UNIVERSITY
OF OSLO

Master's thesis

Validating snow depth measurements from the ICESat-2 space laser in forested terrain

Simen Dalseid Aune

Geomorphology and Geomatics
60 ECTS study points

Department of Geosciences
Faculty of Mathematics and Natural Sciences

Spring 2024

Todo list

Simen Dalseid Aune

Validating snow depth measurements from the ICESat-2 space laser in forested terrain

Supervisors:
Désirée Treichler
Clare Webster

Abstract

NASA's Ice, Cloud, and land Elevation Satellite 2 (ICESat-2) provides a way to measure heights at a global scale, at a minimum of 90-day intervals, with high resolution. An existing project exists to assess snow depths using these measurements. This thesis builds on the work in that project, and attempts to validate these measurements in areas with vegetation.

Don't touch the abstract until the end

Contents

List of Figures

List of Tables

Preface

Is the preface necessary? (It's present in the UiO template)

Chapter 1

Introduction

1 Background

Finish the background section. Describe what has been done, and why the research question is relevant

Repeated measurements of snow depths – at large scale – can serve a multitude of purposes in Geosciences. They can provide estimates of available hydropower and drinking water resources; As time series spanning several years they can serve as an indicator of climate change; Given high spatio-temporal resolution they can even help predict the risk of avalanches. In turn, decision makers are able to make better decisions in the future. However, gathering the data from the field is resource demanding, and practically only a viable option in wealthy, developed countries. Developing countries with increasing energy demand, sometimes combined with challenges brought on by climate change, generate the need for low-cost methods of supervising the amounts of snow in large catchments.

Satellite-born sensors may provide one solution to this problem. The Ice, Cloud and land Elevation Satellite (ICESat-2) mission carries an instrument called the Advanced Topographic Laser Altimeter System (ATLAS), with a near-global coverage and 91-day repeat period (Thomas A. Neumann et al. 2019). The instrument has six beams of a green laser (532 nm), using the time of flight for each photon to calculate the distance to the surface.

2 Problem Definition

Rewrite the problem statement to better describe what I am doing

Several methods have been developed to measure heights using the ICESat-2 ATLAS instrument in forests, and some studies have looked into its ability to accurately measure snow surfaces. Few — if any — attempts have been made to combine the two, and evaluate the effect vegetation can have on the instruments ability to measure the surface in snow-covered terrain. The ability to accurately estimate snow depths at a large scale can aid decision makers in knowing how much hydropower and drinking water is available, and possibly — over time — provide a new metric to assess the effects of global warming.

This thesis will evaluate whether forests and dense vegetation have a significant affect on the height measurements made by the ATLAS instrument, when the surface is covered in snow. The problem statement for the thesis is thus:

> *The goal of this project is to evaluate the ATLAS instruments ability to measure snow-covered surface heights in forests, by comparing photon heights to drone-borne LiDAR datasets in three sample sites.*

Drone-borne LiDAR datasets have been collected in the chosen study sites, to generate high-resolution terrain models and canopy height models. These models are used to verify the photon heights from the ATLAS instrument, which have been filtered with one of three different methods:

1. Percentile-based threshold distance from a ground-surface DTM
2. A simple Nearest-neighbour averaging algorithm
3. Yet Another Photon Classifier (YAPC) algorithm

The output from the three methods are then compared to each other, to evaluate their efficiency in filtering out noise and stray photons.

