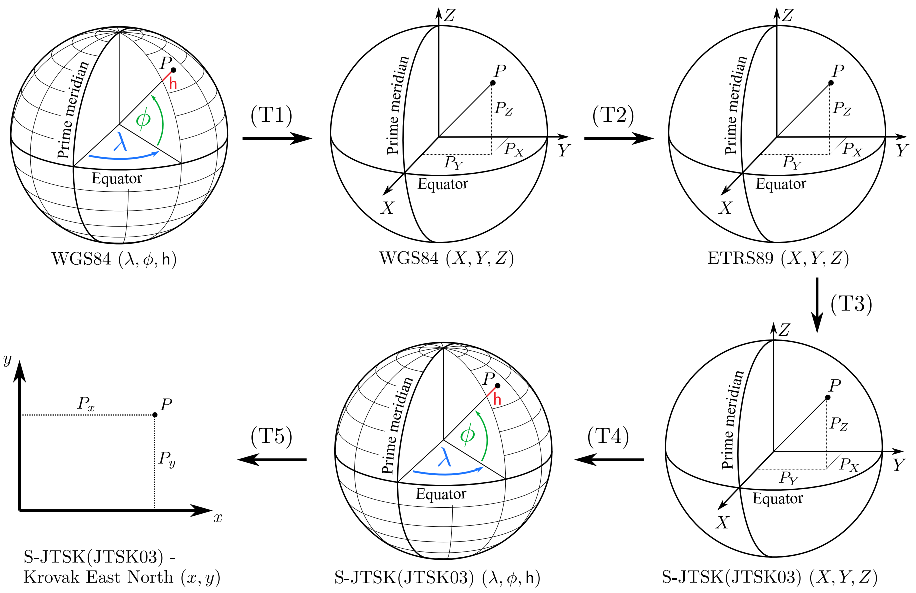


Figure 2.5: Transformation from WGS84 to S-JTSK(JTSK03), source of the image: [12].

2.3 Computational grid

We created 2 different methods for establishing the computational domain denoted as Ω. For both the user must define the desired pixel size h [m]. In case of an area chosen via the Map Client ZBGIS, we take advantage of having the DTM. Let us denote $\mathrm{nx}_{\mathrm{terrain}}$ and

[7]World Geodetic System 1984.

$\mathrm{ny}_{\mathrm{terrain}}$ as the dimensions of the DTM where each pixel represents $1 \times 1\,\mathrm{m}$ square. Then the dimensions of the domain nx_Ω and ny_Ω will be calculated as

$$\mathrm{nx}_\Omega = \left\lfloor \frac{\mathrm{nx}_{\mathrm{terrain}}}{h} \right\rfloor \quad \text{and} \quad \mathrm{ny}_\Omega = \left\lfloor \frac{\mathrm{ny}_{\mathrm{terrain}}}{h} \right\rfloor .$$

Unless we choose $h = 1$, this method in general yields a grid with a slightly smaller real world size compared to the extent of the DTM. This simpler method was used only in the first testing computations.

The second approach is more universal, as the area of interest can be specified by any generic curve denoted as γ. It can be either a segmented habitat boundary or a general area. Let us define the width and the height of the bounding box of γ (a smallest rectangle which completely encloses a curve, see Figure 2.6) as

$$W_\gamma = |x_{\max} - x_{\min}| \quad \text{and} \quad H_\gamma = |y_{\max} - y_{\min}|.$$

By $x_{\max}$ and $x_{\min}$ we denote the maximum and the minimum of γ points' x coordinates, similarly for y coordinates. We will define the computational domain Ω by offsetting the bounding box of γ outwards, such that the width W_Ω and the height H_Ω must be divisible by the pixel size h. This is ensured by having

$$W_\Omega = \left\lceil \frac{W_\gamma}{h} \right\rceil h + n_p\, h \quad \text{and} \quad H_\Omega = \left\lceil \frac{H_\gamma}{h} \right\rceil h + n_p\, h,$$

where we added another user defined padding parameter n_p to have the ability to include a larger area around the curve γ, see Figure 2.7. We can calculate the unknown offset lengths Δ_x and Δ_y (see Figure 2.6) as follows:

$$\Delta_x = \frac{W_\Omega - W_\gamma}{2} \quad \text{and} \quad \Delta_y = \frac{H_\Omega - H_\gamma}{2}.$$

Thus we can define Ω by adding or subtracting the corresponding offsets from all four vertices of the bounding box of γ. The dimensions of the domain are obtained simply as $\mathrm{nx}_\Omega = W_\Omega / h$ and $\mathrm{ny}_\Omega = H_\Omega / h$.

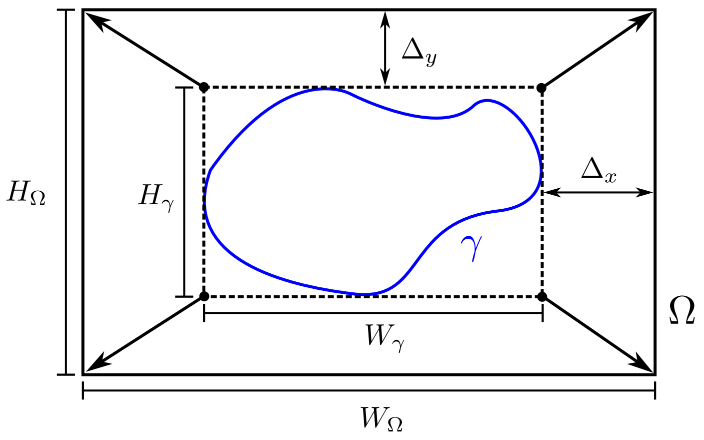


Figure 2.6: Construction of Ω by offsetting the bounding box (dashed rectangle) of γ.

Furthermore, we separated all grid pixels into 2 categories: inside and outside the curve γ. If a pixel center belongs to the interior of γ then this pixel is labeled as an *internal* pixel, otherwise as an *external* pixel. To find all the internal pixels we utilized the Matlab function `inpolygon()`. This information is important in the formation of the learning data-set because we are interested only in the LiDAR metrics within the representative squares. It also enables us to speed up the computation process by skipping all external pixels.

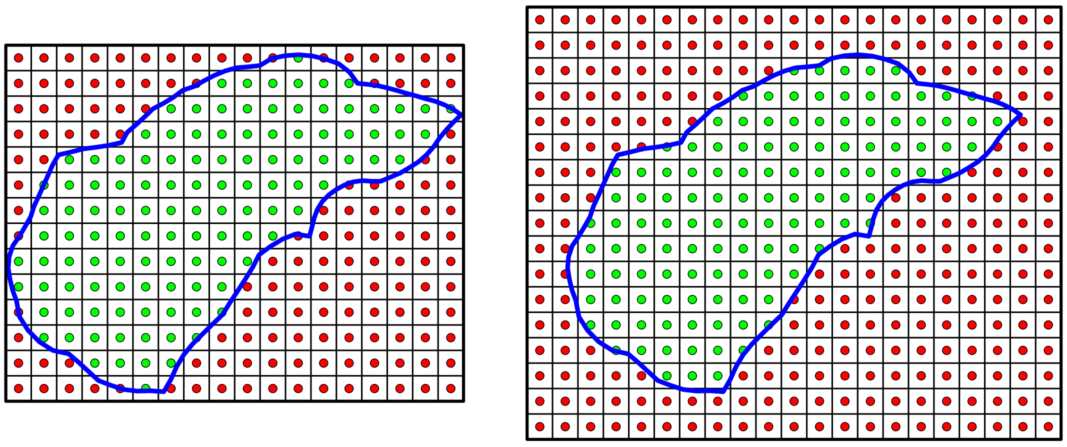


Figure 2.7: Comparison of 2 grids created with different values of n_p around one of the habitat curves (blue). For the grid on the left we used $n_p = 0$. The one on the right used $n_p = 3$. Internal pixels are colored in green, external in red color.

Additionally, we will denote by Ω_n the grid of pixels for PC normalization (see section 2.4.3) with the fixed size of $1 \times 1\,\mathrm{m}$. By Ω_h we understand the grid of pixels for LiDAR metrics computation.

2.4 Point cloud processing

2.4.1 LAZ file search

For the first approach mentioned in the previous section 2.3, this part is very straightforward. All the LAZ files for the selected area can be simply downloaded using the link sent to the user's email address as mentioned in section 2.1. For the second approach, we must select the LAZ files ourselves.

Since we have the information about each LAZ file's territorial extent in the form of the footprint curve, we can simply check for the intersection between the domain Ω and each footprint curve. If they intersect, we will include the corresponding LAZ file. As we mentioned previously, by storing all curves as `polyshape` objects, we can simply use the Matlab `intersect()` function.

To make this more computationally efficient, we first check for the intersection between the domain Ω and lots' boundaries. This way we don't need to check for all the available footprint curves. We only iterate through footprints of those lots where Ω is located.

2.4.2 Selection of points

This step is identical for both approaches. Firstly, we filter out all the points outside of the domain Ω. Thus we can significantly reduce memory requirements because the majority of the points usually fall outside of the domain Ω. Secondly, we select only points classified either as ground or vegetation.

2.4.3 Point cloud normalization

PC data typically have the z coordinates (height) as the absolute height with respect to some vertical reference system. We need it, however, as height relative to the ground. The normalization[8] can be done in 2 ways. We can use the DTM because it contains information about the terrain height. For each DTM pixel, we subtract its height value from z coordinate

[8]In this context it means to eliminate the influence of the terrain.

of all points within this pixel. It can, however, lead to negative z values in places where points are lower than the interpolated height from the DTM. In such case, we set all negative values to zero.

The other method, as was used in [15], is subtracting the minimum height of all the points within a 1×1 m pixel. Advantages of this approach include having no negative height values as well as not being dependent on the availability of the DTM. We implemented both methods, but chose to use the latter as it does not require any additional data.

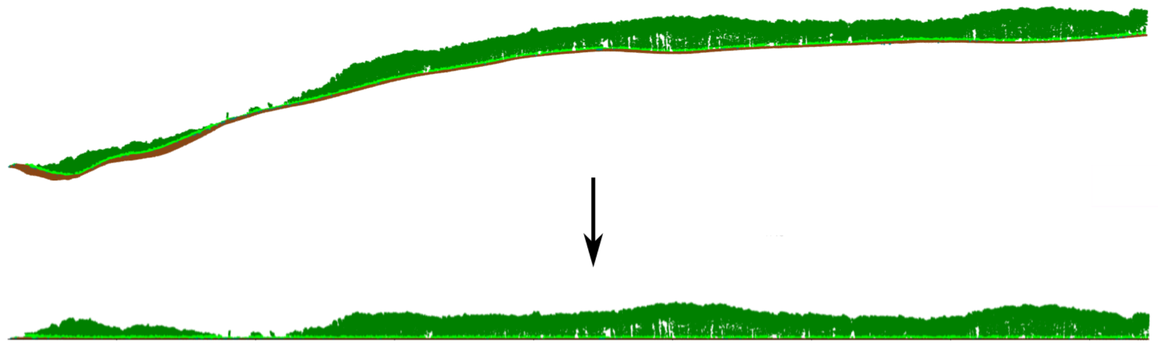


Figure 2.8: Comparison of the PC data before and after the normalization.

2.5 LiDAR metrics

The utilization of a classified PC to describe the vegetation structure of selected Natura 2000 habitats was inspired by the paper [15]. Its authors introduced an effective workflow for processing large-scale PC data into ecologically meaningful information in the form of geo-referenced raster images. These images contain various statistical metrics, e.g. maximal height of vegetation or vegetation density within different height layers, see [15]. In total, they extracted 25 LiDAR metrics from the PCs covering the whole country of Netherlands.

As is described in [15], their workflow consists of 4 steps. First, the original LAZ files are split into multiple smaller files in the process called *re-tiling*. This allows the workflow to be easily parallelized. In the second step, the *normalization* of re-tiled data includes subtracting the ground to eliminate the influence of the terrain. Thus the z values represent the true height of vegetation above the ground. The last 2 steps deal with *features extraction* from the normalized data and the subsequent *export* into GeoTIFF images.

For our purposes we adopted this workflow and implemented parts of it in Matlab and C++ with emphasis on the segmented habitat curves. We used 23 (of the 25 original) LiDAR metrics described in Tables 2.1, 2.2, and 2.3. Similarly to [15], one of the outputs of our workflow also includes multi-band raster images with LiDAR metrics, see Figure 2.9. Additionally, we used the LiDAR metrics to create a multi-dimensional feature space as input for 2 classification models, see sections 3.1 and 3.3.

Table 2.1: Ecosystem height metrics.

Index	Abbreviation	Metric name	Description
1	Hmax	Maximum vegetation height	Maximum of normalized z coordinates in pixel.
2	Hmean	Mean of vegetation height	Mean of normalized z coordinates in pixel.
3	Hmedian	Median of vegetation height	Median of normalized z coordinates in pixel.
4-6	Hp25, Hp75 Hp95	25th, 75th, and 95th percentile of vegetation height	Corresponding percentiles of normalized z coordinates in pixel.

Table 2.2: Ecosystem cover metrics.

Index	Abbreviation	Metric name	Description
7	PPR	Pulse penetration ratio	Ratio of ground points to all points in pixel.
8	DAM_z	Canopy cover	Number of laser returns above mean height in pixel.
9-17	BR_below_z2, BR_z1_z2, BR_above_z1	Vegetation points density within height layers (<1 m, 1-2 m, 2-3 m, >3 m, 3-4 m, 4-5 m, <5 m, 5-20 m, >20 m)	Ratios of vegetation points within specified height layer to all vegetation points in pixel.

Table 2.3: Ecosystem structural complexity metrics.

Index	Abbreviation	Metric name	Description
18	Coef_var_z	Coefficient of variation	Ratio of standard deviation and mean of normalized z coordinates in pixel.
19	Hkurt	Kurtosis of vegetation height	Kurtosis of normalized z coordinates in pixel.
20	Hskew	Skewness of vegetation height	Skewness of normalized z coordinates in pixel.
21	Hstd	Standard deviation of vegetation height	Standard deviation of normalized z coordinates in pixel.
22	Hvar	Variance of vegetation height	Variance of normalized z coordinates in pixel.
23	Shannon	Shannon index	Equal to $-\sum p_i \ln p_i$ where p_i is the proportion of the vegetation points within 0.5 m height layers in pixel.