Chapter 2

Study Area and Data

3 Study Area

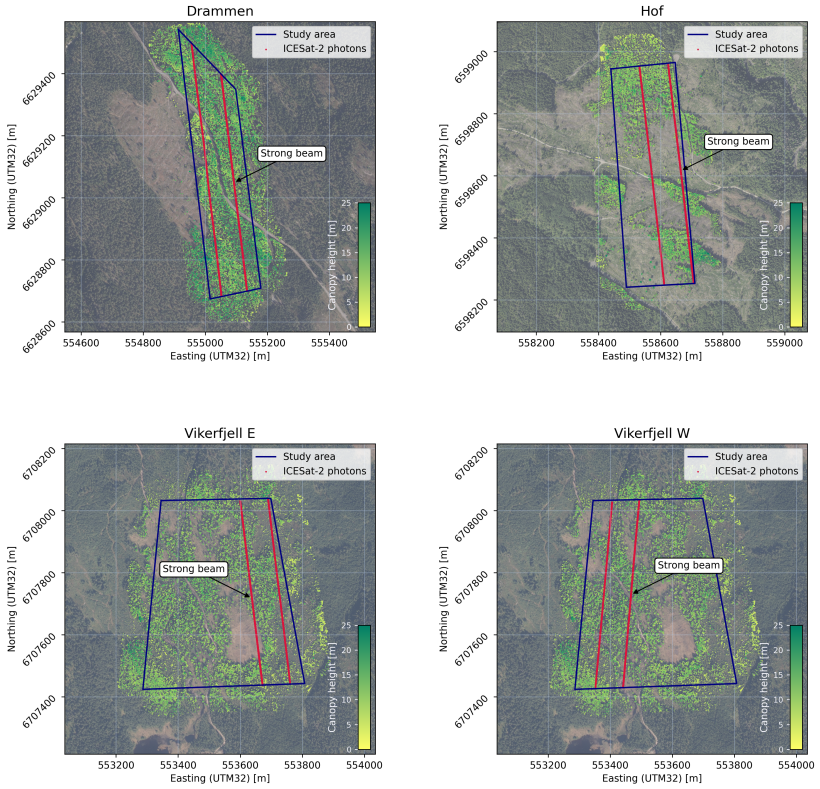

Figure 2.1: All of the sites are shown in this figure. Larger format figures are shown in each subsection.

The sample sites used in this thesis all lie to the west of Oslo (fig. 2.1). A total of four sites were chosen, all containing varying slope, aspect and canopy cover. One of the chosen sites, in Jevnaker, was later excluded from further analysis due to bad overlap between the ICESat-2 and drone data. The canopy structure at all the sites mostly consists large (>10 m) coniferous trees like spruce and pine, with a insignificant presence of deciduous trees.

Move some of this to discussion?

The main challenge in the site selection for this thesis was timing. ICESat-2 has a 91-day exact repeat orbit, which in turn means that each site can essentially only be used once. Snow properties can change significantly during the course of a few hours, so to ensure a similar snow cover in both satellite and drone acquisition, the data collection should have temporal overlap. This means that weather becomes and important factor, as both sensors have their own limitations. The ATLAS instrument can not penetrate clouds , which are prevalent during the winter season, while the drone is unable to operate if the wind speed gets too high. Low temperatures also reduce battery life, and thus flying time. Sufficiently accurate weather predictions are only available a day or to in advance, meaning that we had to maintain a high readiness and flexibility in deciding where to go. For practical and logistical reasons, this also meant that the study area should be close to where the equipment and people were, i.e. within a few hours drive from Oslo.

Cite ATLAS requirements

A significant and measurable snow cover has to be present in the area, in combination

with a tree canopy suitable for answering the thesis' research question. Drone flight in Norway has its own requirements (Luftfartstilsynet 2024). The most important one in this case was to keep a sufficient distance (>150 m) to populated areas, such as houses, and to stay out of no-fly zones. At the same time, having road access in or near the site makes the logistics significantly easier.