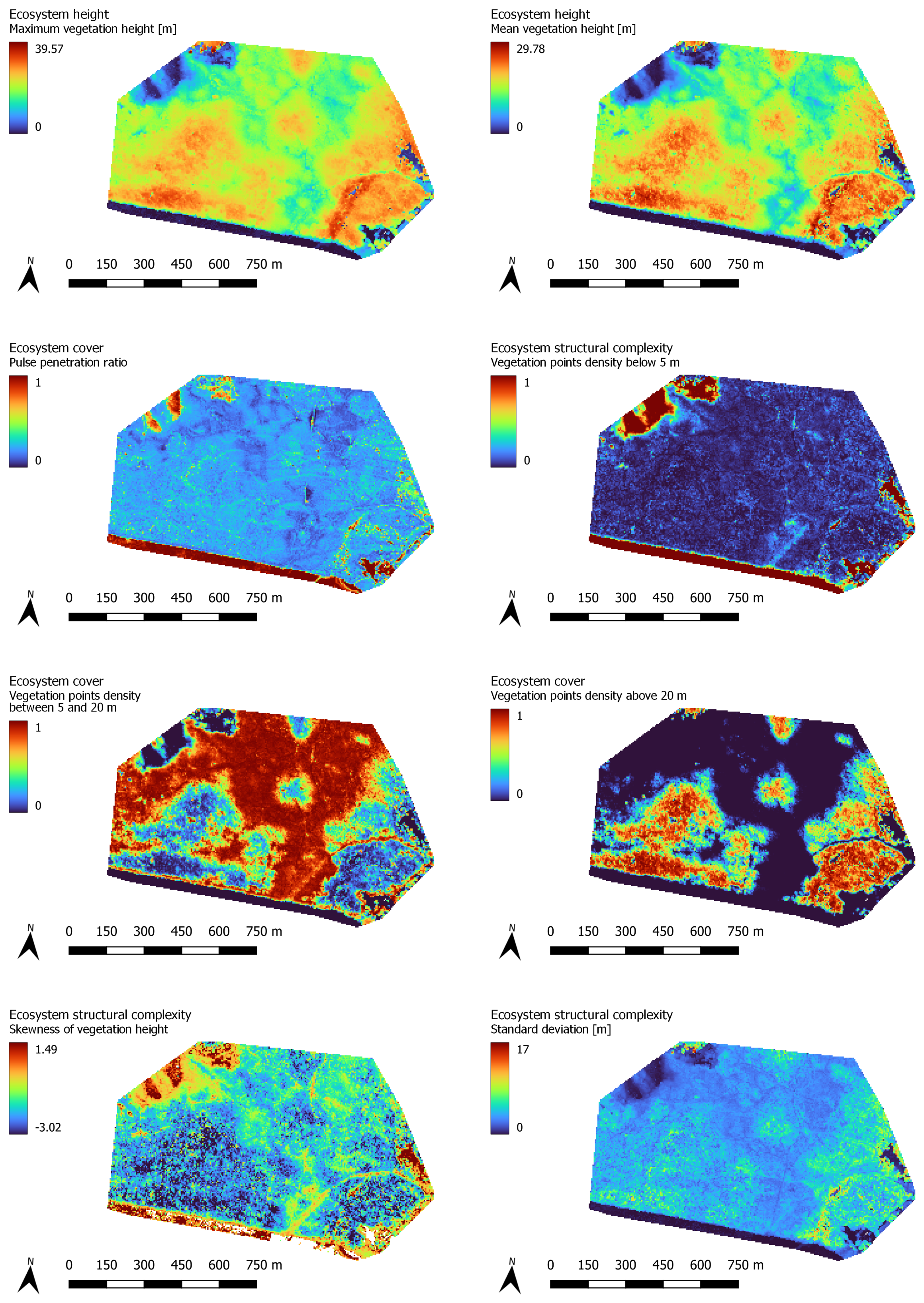

Ecosystem height
Maximum vegetation height [m]
39.57
0
Ecosystem height
Mean vegetation height [m]
29.78
0
Ecosystem cover
Pulse penetration ratio
1
0
Ecosystem structural complexity
Vegetation points density below 5 m
1
0
Ecosystem cover
Vegetation points density between 5 and 20 m
1
0
Ecosystem cover
Vegetation points density above 20 m
1
0
Ecosystem structural complexity
Skewness of vegetation height
1.49
-3.02
Ecosystem structural complexity
Standard deviation [m]
17
0
N
0 150 300 450 600 750 m

Figure 2.9: Georeferenced raster images ($5 \times 5\,$m per pixel) showing some of the LiDAR metrics computed for the area near Biely kríž cottage.

Chapter 3

Statistical methods for further analysis

In this chapter we describe the methods for analyzing the dataset derived from the LiDAR metrics. The goal is to determine if we can differentiate individual habitat types and artificial monocultures using LiDAR data. Section 3.1 provides the basics behind the NatNets model, currently implemented in NaturaSat software for habitats classification [18]. As an alternative, we use also the Linear Discriminant Analysis, which is explained in section 3.3. For reducing the number of predictors, we utilized the Principal Component Analysis described in section 3.2. To visualize the dataset in a lower dimension, we used the Canonical Discriminant Analysis described in section 3.3.1.

3.1 Natural numerical networks

Natural Numerical Networks (NatNets) is a deep learning method developed in order to perform automated classification of protected Natura 2000 habitats [18]. A deep learning network typically contains multiple hidden layers where all the computations occur. In the case of the NatNets, these layers are defined by the discretization of the *nonlinear forward-backward diffusion equations* on a complete graph. So one layer is represented as one time step of the numerical scheme.

The basic idea of this algorithm is to utilize the clustering effect of the forward diffusion. This means that points with similar features in the feature space are brought closer together, forming clusters. To classify a new observation, we look at which cluster it was assigned to and evaluate it with a relevancy coefficient. The number of clusters N_C is given by the number of habitat types we assume. For the purposes of this thesis, we will explain only the core ideas behind this model.

Let $G = \big(V(G), E(G)\big)$ denote a complete graph, meaning all vertices are connected by edges. We will describe the location $X(v,t) = \big(x_1(v,t), \ldots, x_p(v,t)\big) \in \mathbb{R}^p$ of a vertex $v \in V(G)$ at time $t \in [0,T]$ in the p-dimensional feature space by the function $X\colon G \times [0,T] \to \mathbb{R}^p$. The diffusion process can be formulated by the following PDE

$$\partial_t X(v,t) = \nabla \cdot \big(g \, \nabla X(v,t)\big), \tag{3.1}$$

coupled with the initial conditions $X(v,0) = X^0(v)$ for all $v \in V(G)$. The basic definition of the diffusion coefficient $g = g\,(e_{uv})$ for each edge e_{uv} connecting vertices u and v is as follows

$$g\,(e_{uv}) = \varepsilon\,(e_{uv}) \, \frac{1}{1 + KL^2\,(e_{uv})},$$

where $L\,(e_{uv})$ denotes the Euclidean distance between points $X(u,\cdot)$ and $X(v,\cdot)$, the coefficient $K \geq 0$ is controlling the influence of the length $L\,(e_{uv})$ and the parameter $\varepsilon\,(e_{uv})$ for switching between forward and backward diffusion.

The diffusion coefficient g is further modified, for example by introducing multiple parameters K_i or reducing the influence of observations outside the so-called "δ-diffusion neighborhood", see [18].

For the numerical discretization of the model (3.1) we first approximate the diffusion flux to or from the vertex v along the edge e_{uv} as

$$\mathcal{F}_i\left(v, e_{uv}, t\right) = g\left(e_{uv}\right)\big(x_i(u,t) - x_i(v,t)\big),$$

where $x_i(u,t)$ denotes the i-th coordinate of $X(v,t)$. This can be understood as an analogy to Fick's law. If the value of $\mathcal{F}_i$ is positive we regard it as a diffusion *inflow* to the vertex v. On the other side, if $\mathcal{F}_i$ is negative, it means there is a diffusion *outflow* from the vertex v.

Next, we can describe the balance of the diffusion fluxes by summing up contributions along all edges connected to the vertex v

$$\partial_t\, x_i(v,t) = \sum_{e_{uv}} \mathcal{F}_i\left(v, e_{uv}, t\right) = \sum_{e_{uv}} g\left(e_{uv}\right)\big(x_i(u,t) - x_i(v,t)\big), \tag{3.2}$$

where the expression on the right is the "graph-Laplacian" from graph theory [7]. Last step is the time discretization by the finite difference method

$$\partial_t\, x_i(v,t) \approx \frac{x_i^m(v) - x_i^{m-1}(v)}{\tau} \tag{3.3}$$

where τ is the time step and $x_i^m(v) = x_i(v, t_m)$. By plugging (3.3) into (3.2) and some rearranging we obtain in each time step the following systems of linear equations

$$\left(1 + \tau \sum_{e_{uv}} g_{e_{uv}}^{m-1}\right) x_i^m(v) - \tau \sum_{e_{uv}} g_{e_{uv}}^{m-1} x_i^m(u) = x_i^{m-1}(v)\,, \quad \forall v \in V(G)\,, \quad i = 1, \ldots, p$$

which we solve for each coordinate x_i separately. As mentioned before, these systems represent the dynamics of the NatNets. For further information see [18].

To determine the relevancy of classification of the new observation w, the authors of [18] introduced the so-called *relevancy coefficient* $R\,(w)$ which is based on distances to the cluster centroids $\mathcal{C}_i$. These can be defined simply as the arithmetic mean of all points from the cluster C_i at the final time step T

$$\mathcal{C}_i = \frac{1}{N_{C_i}} \sum_{v \in C_i} X(v,T)\,, \quad i = 1, \ldots, N_C.$$

After the network dynamics is stopped, we look for the closest point of the network (from the cluster C_a) to the new observation w. If its distance is smaller than some predefined tolerance, we assign w to the cluster C_a. Otherwise it is regarded as an outlier with zero relevancy. Next, we calculate the distance between the original position $X(w,0)$ and the centroid $\mathcal{C}_a$

$$d_a(w) = |X(w,0) - \mathcal{C}_a|,$$

and the average distance to the other centroids

$$D_a(w) = \frac{1}{N_C - 1} \sum_{\substack{i=1 \\ i \neq a}}^{N_C} |X(w,0) - \mathcal{C}_i|.$$

Using the distances d_a and D_a we define the quantity

$$R_a^{\circ}(w) = 1 - \frac{d_a(w)}{d_a(w) + D_a(w)} \in [0,1]\,.$$

As mentioned in [18], if the position $X(w,0)$ is near the centroid $\mathcal{C}_a$, then the distance d_a is almost 0 and the value of $R_a^{\circ}(w)$ is close to 1, thus indicating high relevancy. If the quantity $R_a^{\circ}(w)$ is less than 0.5, then the relevancy of classification should be reduced. For this purpose the authors used the logistic function

$$\mathcal{L}\left(x\right) = \frac{1}{1 + \mathrm{e}^{\lambda(0.5-x)}},$$

where $\lambda = 12$. The relevancy coefficient is then defined as

$$R_a\left(w\right) = \frac{\mathcal{L}\big(R_a^{\circ}(w)\big) - \mathcal{L}\left(0\right)}{\mathcal{L}\left(1\right) - \mathcal{L}\left(0\right)}.$$

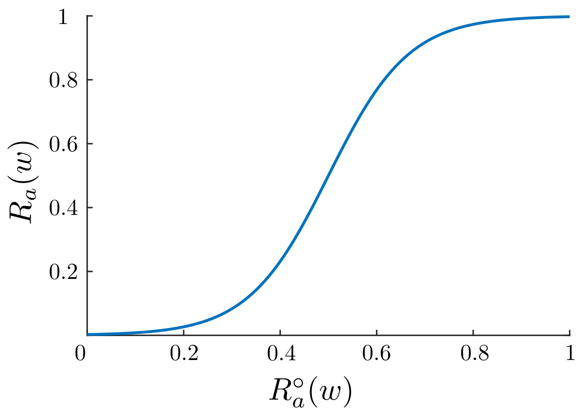


Figure 3.1: The relevancy coefficient $R\left(w\right)$ plotted as a function of $R_a^{\circ}(w)$.

3.2 Principal Component Analysis

Principal Component Analysis (PCA) is an unsupervised learning method used to reduce the dimensionality of multivariate data-set. It does so by finding the directions (principal components) that capture the maximum variance. To reduce the dimension we project the original data into the sub-space with its basis composed of the first two or three principal components.

Let $\mathbf{X}$ denote the $n \times p$ matrix containing coordinates of n observations in the original p-dimensional feature space. We assume the coordinates to be standardized, meaning each column of $\mathbf{X}$ has zero mean and is scaled to have unit standard deviation. This is important because PCA favors the features with greater variances and this can lead to skewed results. The standardization ensures that all features contribute proportionally to finding the principal components.

To simplify, let us assume only the first principal component represented by the axis Z in the direction of the greatest variance. As mentioned in [1], this axis can be expressed by a linear combination of the original feature space axes $X_1, \ldots, X_p$ as

$$Z = v_1 X_1 + \cdots + v_p X_p$$

where $\mathbf{v} = (v_1, \ldots, v_p)^{\mathsf{T}}$ is called *the loadings vector* with unit norm $\|\mathbf{v}\| = 1$. Coordinates of observations on the axis Z, denoted as $\mathbf{z}$, will be calculated as $\mathbf{z} = \mathbf{X}\mathbf{v}$. As mentioned before, the goal is to maximize the variance $\mathrm{var}[Z]$ which we can express as

$$\mathrm{var}[Z] = \mathbb{E}\left[Z^2\right] = \frac{1}{n}\mathbf{z}^\intercal\mathbf{z} = \frac{1}{n}\mathbf{v}^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{v} = \mathbf{v}^\intercal\left(\frac{1}{n}\mathbf{X}^\intercal\mathbf{X}\right)\mathbf{v} = \mathbf{v}^\intercal\mathbf{\Sigma}\mathbf{v}$$

where $\mathbf{\Sigma}$ is the symmetric covariance matrix of our data $\mathbf{X}$. To find the vector $\mathbf{v}_1$ corresponding to the first principal component, we can solve the following optimization problem

$$\mathbf{v}_1 = \arg\max_{\mathbf{v}} \mathbf{v}^\intercal\mathbf{\Sigma}\mathbf{v} \quad \text{s.t.} \quad \mathbf{v}^\intercal\mathbf{v} = 1. \tag{3.4}$$

A simple way is to use the Lagrange multipliers method where we define the Lagrange function as $L(\mathbf{v},\lambda) = \mathbf{v}^\intercal\mathbf{\Sigma}\mathbf{v} - \lambda\left(\mathbf{v}^\intercal\mathbf{v} - 1\right)$. Then we differentiate with respect to $\mathbf{v}$ and set the derivative equal to zero.

$$\nabla_{\mathbf{v}} L(\mathbf{v},\lambda) = 2\mathbf{\Sigma}\mathbf{v} - 2\lambda\mathbf{v} = 0 \implies \mathbf{\Sigma}\mathbf{v}_1 = \lambda_1\mathbf{v}_1. \tag{3.5}$$

As we can see from (3.5), the direction of the greatest variance is given by the eigenvector $\mathbf{v}_1$ of the matrix $\mathbf{\Sigma}$ corresponding to its largest eigenvalue λ_1. Subsequently, the direction of the second greatest variance will be given by the eigenvector $\mathbf{v}_2$ corresponding to the second largest eigenvalue λ_2, and so on. To find all principal components at once, we can diagonalize the matrix $\mathbf{\Sigma}$ to obtain its eigenvalue decomposition [22]

$$\mathbf{\Sigma} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\intercal$$

where $\mathbf{V}$ is the orthogonal $p \times p$ matrix whose columns correspond to the eigenvectors. This is the matrix that transforms the data $\mathbf{X}$ to a new coordinate system represented by the eigenvectors $\mathbf{v}_i$. The projected coordinates, also called *scores*, will be calculated as $\mathbf{Z} = \mathbf{X}\mathbf{V}$. Eigenvalues are given by the diagonal matrix $\mathbf{\Lambda} = \mathrm{diag}\left(\lambda_1, \ldots, \lambda_p\right)$, and they explain the variance of the corresponding principal components [22]

$$\mathrm{var}[\mathbf{z}_i] = \frac{1}{n}\mathbf{z}_i^\intercal\mathbf{z}_i = \frac{1}{n}\mathbf{v}_i^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{v}_i = \mathbf{v}_i^\intercal\mathbf{\Sigma}\mathbf{v}_i = \mathbf{v}_i^\intercal\lambda_i\mathbf{v}_i = \lambda_i\mathbf{v}_i^\intercal\mathbf{v}_i = \lambda_i.$$

There is no straight-forward method to tell how many principal components to retain. To help us decide we can use a *scree plot* (see Figure 3.2) which graphically shows, for instance the Proportion of Variance Explained

$$\mathrm{PVE}\left(i\right) = \frac{\lambda_i}{\sum\limits_{j=1}^{p}\lambda_j}$$

for each principal component. Then by applying the "elbow method" we look for the component where the curve flattens out which indicates that subsequent components explain much smaller proportion of variance [1]. Alternatively, we can set a variance threshold (e.g. 90 %) and choose the number of components that cumulatively sum up to that desired threshold.

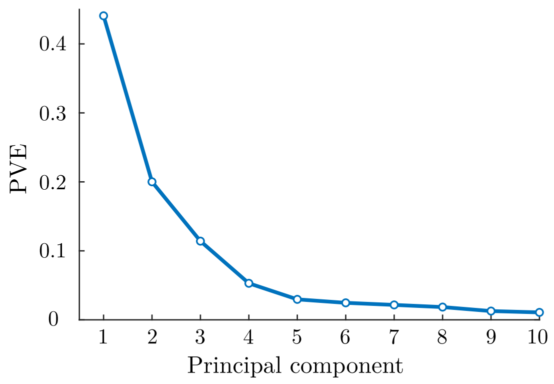


Figure 3.2: Scree plot for the first 10 principal components.

3.3 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) belongs to the family of generative classifiers. In contrast to discriminative methods, they take an indirect approach for modeling the conditional probability $\Pr\left[Y = k | \boldsymbol{X} = \mathbf{x}\right]$, meaning the probability of assigning the observation $\mathbf{x}$ to the k-th class. By Y we denote the response variable (in our case 5 classes - 4 habitats and monocultures) and by $\boldsymbol{X}$ the vector of predictors. Generative methods try to learn the probability distributions f_k of predictors for each value of the response Y. Then the posterior probability is computed using the Bayes' theorem as

$$\Pr\left[Y = k | \boldsymbol{X} = \mathbf{x}\right] = \frac{\pi_k f_k(\mathbf{x})}{\sum\limits_{j=1}^{K} \pi_j f_j(\mathbf{x})} \tag{3.6}$$

where $\pi_k = \Pr\left[Y = k\right]$ is called the prior probability of class k. It can be easily estimated as $\frac{n_k}{n}$ where n_k denotes the number of observations belonging to the class k. In general, probability densities f_k may be more difficult to estimate.

For this reason, LDA makes an assumption about joint normal probability distribution for each class k, meaning $\boldsymbol{X} \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$. The probability density function is thus given as

$$f_k(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{p}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\left(\mathbf{x} - \boldsymbol{\mu}_k\right)^{\intercal} \boldsymbol{\Sigma}^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_k\right)\right) \tag{3.7}$$

where $\boldsymbol{\mu}_k = \mathbb{E}\left[\boldsymbol{X} | Y = k\right]$ is the vector of mean values for the class k. The covariance $p \times p$ matrix $\boldsymbol{\Sigma}$ is assumed to be equal for all classes. After plugging (3.7) into (3.6) we get

$$\Pr\left[Y = k | \boldsymbol{X} = \mathbf{x}\right] = \frac{\pi_k \cancel{\frac{1}{(2\pi)^{\frac{p}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}}} \exp\left(-\frac{1}{2}\left(\mathbf{x} - \boldsymbol{\mu}_k\right)^{\intercal} \boldsymbol{\Sigma}^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_k\right)\right)}{\sum\limits_{j=1}^{K} \pi_j \cancel{\frac{1}{(2\pi)^{\frac{p}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^{\intercal} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)\right)} \tag{3.8}$$

Note that the denominator in (3.8) remains constant across all classes so we can focus on the numerator only. As we classify the observation $\mathbf{x}$ according to the highest posterior probability, we can simplify this calculation by applying the logarithm. Because the logarithm is a strictly increasing function, applying it to (3.8) will preserve the order of the posterior probabilities.

$$\ln \Pr\left[Y = k | \boldsymbol{X} = \mathbf{x}\right] = \ln \pi_k \cancel{- \frac{1}{2} \mathbf{x}^{\intercal} \boldsymbol{\Sigma}^{-1} \mathbf{x}} + \mathbf{x}^{\intercal} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2} \boldsymbol{\mu}_k^{\intercal} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k \cancel{- \ln \sum_{j=1}^{K} (\dots)} \tag{3.9}$$

Furthermore, we can omit the terms in (3.9) that are independent of k because adding constant terms does not affect the classification outcome. This leads us to the so-called *discriminant function* δ_k given as

$$\delta_k(\mathbf{x}) = \mathbf{x}^{\intercal} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2} \boldsymbol{\mu}_k^{\intercal} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \ln\, \pi_k, \tag{3.10}$$

which preserves order of classes, so we can classify the observation $\mathbf{x}$ according to the highest value of $\delta_k(\mathbf{x})$. Note that δ_k is linear in its argument, thus explaining why this method is called the linear discriminant analysis.

The discriminant function gives us also a tool to find the decision boundary between 2 classes k_1 and k_2. For an observation to be on the boundary we must have

$$\delta_{k_1}(\mathbf{x}) = \delta_{k_2}(\mathbf{x})$$

which reduces to the following equation

$$\mathbf{x}^\intercal \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2}) + \frac{1}{2}(\boldsymbol{\mu}_{k_2}^\intercal \mathbf{\Sigma}^{-1}\boldsymbol{\mu}_{k_2} - \boldsymbol{\mu}_{k_1}^\intercal \mathbf{\Sigma}^{-1}\boldsymbol{\mu}_{k_1}) + \ln\frac{\pi_{k_1}}{\pi_{k_2}} = 0,$$

which is a linear equation in the variable $\mathbf{x}$. Therefore, the decision boundary for a 2-dimensional problem would be a line, for a multi-dimensional problem a hyperplane.

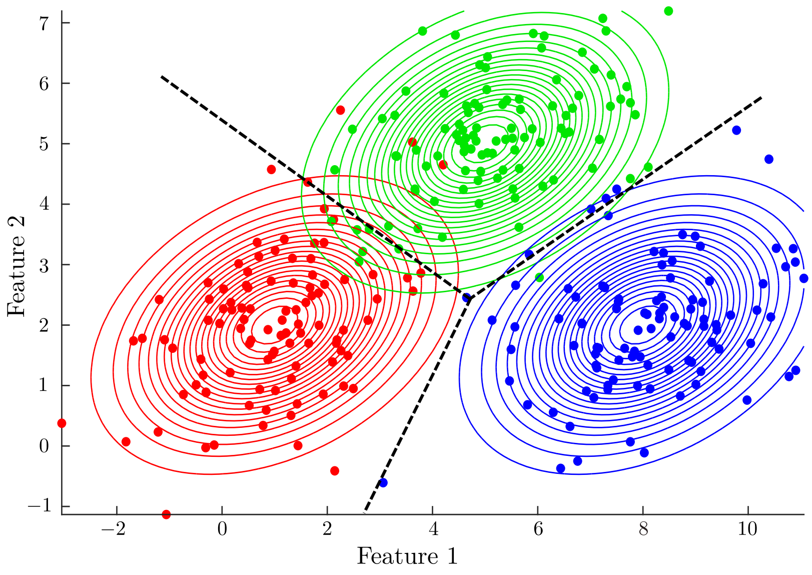


Figure 3.3: Synthetic data from 3 classes plotted with their mutual decision boundaries.

3.3.1 Canonical Discriminant Analysis

Canonical Discriminant Analysis (CDA), as a special case of LDA, is a technique that can be used both for dimensionality reduction and classification. Similarly to PCA, from an exploratory point of view we look for the *canonical axis* defined by some vector $\mathbf{u}$ such that the linear combination $\mathbf{z} = \mathbf{X}\mathbf{u}$ adequately separates classes [22]. Ideally, we want to maximize the between-class dispersion while minimizing the within-class dispersion. It means having the class centroids well separated with observations concentrated closely around them. Unfortunately, it is not possible to achieve both at the same time, so we need to find a compromise.

As mentioned in [1, 22], the overall variance $\boldsymbol{V}$ can be decomposed as $\boldsymbol{V} = \boldsymbol{W} + \boldsymbol{B}$, meaning into the *within-class* variance matrix

$$\boldsymbol{W} = \frac{1}{n-1}\sum_{k=1}^{K} \mathbf{X}_k^\intercal \mathbf{X}_k,$$

where $\mathbf{X}_k$ is the k-th class centered data matrix, and the *between-class* variance matrix

$$\boldsymbol{B} = \frac{1}{n-1}\sum_{k=1}^{K} n_k\,(\boldsymbol{\mu}_k - \boldsymbol{\mu})\,(\boldsymbol{\mu}_k - \boldsymbol{\mu})^\intercal\,,$$

where $\boldsymbol{\mu}_k$ and $\boldsymbol{\mu}$ denote the k-th class and global centroids, respectively. Alternatively, this decomposition can be also expressed as a quadratic form

$$\mathbf{u}^\intercal \boldsymbol{V}\mathbf{u} = \underbrace{\mathbf{u}^\intercal \boldsymbol{W}\mathbf{u}}_{\text{minimize}} + \underbrace{\mathbf{u}^\intercal \boldsymbol{B}\mathbf{u}}_{\text{maximize}}\,.$$

A reasonable compromise is to use the so-called F-ratio criterion [22, 1] which is given as

$$\max\left\{\frac{\mathbf{u}^\intercal \boldsymbol{B}\mathbf{u}}{\mathbf{u}^\intercal \boldsymbol{W}\mathbf{u}}\right\},$$

and can be reformulated using a normalization restriction $\mathbf{u}^\intercal \boldsymbol{W}\mathbf{u} = 1$ to give the following optimization problem

$$\max\ \mathbf{u}^\intercal \boldsymbol{B}\mathbf{u} \quad \text{s.t.} \quad \mathbf{u}^\intercal \boldsymbol{W}\mathbf{u} = 1.$$

This can be solved analogously to the problem (3.4) from section 3.2. Once again we define the Lagrange function $L(\mathbf{u}, \lambda) = \mathbf{u}^\intercal \boldsymbol{B}\mathbf{u} - \lambda\left(\mathbf{u}^\intercal \boldsymbol{W}\mathbf{u} - 1\right)$, we take a gradient with respect to $\mathbf{u}$ and set it equal to zero

$$\nabla_{\mathbf{u}} L(\mathbf{u}, \lambda) = 2\boldsymbol{B}\mathbf{u} - 2\lambda\boldsymbol{W}\mathbf{u} = 0,$$

from which we obtain

$$\boldsymbol{B}\mathbf{u} = \lambda\boldsymbol{W}\mathbf{u}, \tag{3.11}$$

which is called the generalized eigen-value problem (see [9]). Similarly to (3.5), the canonical axes here will also correspond to the eigen-vectors we obtain from (3.11). When the first canonical axis has been found, we search for the next one until we find the total number of canonical axes determined by $\min\{K-1, p\}$ [1, 22]. Intuitively, we can not have more than the original p axes. Additionally, the number of canonical axes is also limited by the rank of the between-class matrix $\boldsymbol{B}$ which is at most $K-1$ [3].

For classification purposes, CDA uses the so-called *Mahalanobis distance* which is a metric induced by the matrix $\boldsymbol{W}^{-1}$

$$d_M^2(\mathbf{x}, \boldsymbol{\mu}_k) = (\mathbf{x} - \boldsymbol{\mu}_k)^\intercal\, \boldsymbol{W}^{-1}\left(\mathbf{x} - \boldsymbol{\mu}_k\right). \tag{3.12}$$

This metric measures the squared distance between the observation $\mathbf{x}$ and the centroid $\boldsymbol{\mu}_k$ by taking into consideration the correlation structure of the predictors [22]. Naturally, if the observation is closest to the centroid $\boldsymbol{\mu}_k$ in terms of Mahalanobis distance, we classify the observation $\mathbf{x}$ to the class k. Note that CDA is a special case of LDA. LDA reduces to CDA under the assumption that the prior probabilities are equal for all classes

$$\pi_1 = \pi_2 = \cdots = \pi_K = \pi.$$

Let us expand the expression (3.12)

$$d_M^2(\mathbf{x}, \boldsymbol{\mu}_k) = \cancel{\mathbf{x}^\intercal \boldsymbol{W}^{-1}\mathbf{x}} - 2\mathbf{x}^\intercal \boldsymbol{W}^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\intercal \boldsymbol{W}^{-1}\boldsymbol{\mu}_k$$

where we can once again omit the term independent of k (terms constant across all classes do not affect the classification outcome). As we can see, this can be seen as a special case of the discriminant function (3.10) from the previous section 3.3

$$-2\mathbf{x}^\intercal \boldsymbol{W}^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\intercal \boldsymbol{W}^{-1}\boldsymbol{\mu}_k \quad \sim \quad -2\delta_k(\boldsymbol{x}) = -2\mathbf{x}^\intercal \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\intercal \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \cancel{-2\ln\pi}.$$

Chapter 4

Implementation

In this chapter, we describe our workflow implementations in Matlab and C++. Additionally, we also show two UI application we developed for visual analysis of the results.

4.1 Matlab workflow

During the initial stages of this thesis, the computation of the LiDAR metrics was done in Matlab as it provided numerous convenient functionalities. Later we also employed Matlab to create the learning dataset and to perform statistical analysis. For visualization purposes, we created two UI applications using the Matlab App Designer. Individual steps of our Matlab workflow can be described as follows:

(M1) **Specifying the area (or areas) of interest.** We can either choose from the already segmented habitat curves, representative squares or manually create a boundary of some area, for example in Google Earth via the "Add - Path" functionality. In both cases, the curves should be stored in the KML format.

(M2) **Import and processing of KML curves.** For each one of the specified curves we first read and save the GPS coordinates of their nodal points. As explained in section 2.2.1, we must transform the nodal points of each curve from WGS84 geodetic coordinates to the S-JTSK(JTSK03) - Krovak East North projected coordinates, see Figure 2.5.

(M3) **Domain specification.** We choose the desired pixel size h, for example $h = 10\,\mathrm{m}$, and the optional padding parameter n_p to define the domain Ω as was described in section 2.3. For further processing we save it as the `polyshape` object.

(M4) **LAZ files search.** As stated in section 2.4.1, we first search for the intersection of Ω with the lots' footprints. Subsequently, we seek which of the relevant lots' LAZ footprints overlap with Ω and save the names of the corresponding LAZ files.

(M5) **Normalization of PC** to get the points' heights relative to the ground. To do so, we iterate through all the pixels of the grid Ω_n. From each point within a pixel we subtract either the value of the DTM or minimum of z coordinates. The comparison of PC data before and after the normalization can be seen in Figure 2.8.

(M6) **Computation of LiDAR metrics.** For each pixel of the grid Ω_h we compute the LiDAR metrics described in Tables 2.1, 2.2 and 2.3.

(M7) **Computation of representative metrics.** This step assumes that the representative squares were chosen as the areas of interest. For each LiDAR metric we compute mean, standard deviation, minimum and maximum but only using the values from internal pixels.

(M8) **Processing of the results.** LiDAR metrics rasters are exported as geo-referenced TIFF images. The representative metrics are saved in a CSV file format, easily accessible for further statistical analysis using methods from chapter 3. For visualization purposes, we created two UI applications in Matlab, see Figures 4.1 and 4.2.

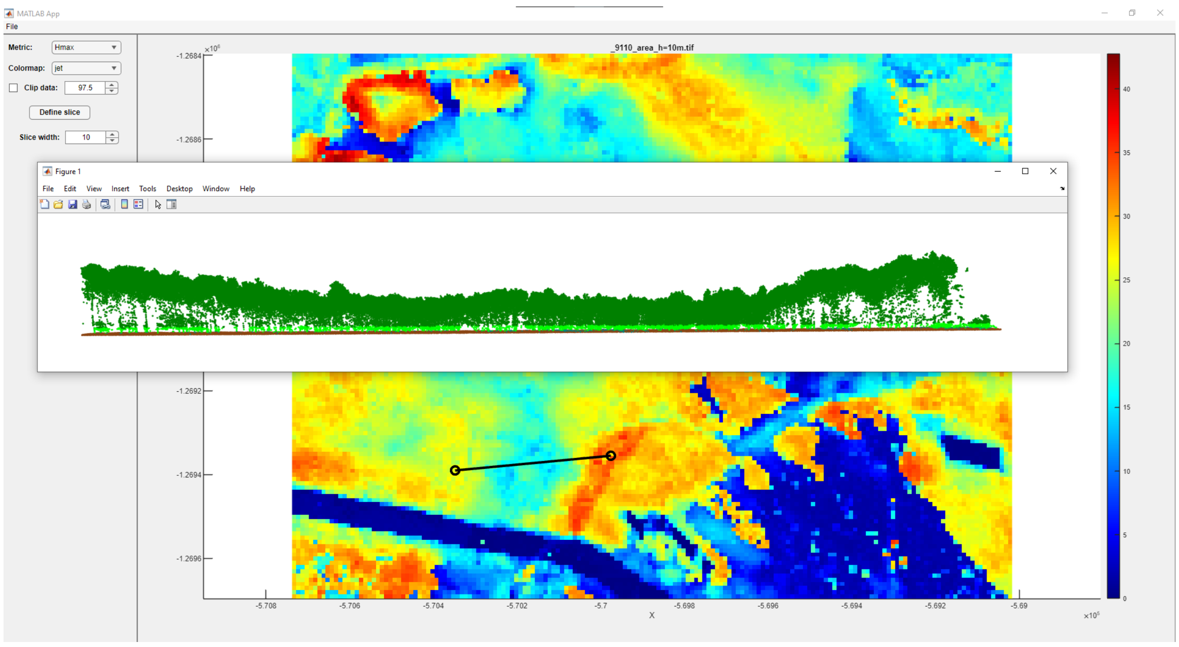


Figure 4.1: UI Matlab application for visualizing LiDAR metrics raster. User can also plot the normalized PC along an arbitrary line segment by selecting two points in the image.

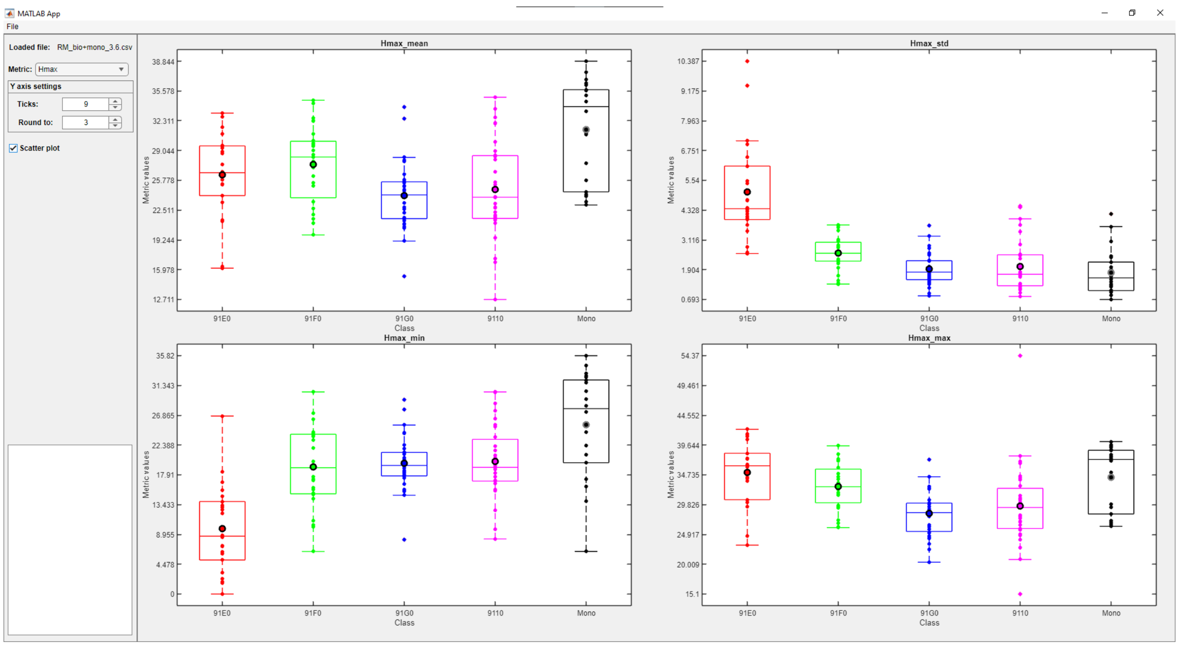


Figure 4.2: UI Matlab application for visual analysis of the representative metrics.

4.2 C++ workflow

For larger grids, the number of points quickly grew to hundreds of millions which resulted in severely prolonged computation time (several hours). Therefore we decided to implement parts of the Matlab workflow using C++. To speed up the computations even further we also parallelized steps (C3), (C4) and (C5) using the *OpenMP* library [2]. We used this workflow also for large-scale computations, further described in section 5.3.

(C1) **Domain specification and LAZ files search** were performed in Matlab, since these steps were sufficiently fast. Therefore, there was no need to implement them in C++. Afterwards, we exported all essential information into a text file, which serves as the input for the C++ program. This text file contains coordinates x_{ul}, y_{ul} of the upper-left corner of the domain Ω, dimensions for the grids Ω_{n} and Ω_h along with paths to all the necessary LAZ files.

(C2) **Load LAZ files** using the *LAStools* library [14] which provides a fast low level access for reading PC data. One of its advantages we utilized was reading the points one by one. This enabled us to store only the relevant points and ignore the rest. We can easily find the indices r and c (row and column) of the pixel in which the given point is located by

$$r = \lfloor \mathrm{abs}(\underbrace{y_P - y_{ul}}_{\delta_y})/h \rfloor \quad \text{and} \quad c = \lfloor \mathrm{abs}(\underbrace{x_P - x_{ul}}_{\delta_x})/h \rfloor, \tag{4.1}$$

where x_P, y_P are x and y coordinates of a point P. At this stage, we use the value $h = 1$ because we distribute points to the grid Ω_{n}. Furthermore, we must check whether the point P is inside Ω_{n} by the following criterion:

$$\text{If } (\delta_x < 0) \text{ or } (\delta_y > 0) \text{ or } (c \geq \mathrm{nx}_{\Omega_{\mathrm{n}}}) \text{ or } (r \geq \mathrm{ny}_{\Omega_{\mathrm{n}}}) \text{ then ignore the point.}$$

We also ignore all points that are not classified as ground or vegetation.

(C3) **Normalization of PC** by subtracting the minimum of z coordinates as described in section 2.4.3.

(C4) **Redistribution of points** to the pixels of the grid Ω_h. This process is analogous to image downsampling by averaging. But instead of averaging we take points from all pixels of Ω_{n} that correspond to a pixel in Ω_h and reassign them to that pixel, see Figure 4.3.

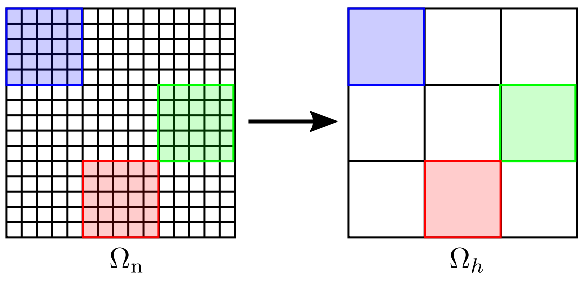


Figure 4.3: Redistribution of points from Ω_{n} to Ω_h with $h = 5$.

(C5) **Computation of LiDAR metrics** similarly as described in step (M6). The only difference is that we had to implement auxiliary functions, for instance for mean value, skewness, etc.

(C6) **Export of the results** using the *GDAL* library [8]. One of the features it provides is a C++ API for seamless data export to a multi-band geo-referenced TIFF image, where each band represents one LiDAR metric.

Chapter 5

Results

5.1 LiDAR metrics with point cloud view

We can review the LiDAR rasters via QGIS, an open source geographic information system [20]. It also provides great features for making images from geo-referenced data, e.g. Figures 5.1a, 5.1b or 5.7. To better understand the vegetation based on individual LiDAR metrics, we also wanted to plot the normalized PC along arbitrary line segments.

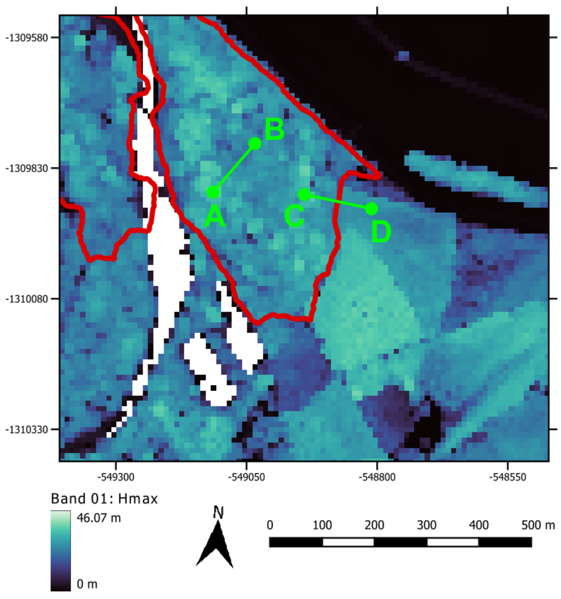
For this purpose we created a simple UI application using the Matlab App Designer, see Figure 4.1. The user can upload a GeoTIFF image containing LiDAR metrics and its corresponding normalized PC data. Once uploaded, the user can specify 2 points on the image. The program will then plot all of the points in the area along the line segment connecting the two specified points. Additionally, the user can also specify the width of the area.

Below, we provide 3 examples of how we can visually analyze the vegetation structure using a combination of LiDAR rasters and the normalized PC. We chose LiDAR metrics Hmax, Hskew and BR_above_20, meaning maximum height, skewness and proportion of vegetation points above 20 m, respectively. For reference, we also added orthomosaic images for the selected areas.

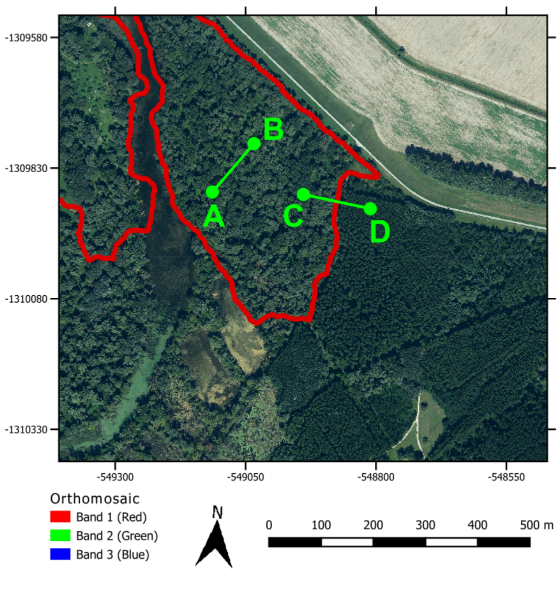
Example 1. In Figure 5.1a, differences between the natural 91E0 habitats and artificially planted monoculture forests are clearly visible. Monocultures are easily distinguishable by areas with almost constant height values, whereas the natural habitats have varying ranges of maximal heights. PC data in Figure 5.1c shows various tree species as expected. In Figure 5.1d we provide an example of PC from natural habitat (left part) and from the monoculture (right part).

Example 2. In this example, we demonstrate the expected vegetation structure based on varying values of skewness. From Figure 5.2c it is evident that areas of negative skewness have the majority of vegetation points concentrated in high denser canopies and barely any vegetation points close to the ground. Values around zero indicate a more uniform distribution of vegetation points along the z axis. In regions with positive skewness, we can expect more vegetation points near the ground.

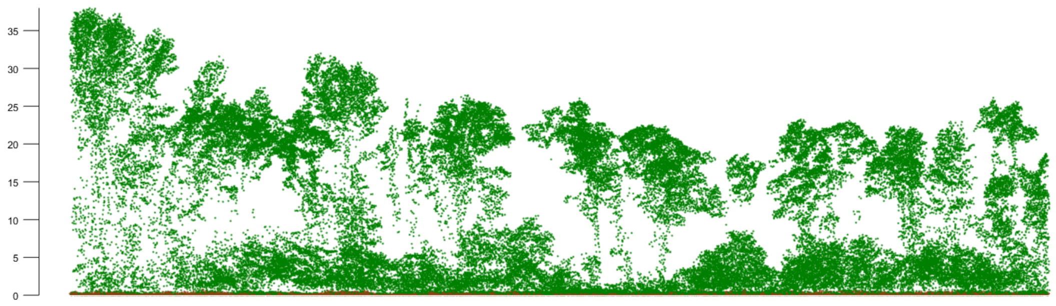
Example 3. LiDAR metrics numbered 9 to 17 are easily interpretable, as they provide information about the proportion of vegetation points either within, below, or above some specific heights. By looking around the point A in Figure 5.3a, we see zero values, which implies the absence of vegetation points above the 20 m mark, as is indicated in Figure 5.3c. On the other hand, in areas with higher values we expect a greater number of vegetation points above 20 m.



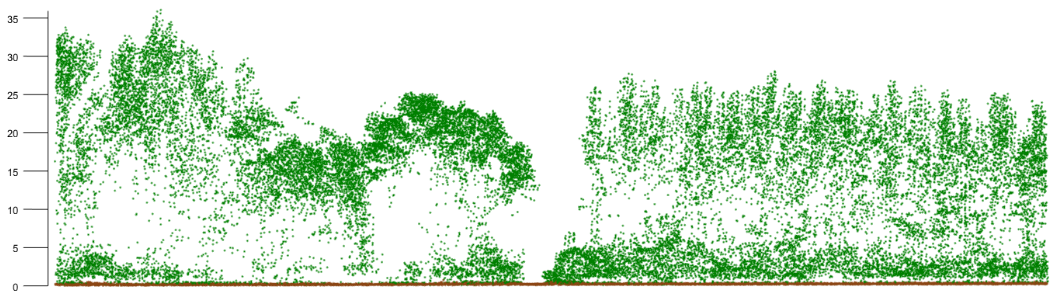
(a) Grid with pixel size $h = 10\,\mathrm{m}$.



(b) Orthomosaic for the selected area.

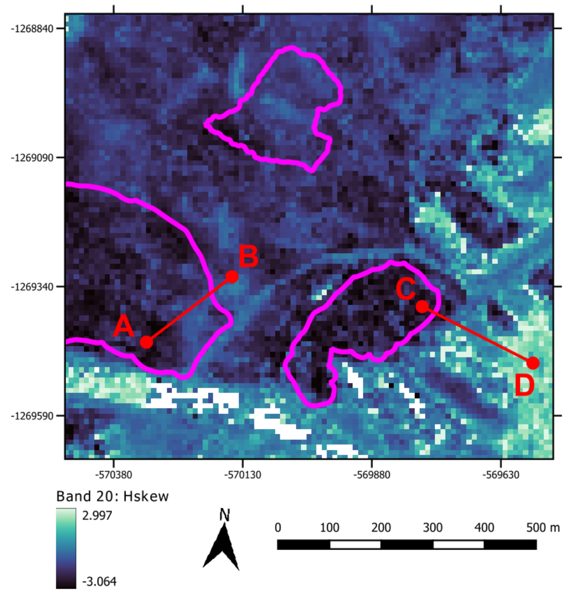


(c) PC along the line segment A-B.

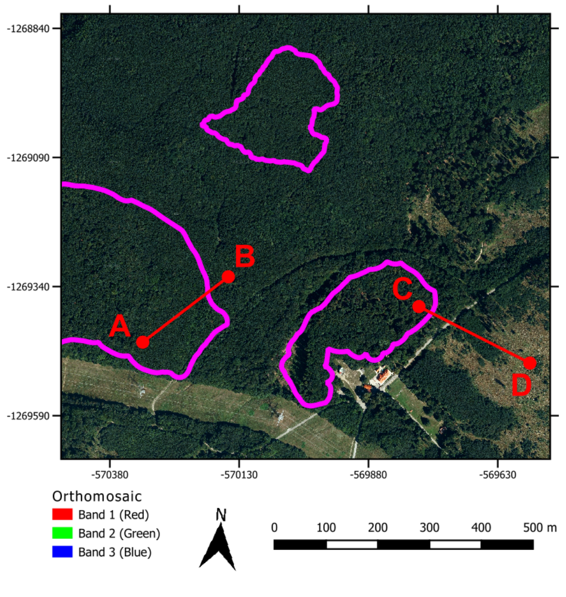


(d) PC along the line segment C-D.

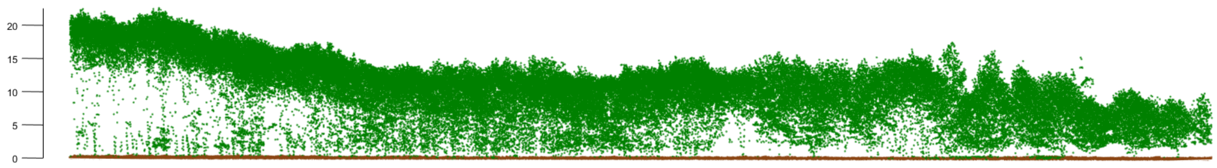
Figure 5.1: Area south-east from village Bodíky depicting natural forest of type 91E0 (red curves) and artificially planted monocultures. LiDAR metric in Figure 5.1a: maximum height of vegetation points in pixel.



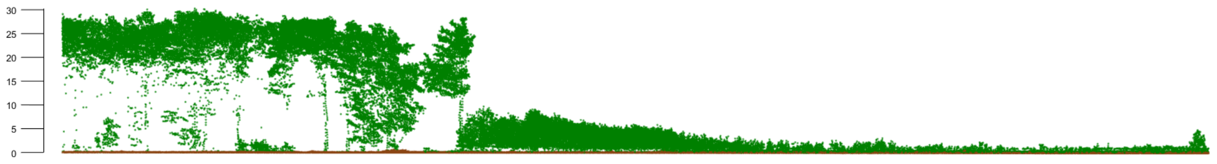
(a) Grid with pixel size $h = 10\,\text{m}$.



(b) Orthomosaic for the selected area.

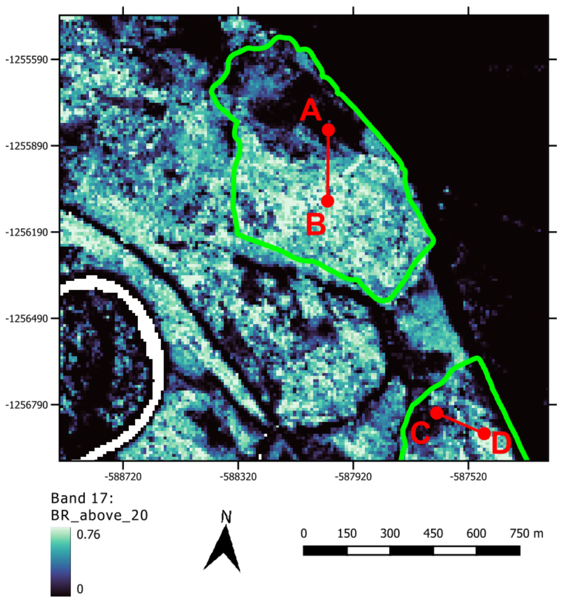


(c) PC along the line segment A-B.

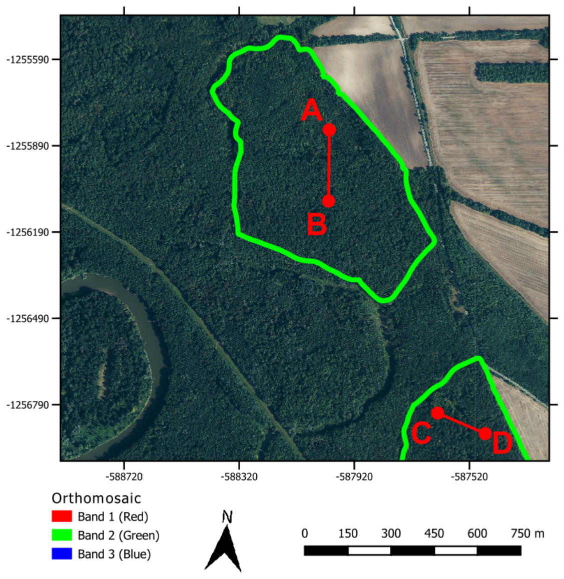


(d) PC along the line segment C-D.

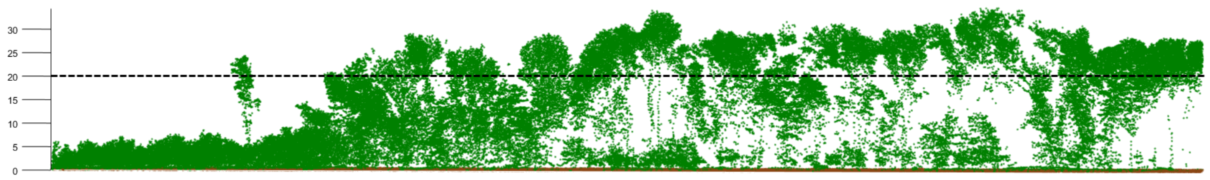
Figure 5.2: Area around Biely Kríž cottage from Figure 2.4 with several of 9110 habitats (magenta curves). LiDAR metric in Figure 5.2a: skewness of height of vegetation points in pixel.



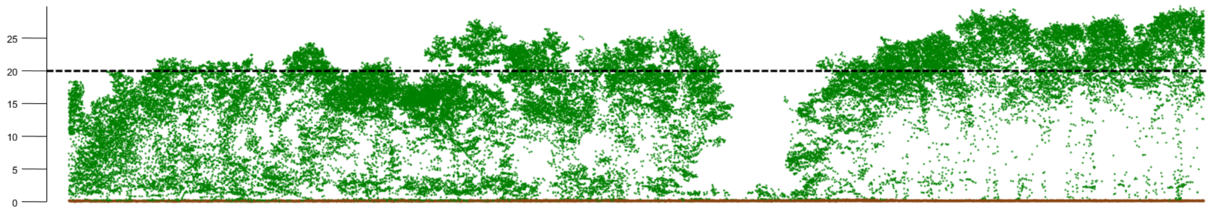
(a) Grid with pixel size $h = 10\,\text{m}$.



(b) Orthomosaic for the selected area.



(c) PC along the line segment A-B.



(d) PC along the line segment C-D.

Figure 5.3: Area around Morava river from Figure 2.4 with some of 91F0 habitats (green curves). LiDAR metric in Figure 5.3a: proportion of vegetation points above 20 m.

5.2 Higher-resolution computations

In this section we present results obtained by using the pixel size $h = 1\,\text{m}$. As the first example we have chosen the maximum height of vegetation points, see Figure 5.4. Thanks to the higher resolution, we can even distinguish shapes of individual trees, which could be potentially useful for tree segmentation.

Interestingly, we may have also identified several trees in this area with height exceeding 40 m, see the zoomed area A in Figure 5.4. In the future, these trees will be measured by the botanists to validate our results.

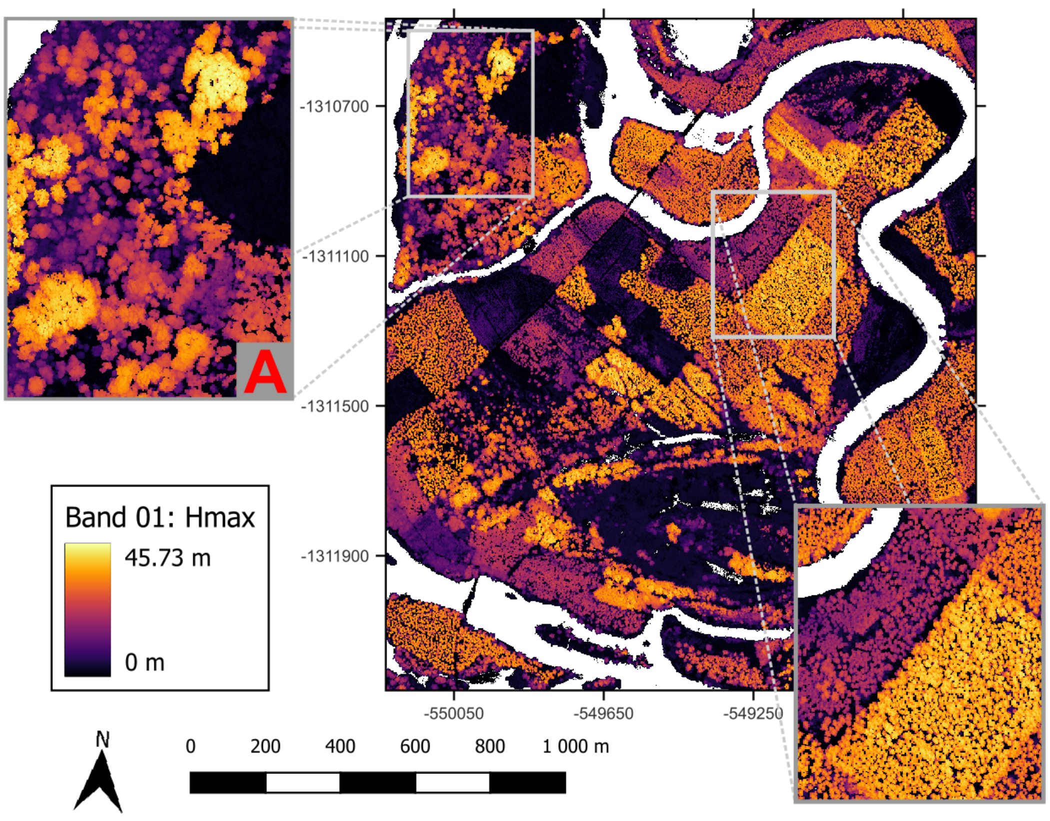
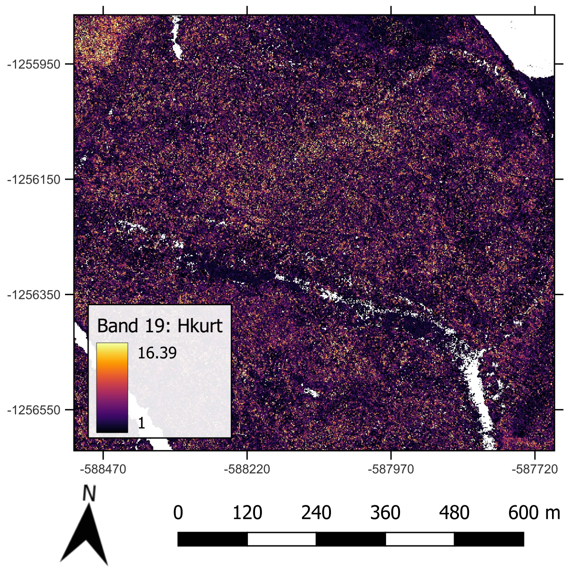
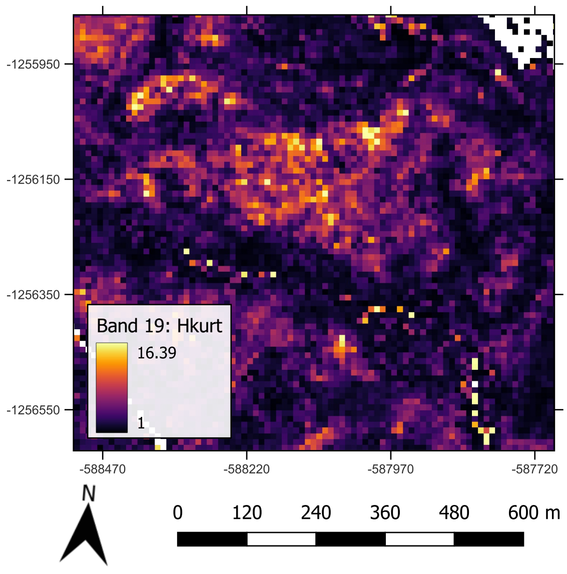


Figure 5.4: Area southeast from Bodíky village. LiDAR metric: maximum height.

It is also worth noting that a finer grid is not always the best choice. For instance, in Figure 5.5a, the image appears to have a lot of noise. On the other hand, with a coarser grid used in Figure 5.5b, we can see some patterns that can be further investigated by plotting the PC, as shown in section 5.1. It should again help us better understand what vegetation structure to expect based on varying values of the kurtosis.



(a) Grid with pixel size $h = 1\,\mathrm{m}$.



(b) Grid with pixel size $h = 10\,\mathrm{m}$.

Figure 5.5: Comparison of the results for kurtosis of vegetation points height with different pixel sizes. In both images we see the area near Vysoká pri Morave.

5.3 Large-scale computations

Because we had access to all PC data for lots 02, 03, 04, 05 and 06 (see Figure 2.2), we decided to compute LiDAR metrics also for a larger area. To be specific, we focused on lots 02 a 03 as they contained most of the segmented curves for Natura 2000 habitats that we had access to.

To do this, we used our grid algorithm from section 2.3 to create a grid of 2×2 km square areas. Subsequently, we selected only those areas that intersected with footprints of lots 02 and 03. However, since we did not have PC data for lot 01 (located above lot 02, see Figure 2.2), we had to exclude some areas that contained PC data from lot 01. For each of the remaining 564 squares (marked by red in Figure 5.6) we created a text file containing all necessary information, as was described in step (C1).

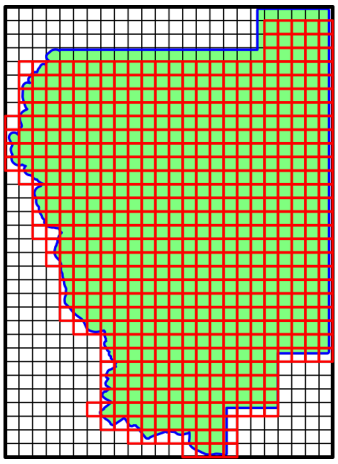


Figure 5.6: Division of lots 02 and 03 into 2×2 km areas, by red color we highlighted squares for which we computed the LiDAR metrics.

Although our implementation in C++ was significantly faster than in Matlab, the computation still took almost 3 days to complete. This was mainly because we did not implement the re-tilling step from [15] which resulted in multiple reads of the LAZ files. Nevertheless, as we can see from Table 5.1, the computation time took only several seconds even with around 200 million points on average per area.

Table 5.1: Statistics from the large-scale computation for the first 41 areas highlighted in red.

# LAZ files	# points	LAZ reading	Normalization	Redistribution	LiDAR metrics
22	203,427,769	6.37 min	0.38 sec	3.96 sec	8.51 sec

After the computation was done, we imported all 564 GeoTIFF images (7 MB per file) into QGIS software [20] and used its "Raster - Miscellaneous - Merge" feature to merge all of them into one large GeoTIFF image (5.5 GB), see Figures 5.7, 5.8 and 5.9. All the 564 individual GeoTIFF images as well as the merged version are available to download at `https://bit.ly/tiff_lots_2_3`.

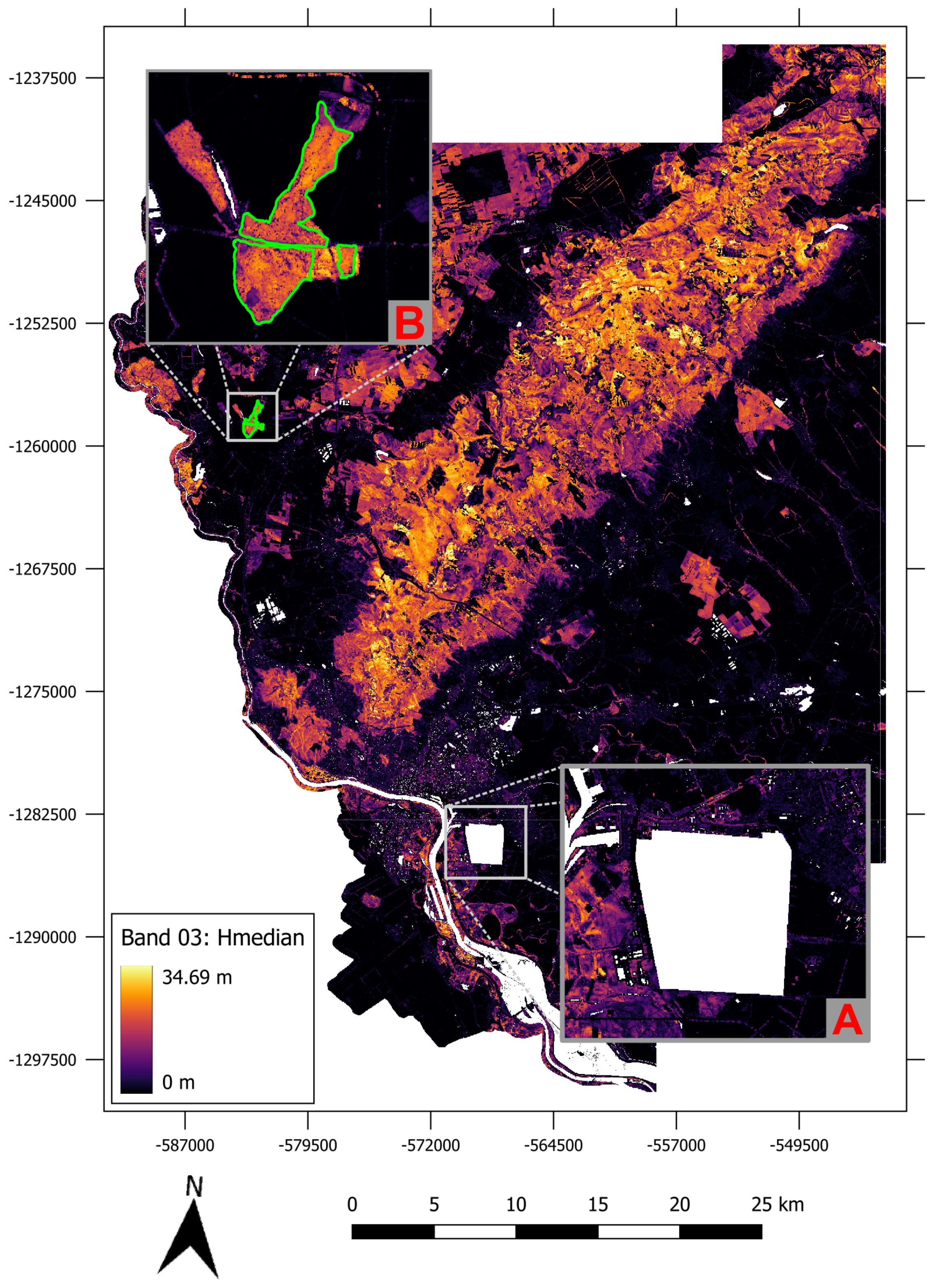


Figure 5.7: Median height of vegetation points for lots 02 and 03. As could be expected, it is mostly the Small Carpathians mountain range that have higher values of the median vegetation height. Additionally, there also regions with NaN values that mostly correspond to rivers or lakes, such as the Danube river. An interesting fact is that several areas were excluded from the areal laser scanning, potentially due to security reasons. One such place is the Slovnaft oil refinery in Bratislava, see the zoomed region A. Other similar areas include, for instance, the nuclear power plants near Mochovce or Jaslovské Bohunice. In the zoomed region B we highlighted one of the 91F0 habitats.

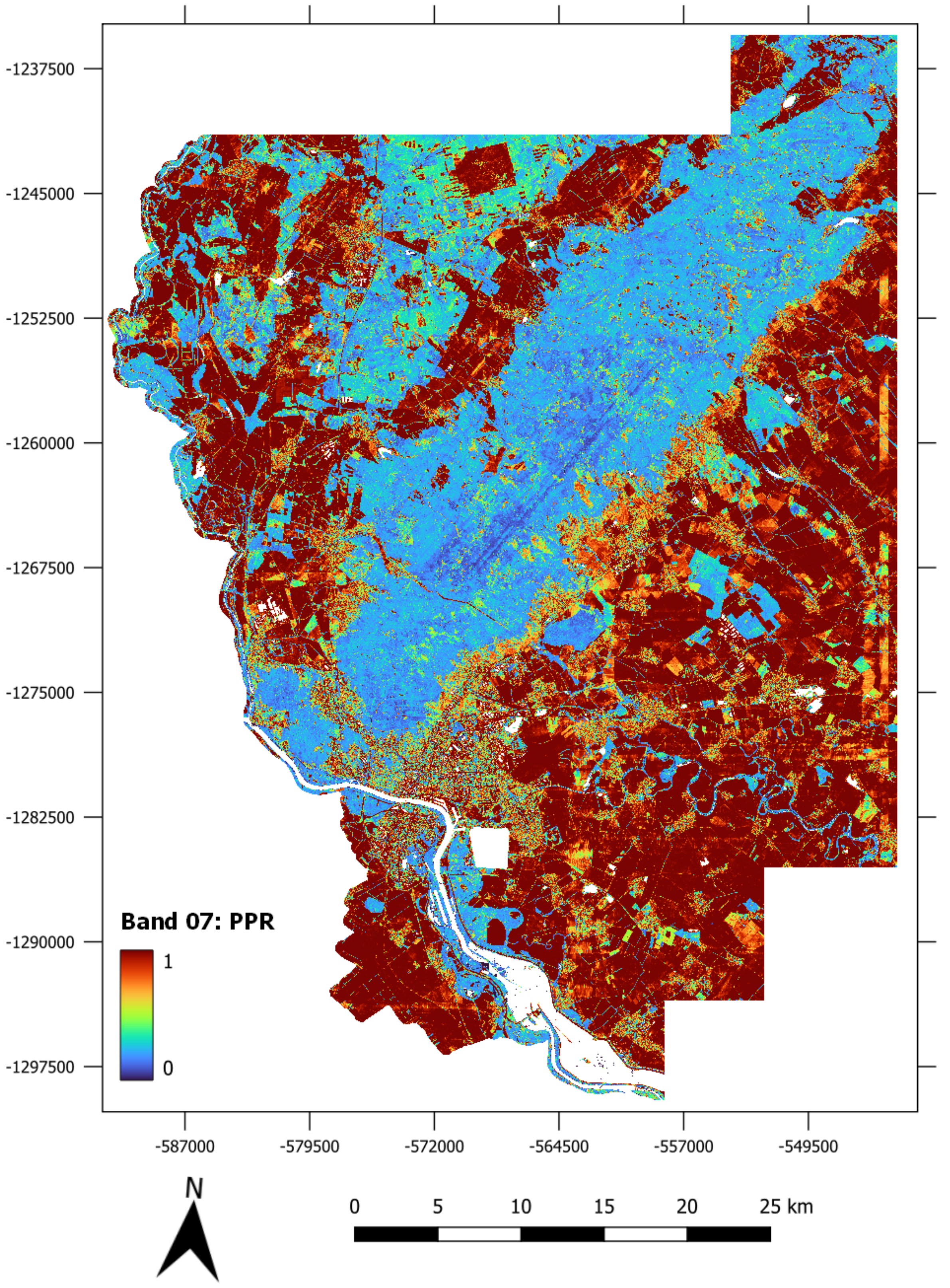


Figure 5.8: Pulse penetration ratio for lots 02 a 03. In contrast to Figure 5.7, places with vegetation in this case correspond to regions with lower values. On the other hand, areas with lower or no vegetation, such as fields or cities, are depicted with values closer to 1.

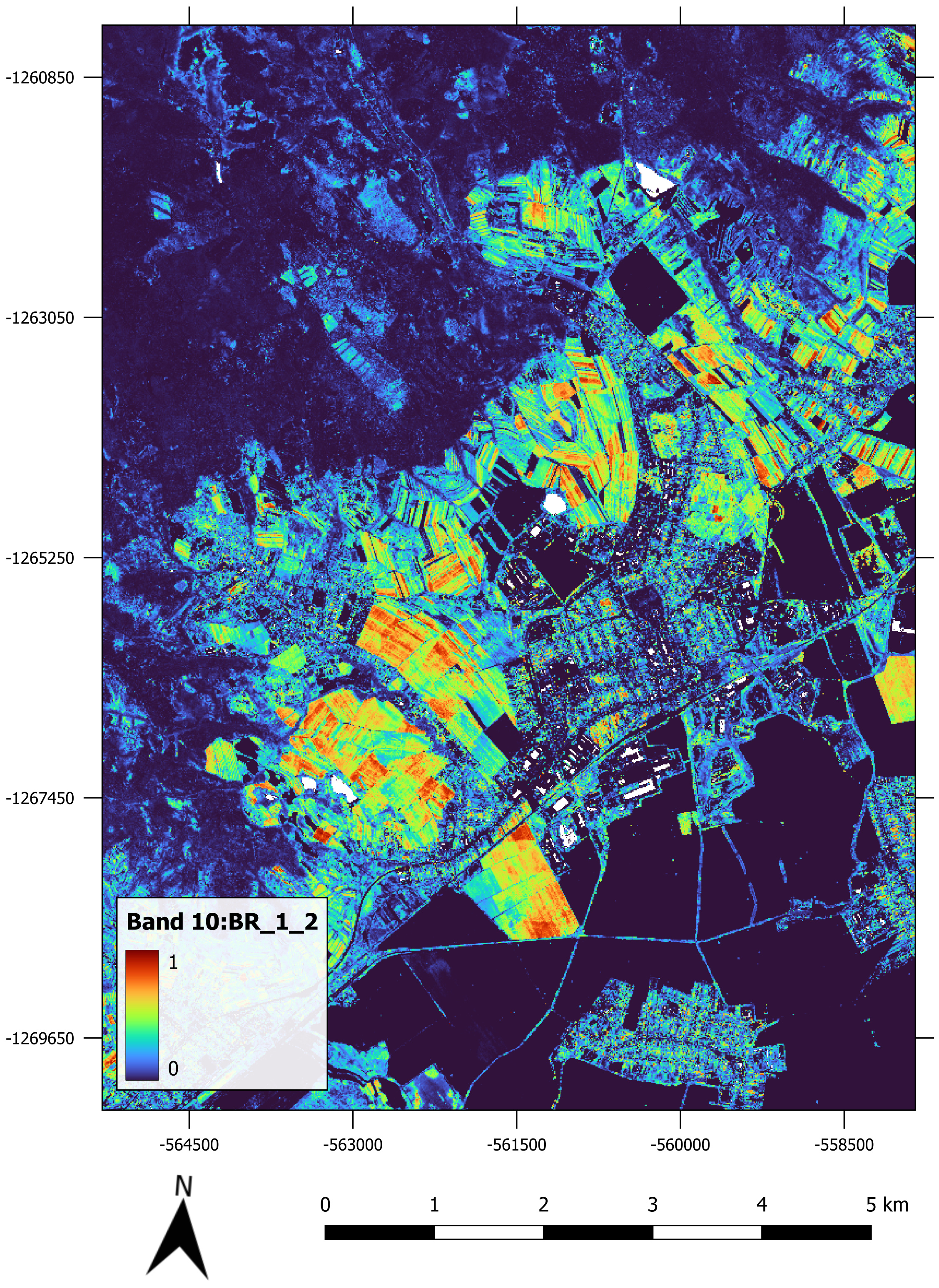


Figure 5.9: Vineyards around Pezinok showing higher proportions of vegetation points between 1 and 2 m above ground.

5.4 Statistical analysis

In this section, we present the results of statistical analysis we conducted in Matlab on a dataset comprising of 133 observations and 92 predictors. Furthermore, we divided the dataset into 3 subsets: habitats only, habitats with monocultures, and only 91E0 habitats with monocultures. Before using CDA as well as the LDA model described in chapter 3, we reduced the number of predictors using PCA. To determine the number of principal components to use, we set a threshold of 99.9% explained variance.

Additionally, since the LDA model assumes a normal distribution of predictors, we checked for normality using the Kolmogorov-Smirnov normality test [16, 23]. This helped us identify any predictors that were unlikely to be normally distributed. Thus we removed these predictors before applying PCA. The number of remaining predictors is shown in Table 5.2.

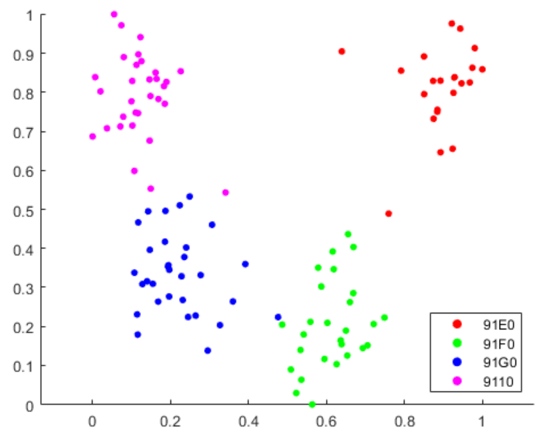
Table 5.2: Number of the remaining predictors.

	All predictors	Only PCA	K-S test + PCA
Only habitats	92	55	32
Habitats + mono	92	58	32
91E0 + mono	92	37	35

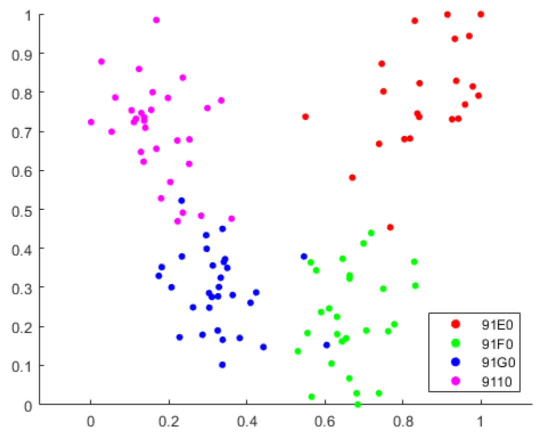
5.4.1 Dimensionality reduction by CDA

As explained in section 3.3.1, Canonical Discriminant Analysis (CDA) is used to project data into a lower dimensional space where classes should be adequately separated. Ideally, we should observe distinct clusters of points for each class. This is evident in Figures 5.10a, and 5.12 when using only the first 2 canonical axes. However, from Figure 5.11 can see that the clusters were better separated when using 3 canonical axes. Interestingly, as we might have expected, the monoculture points in Figures 5.11a and 5.12 are very well separated from the natural habitats.

We may also observe that in Figure 5.12, we added the 91F0 habitat type even though it was not included in the third dataset from Table 5.2. As we explained in section 3.3.1, the number of canonical axes is determined by $\min\{K-1, p\}$. So in order to visualize the clusters using 2 canonical axes, we needed to add at least one more class to the third dataset. We chose the 91F0 habitat type because it is similar to the 91E0 type, as both experience periodical flooding. For classification however, we used only the 91E0 habitat type and the monocultures.

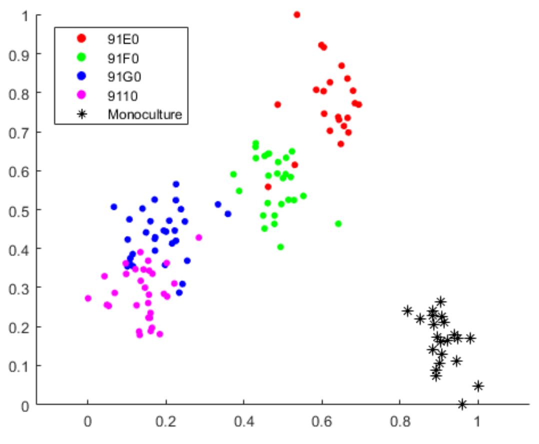


(a) Only PCA applied.

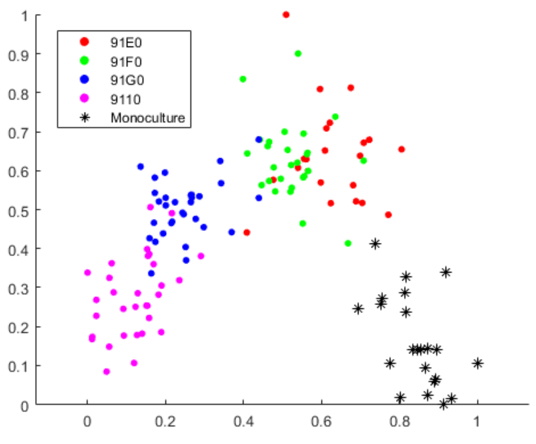


(b) Both K-S test and PCA applied.

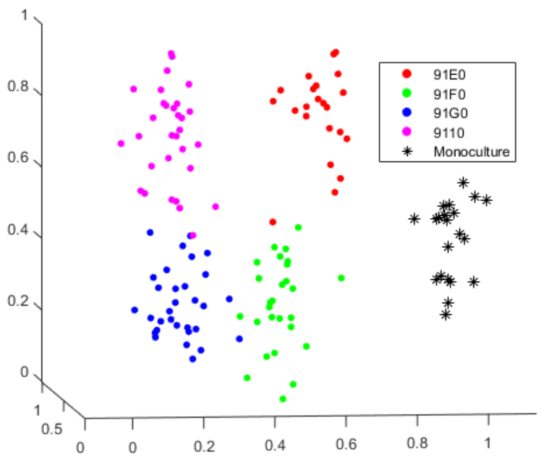
Figure 5.10: Dimensionality reduction via CDA for only the habitats observations.



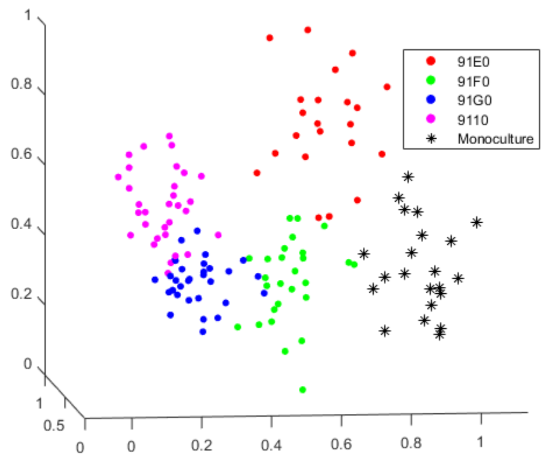
(a) Only PCA applied.



(b) Both K-S test and PCA applied.

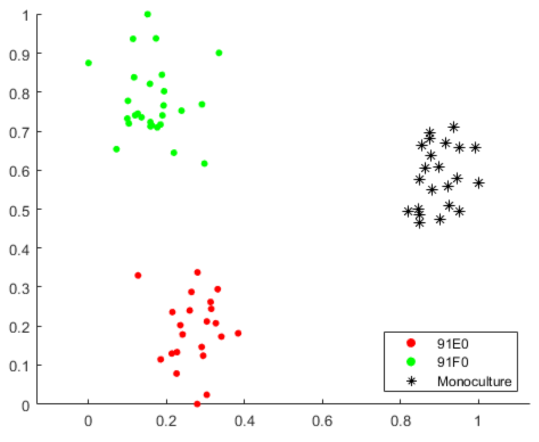


(c) 3D view, only PCA applied.

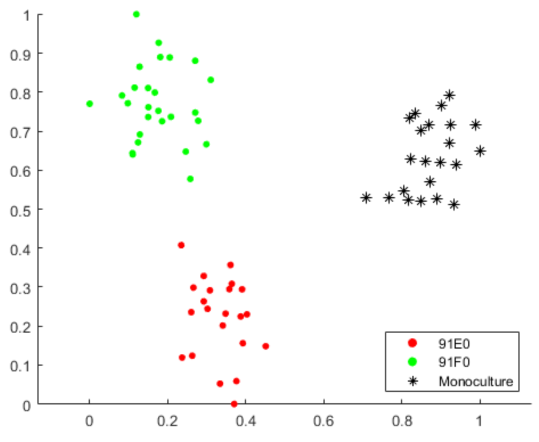


(d) 3D view, both K-S test and PCA applied.

Figure 5.11: Dimensionality reduction via CDA for the habitats and the monoculture observations.



(a) Only PCA applied.



(b) Both K-S test and PCA applied.

Figure 5.12: Dimensionality reduction via CDA for the 91E0 habitats, the 91F0 habitats, and the monoculture observations.

5.4.2 Classification results

In this section, we provide the results of classification from Natural Numerical Networks (NatNets) and the Linear Discriminant Analysis (LDA). Accuracy results were obtained by

Cross Validation (CV) which is a statistical method used to estimate the performance of a classification model. There are various types of CV, but we decided to use k-Fold CV and Leave-One-Out CV.

The k-Fold CV starts with uniformly assigning observations at random into k groups. One group is then selected as a validation sample, while the rest as a training sample to fit the model. We do this for all groups and measure the accuracy of the model each time. Then we calculate an average of these accuracy values to better estimate the model performance. Furthermore, we can repeat this whole process multiple times to get even better accuracy estimate.

Leave-One-Out CV (L-O-O CV) is a specific type of k-Fold CV if the number of groups is equal to the number of observations. L-O-O CV is preferred when the dataset contains a smaller number of observations. The reason being that each time only one observation is selected as a validation sample, thus leaving enough data to fit the model.

Accuracy values for NatNets represent the highest achieved accuracy during its learning phase (estimation of the optimal model parameters using only the first 2 principal components) using only the L-O-O CV.

Below we present the CV results we obtained for the 3 subsets of our dataset. Results from L-O-O CV in Figure 5.13 indicate that the LDA model performed better than the NatNets model in the first 2 cases. However, this could be because the NatNets model was trained using only the first two principal components. On the other hand, it did achieve the highest accuracy of 93.18% in the third case.

The results presented in Figure 5.14 were obtained through k-Fold CV for the LDA model. It indicates that on average the achieved accuracy ranged from 70% to 80%. Figure 5.15 displays the best-case scenarios where the LDA model achieved accuracy between 80% to 90% for all three subsets. In almost all cases we see that selecting only normally distributed predictors improved the accuracy results.

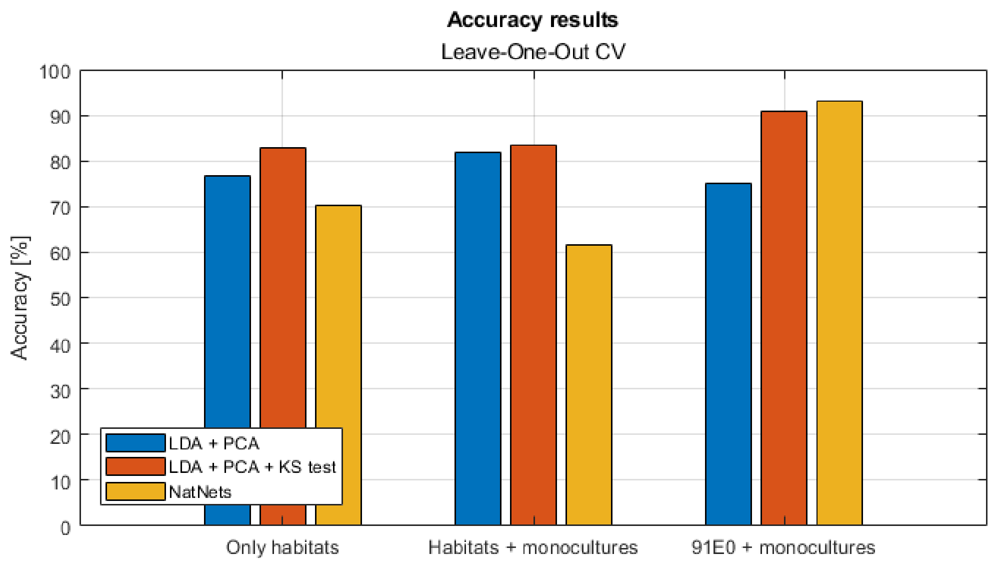


Figure 5.13: Achieved accuracy using the L-O-O CV.

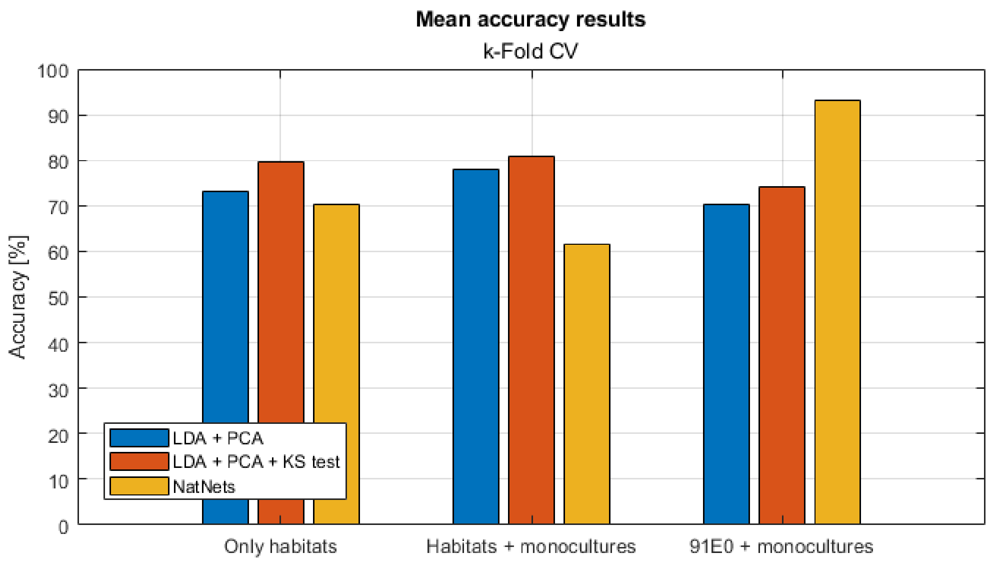


Figure 5.14: Mean achieved accuracy of the LDA model compared to the NatNets model. For the LDA model here we used k-Fold CV with $k = 5$ for the first two subsets and $k = 4$ for the third.

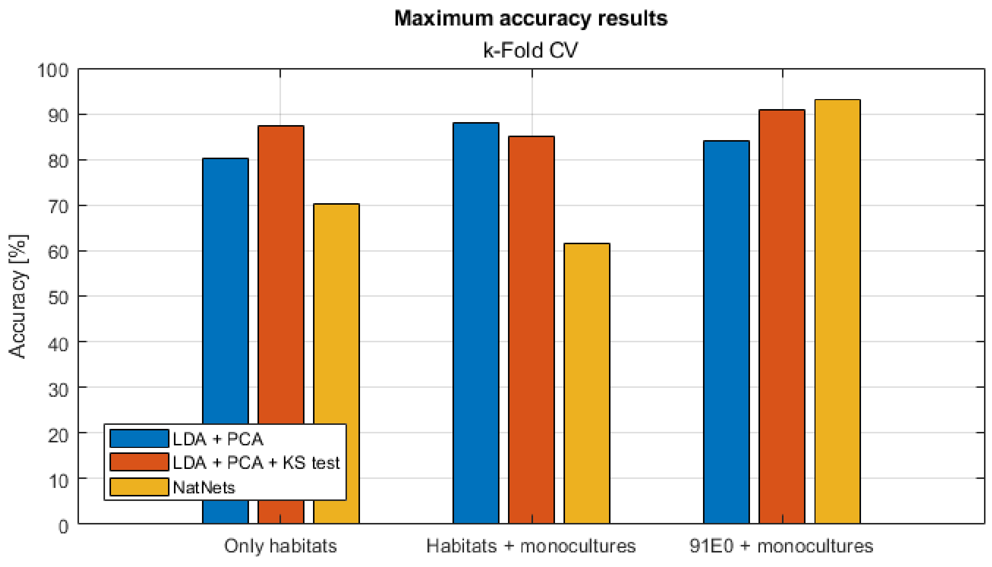


Figure 5.15: Maximum achieved accuracy of LDA model compared to NatNets model. For the LDA model here we used k-Fold CV with $k = 5$ for the first two subsets and $k = 4$ for the third.

Chapter 6

Conclusions

The goal of this thesis was to investigate if classified Point Cloud (PC) data, obtained from aerial laser scanning, can be used to study protected Natura 2000 habitats. The paper [15] served as an inspiration because its authors developed a workflow for extracting useful LiDAR metrics from large-scale PC data into geo-referenced TIFF images. We adopted parts of their workflow with a focus on areas around segmented Natura 2000 habitats.

We have successfully implemented the data import and processing, as well as the algorithm for specification of computational domain and searching for relevant LAZ files using Matlab [13]. We were also able to compute all LiDAR metrics described in Tables 2.1, 2.2, and 2.3, as is visualized in Figure 2.9.

For larger areas, we implemented PC normalization, LiDAR metrics extraction and export in C++ using the LAStools [14], OpenMP [2] and GDAL [8] libraries. This was mainly for computing the LiDAR metrics for the territorial extent of lots 02 and 03, together 564 GeoTIFF images, each covering a 2×2 km region. These were then merged using QGIS software [20] into one GeoTIFF image, see Figures 5.7, 5.8, and 5.9.

Furthermore, we created a UI application using the Matlab App Designer, see Figure 4.1, to visualize the normalized PC along user-specified line segments. This can help us better understand the vegetation structure based on the values of LiDAR metrics. In section 5.1 we provide examples of this functionality in Figures 5.1, 5.2, and 5.3.

Calculations of the representative metrics were successfully performed in Matlab and then exported to a CSV file. To analyze these results visually, we created a second UI application (as shown in Figure 4.2) that can help us identify differences among the 5 assumed classes (4 habitat types and monoculture forests).

The results presented in sections 5.4.1 and 5.4.2 suggest that the classified PC can provide valuable information on the vegetation height structure in order to differentiate habitat types and artificial monocultures.

Regarding our next steps, we plan to implement the re-tilling process from [15]. This will allow us to compute LiDAR metrics for the entire territory of Slovakia, as well as reduce the time required for reading PC data. Additionally, we want to expand our C++ code using the MPI library [17], which will enable us to run the code in parallel on cluster computers.

Resumé

Štúdium biotopov zohráva dôležitú úlohu pri ochrane a zachovaní biodiverzity, pretože rozmanitosť ako aj početnosť rastlín a zvierat úzko súvisí so štruktúrou vegetácie. Za týmto účelom Európska únia zriadila na území Európy sieť chránených oblastí Natura 2000 [4]. Avšak tradičné metódy manuálneho mapovania týchto biotopov sú pre botanikov jednak fyzicky, ale aj časovo veľmi náročné. Našťastie vďaka vývoju moderných technológií diaľkového prieskumu Zeme, máme k dispozícii nástroje, ktoré majú obrovský potenciál zefektívniť celý proces od identifikácie až po monitorovanie chránených biotopov.

Ako príklad môžeme uviesť misiu Sentinel 2 [5], vďaka ktorej máme prístup k multispektrálnym satelitným snímkam s vysokým rozlíšením. Na automatické sledovanie biotopov Natura 2000 boli prirodzene vyvinuté metódy využívajúce tieto snímky, ako je napr. softvér NaturaSat, ktorý ponúka nástroje na segmentáciu aj klasifikáciu [19, 18].

No napriek početným výhodám Sentinel 2 snímok v nich absentuje výšková informácia. Avšak, vďaka celoštátnym leteckým laserovým skenovaniam využívajúcich technológiu LiDAR (Light Detection And Ranging), máme tiež k dispozícii tzv. klasifikované mračno bodov. V jednoduchosti povedané, ide o množinu bodov v $\mathbb{R}^3$, ktoré veľmi detailne popisujú povrch a objekty na Zemi. Každý bod môže byť opísaný okrem jeho x, y, z súradníc aj ďalšou číselnou hodnotou (klasifikácia bodu), ktorá hovorí o tom, čo daný bod predstavuje, napr. zem, vodu, budovy alebo vegetáciu. V tejto práci sme sa zamerali na body klasifikované ako zem alebo vegetácia (triedy 2 až 5). Klasifikované mračno bodov pre celé územie Slovenska [27] poskytuje Úrad Geodézie, Kartografie a Katastra SR.

Na spracovanie klasifikovaného mračna bodov predstavili autori článku [15] pracovný postup s názvom *Laserfarm*. Pozostáva zo 4 krokov, ktoré sa venujú spracovaniu LiDAR dát, ďalej extrakcii rôznych štatistických metrík (LiDAR-ové metriky) a ich následného exportu do formátu GeoTIFF. Týmto postupom teda získame viac-kanálové georeferencované obrázky (pozri obrázok 2.9), pomocou ktorých vieme zistiť napr. maximálnu alebo priemernú výšku vegetácie bez potreby dané miesto navštíviť osobne.

V našej práci sme sa snažili zistiť, či LiDAR-ové metriky, popisujúce vertikálnu štruktúru vegetácie, môžu prispieť novými vedomosťami o chránených biotopoch Natura 2000. Konkrétne sme sa zamerali na biotopy typu 91E0 (mäkké lužné lesy), 91F0 (tvrdé lužné lesy), 91G0 (karpatské a panónske dubodovo-hrabové lesy) a 9110 (kyslomilné bukové lesy), pozri obrázok 2.4. K týmto 4 typom chránených biotopov nám boli poskytnuté ich vysegmentované hraničné krivky a reprezentatívne štvorce (dôležité pre vytvorenie datasetu pre klasifikáciu). Navyše okrem biotopov sme mali k dispozícii aj 22 reprezentatívnych štvorcov pre umelo vysadené monokultúry v blízkosti biotopov 91E0 v povodí Dunaja.

Na spracovanie mračien bodov a extrakciu LiDAR metrík sme využili niektoré postupy z článku [15], ktoré sme bližšie popísali v sekcii 2.4. Na definovanie výpočtovej siete sme vytvorili vlastný algoritmus (pozri sekciu 2.3), pomocou ktorého vieme vytvoriť výpočtovú sieť okolo ľubovoľnej oblasti, ktorú zadá užívateľ. Taktiež pomocou parametra h vieme definovať veľkosť pixela výpočtovej mriežky. Napr. $h = 10$ znamená, že jeden pixel predstavuje v skutočnosti územie o rozmeroch 10×10 metrov. Ďalej sme rozdelili pixely na vnútorné a vonkajšie podľa toho, či sa ich stred nachádzal v krivke γ alebo nie (pozri obrázok 2.7). Toto bolo dôležité pri výpočte reprezentatívnych metrík, keďže tie sa počítali iba z hodnôt vo vnútri repr. štvorcov. Myšlienku o výpočte repr. metrík sme prevzali z článku [18]. V jednoduchosti povedané, pre všetky hodnoty LiDAR-ových metrík z vnútorných pixelov repr. štvorcov vyrátame priemer, štandardnú odchýlku, minimum a maximum. Týmto spôsobom dostaneme pre každý biotop a monokultúru bod v 92-rozmernom priestore črt.

Na klasifikáciu biotopov sme sa rozhodli využiť 2 modely, prirodzené numerické siete (NatNets) a lineárnu diskriminačnú analýzu (LDA). V prvom prípade ide o klasifikačný model, ktorý sa aktuálne využíva v softvéri NaturaSat [18]. Jeho základné princípy vysvetľujeme v sekcii 3.1. V sekcii 3.3 v skratke popisujeme, ako fungujú modely z triedy generatívnych klasifikátorov a aké zjednodušujúce predpoklady uvažujeme v prípade LDA [22, 1]. Kvôli veľkej dimenzii priestoru črt sme využili taktiež analýzu hlavných komponentov (PCA) s cieľom zníženia dimenzie, pozri sekciu 3.2. Navyše sme aplikovali aj kanonickú diskriminačnú analýzu (CDA), sekcia 3.3.1, ktorej úlohou je nájsť taký podpriestor, v ktorom sú zhluky bodov pre jednotlivé triedy od seba dobre oddelené.

Všetky výpočty sme implementovali v kombinácii programu Matlab [13] a jazyk C++. Jednotlivé kroky nášho pracovného postupu sme popísali v kapitole 4. Na prezentáciu výsledkov sme vytvorili pomocou prostredia Matlab App Designer dve aplikácie s užívateľským rozhraním. V prvej aplikácii (pozri obrázok 4.2), vieme vizuálne analyzovať reprezentatívne metriky a takto identifikovať potenciálne zaujímavé rozdiely medzi jednotlivými triedami. Druhá aplikácia (pozri obrázok 4.1), slúži na zobrazenie multi-kanálových GeoTIFF obrázkov, v ktorých sú uložené extrahované LiDAR-ové metriky. Navyše sme v nej implementovali aj funkcionalitu zobrazenia klasifikovaného mračna bodov pozdĺž užívateľom zadanej úsečky. Týmto spôsobom vieme lepšie porozumieť, akej vegetácii zodpovedajú rôzne hodnoty LiDAR-ových metrík.

V sekcii 5.1 ukazujeme na 3 príkladoch využitie funkcionality zobrazenia mračna bodov v rôznych rezoch na analýzu LiDAR-ových metrík. Taktiež v sekcii 5.2 uvádzame výhody a nevýhody voľby jemnejšej výpočtovej siete. Keďže normalizácia a výpočet LiDAR-ových metrík pre väčšie oblasti trval v Matlabe príliš dlho (aj niekoľko hodín), rozhodli sme sa ich implementovať aj v jazyku C++. Načítanie LiDAR dát nám umožnila knižnica LAStools [14] a export metrík do multi-kanalových GeoTIFF obrázkov knižnica GDAL [8]. Navyše sme výpočty v C++ aj paralelizovali pomocou knižnice OpenMP [2]. Týmto sa nám úspešne podarilo zredukovať potrebný čas na úroveň sekúnd, ak nerátame samotné načítanie LiDAR dát. Vďaka tomu sme mali možnosť realizovať výpočet aj pre takmer celé územie lokalít 02 a 03, výsledky môžeme vidieť na obrázkoch 5.7, 5.8 a 5.9. Všetky GeoTIFF obrázky zo sekcie 5.3 sú k dispozícii na stiahnutie cez `https://bit.ly/tiff_lots_2_3`.

Nakoniec v sekcii 5.4.1 prezentujeme dosiahnuté výsledky z redukcie dimenzie pomocou CDA. Ako je vidno z obrázkov 5.10 5.12, jednotlivé klastre boli dostatočne oddelené už pri dvoch dimenziách. Z obrázku 5.11 je zjavné, že pri vyššom počte tried potrebujeme viac kanonických osí na to, aby boli klastre od seba adekvátne oddelené. Na základe dosiahnutých úspešností klasifikácie (pozri obrázky 5.13, 5.14 a 5.15), môžeme usúdiť, že LiDAR dáta nám vedia poskytnúť relevantné informácie o vertikálnej štruktúre vegetácie aj na účely klasifikácie chránených biotopov Natura 2000.

Ďalšie možné kroky výskumu zahŕňajú konzultovanie a validáciu dosiahnutých výsledkov s expertami z oblasti botaniky. Taktiež v budúcnosti plánujeme rozšíriť náš C++ kód pomocou knižnice MPI [17], aby sme vedeli realizovať výpočet LiDAR-ových metrík pre celé územie Slovenska aj s pomocou výpočtových klastrov.

Bibliography

1. BACIGÁL, T. *Pokročilé metódy štatistického modelovania.* 2023. Available also from: https://tomas-bacigal.quarto.pub/pokrocile-metody-statistickeho-modelovania.
2. CHANDRA, R.; DAGUM, L.; KOHR, D.; MENON, R.; MAYDAN, D.; MCDONALD, J. *Parallel programming in OpenMP.* Morgan kaufmann, 2001.
3. CHEN, S.; LIU, C. Clustering-Based Discriminant Analysis for Eye Detection. *Image Processing, IEEE Transactions on.* 2014, vol. 23, pp. 1629–1638. Available from DOI: 10.1109/TIP.2013.2294548.
4. COMMISSION, E.; ENVIRONMENT, D.-G. for; MÉZARD, N; SUNDSETH, K; WEGEFELT, S. *Natura 2000 – Protecting Europe's biodiversity.* Ed. by WEGEFELT, S. European Commission, 2008. Available from DOI: doi/10.2779/45963.
5. DRUSCH, M.; DEL BELLO, U.; CARLIER, S.; COLIN, O.; FERNANDEZ, V.; GASCON, F.; HOERSCH, B.; ISOLA, C.; LABERINTI, P.; MARTIMORT, P.; MEYGRET, A.; SPOTO, F.; SY, O.; MARCHESE, F.; BARGELLINI, P. Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services. *Remote Sensing of Environment.* 2012, vol. 120, pp. 25–36. ISSN 0034-4257. Available from DOI: https://doi.org/10.1016/j.rse.2011.11.026. The Sentinel Missions - New Opportunities for Science.
6. *ESRI Shapefile Technical Description.* Redlands, CA, 1998. Tech. rep. Environmental Systems Research Institute, Inc. Available also from: https://www.esri.com/content/dam/esrisites/sitecore-archive/Files/Pdfs/library/whitepapers/pdfs/shapefile.pdf.
7. FRIEDMAN, J.; TILLICH, J.-P. Wave equations for graphs and the edge-based Laplacian. *PACIFIC JOURNAL OF MATHEMATICS.* 2004, vol. 216. Available from DOI: 10.2140/pjm.2004.216.229.
8. GDAL/OGR CONTRIBUTORS. *GDAL/OGR Geospatial Data Abstraction software Library.* Open Source Geospatial Foundation, 2024. Available from DOI: 10.5281/zenodo.5884351.
9. GHOJOGH, B.; KARRAY, F.; CROWLEY, M. Eigenvalue and Generalized Eigenvalue Problems: Tutorial. 2019. Available also from: https://www.researchgate.net/publication/332032248_Eigenvalue_and_Generalized_Eigenvalue_Problems_Tutorial.
10. GOOGLE FOR DEVELOPERS. *What is KML?* 2023. Available also from: https://developers.google.com/kml.
11. HARIKUMAR, A. *Advanced methods for tree species classification and biophysical parameter estimation using crown geometric information in high density LiDAR data.* 2020. Available from DOI: 10.13140/RG.2.2.10395.69922. PhD thesis.
12. HOMOLA, T. *Correction of watercourses in maps using airborne laser scanning data.* 2022. Available also from: https://opac.crzp.sk/?fn=detailBiblioForm&sid=EC9507141B5FF6F04102CCC822A1. Bachelor's Thesis.

13. INC., T. M. *MATLAB version: 9.13.0 (R2022b)*. Natick, Massachusetts, United States: The MathWorks Inc., 2022. Available also from: `https://www.mathworks.com`.
14. ISENBURG, M. *LAStools - efficient LiDAR processing software*. 2024. Available also from: `http://rapidlasso.com/LAStools`.
15. KISSLING, W. D.; SHI, Y.; KOMA, Z.; MEIJER, C.; KU, O.; NATTINO, F.; SEIJMONSBERGEN, A. C.; GROOTES, M. W. Laserfarm – A high-throughput workflow for generating geospatial data products of ecosystem structure from airborne laser scanning point clouds. *Ecological Informatics*. 2022, vol. 72, p. 101836. ISSN 1574-9541. Available from DOI: `https://doi.org/10.1016/j.ecoinf.2022.101836`.
16. KOLMOGOROV, A. L. Sulla determinazione empirica di una legge di distribuzione. *G. Ist. Ital. Attuari*. 1933, vol. 4, pp. 83–91. Available also from: `https://cir.nii.ac.jp/crid/1571135650766370304`.
17. MESSAGE PASSING INTERFACE FORUM. *MPI: A Message-Passing Interface Standard Version 4.1*. 2023. Available also from: `https://www.mpi-forum.org/docs/mpi-4.1/mpi41-report.pdf`.
18. MIKULA, K.; KOLLÁR, M.; OŽVAT, A. A.; AMBROZ, M.; ČAHOJOVÁ, L.; ŠIBÍK, J.; JAROLÍMEK, I.; ŠIBÍKOVÁ, M. Natural numerical networks for Natura 2000 habitats classification by satellite images. *Applied Mathematical Modelling*. 2023, vol. 116, pp. 209–235. ISSN 0307-904X. Available from DOI: `https://doi.org/10.1016/j.apm.2022.11.021`.
19. MIKULA, K.; URBÁN, J.; KOLLÁR, M.; AMBROZ, M.; JAROLÍMEK, I.; ŠIBÍK, J.; ŠIBÍKOVÁ, M. An automated segmentation of NATURA 2000 habitats from Sentinel-2 optical data. *Discrete & Continuous Dynamical Systems - S*. 2021, vol. 14, no. 3, pp. 1017–1032.
20. QGIS DEVELOPMENT TEAM. *QGIS Geographic Information System*. QGIS Association, 2024. Available also from: `https://www.qgis.org`.
21. RUFFHEAD, A.; WHITING, B. *Introduction to geodetic datum transformations and their reversibility*. School of Architecture, Computing and Engineering, University of East London, 2020. Available also from: `https://www.researchgate.net/publication/339887497`.
22. SANCHEZ, G.; MARZBAN, E. *All Models Are Wrong: Concepts of Statistical Learning*. 2020. Available also from: `https://allmodelsarewrong.github.io`.
23. SMIRNOV, N. V. Table for Estimating the Goodness of Fit of Empirical Distributions. *Annals of Mathematical Statistics*. 1948, vol. 19, pp. 279–281. Available also from: `https://api.semanticscholar.org/CorpusID:120842954`.
24. THE MATHWORKS INC.. *Lidar Toolbox version: 2.2 (R2022b)*. Natick, Massachusetts, United States: The MathWorks Inc., 2022. Available also from: `https://www.mathworks.com`.
25. THE MATHWORKS INC.. *Mapping Toolbox version: 5.4 (R2022b)*. Natick, Massachusetts, United States: The MathWorks Inc., 2022. Available also from: `https://www.mathworks.com`.
26. ÚGKK SR. *1st project cycle (2017 – 2023) and creation of DTM 5.0*. 2024. Available also from: `https://www.geoportal.sk/en/zbgis/als/1st-cycle/`.
27. ÚGKK SR. *Airborne Laser Scanning*. 2024. Available also from: `https://www.geoportal.sk/en/zbgis/als/`.
28. ÚGKK SR. *Provision of ALS products*. 2024. Available also from: `https://www.geoportal.sk/en/zbgis/airborne-laser-scanning/als-products-providing/provision-als-products.html`.