Describe snow cover

3.1 Drammen

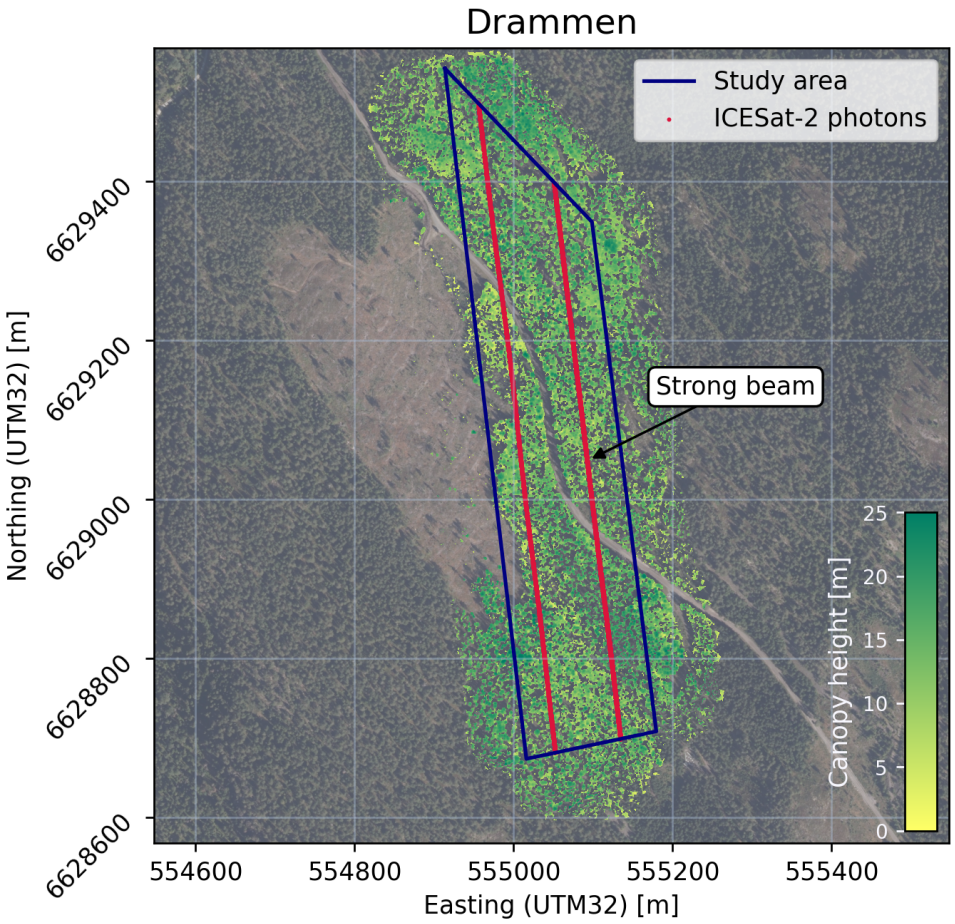

Figure 2.2: The site in Drammen municipality.

The first site lies in the western part of Drammen municipality, just north of Krokstadelva.

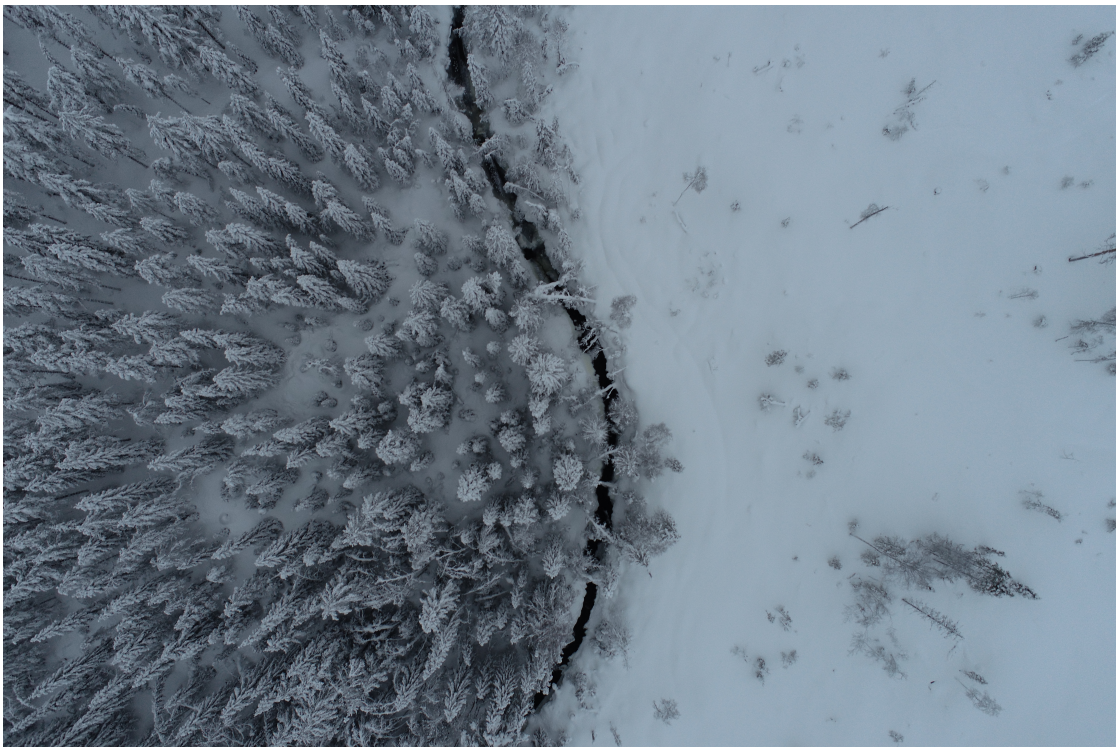

Figure 2.3: Sample image from the UAV flight, showing the canopy structure, covered in snow.

3.2 Hof

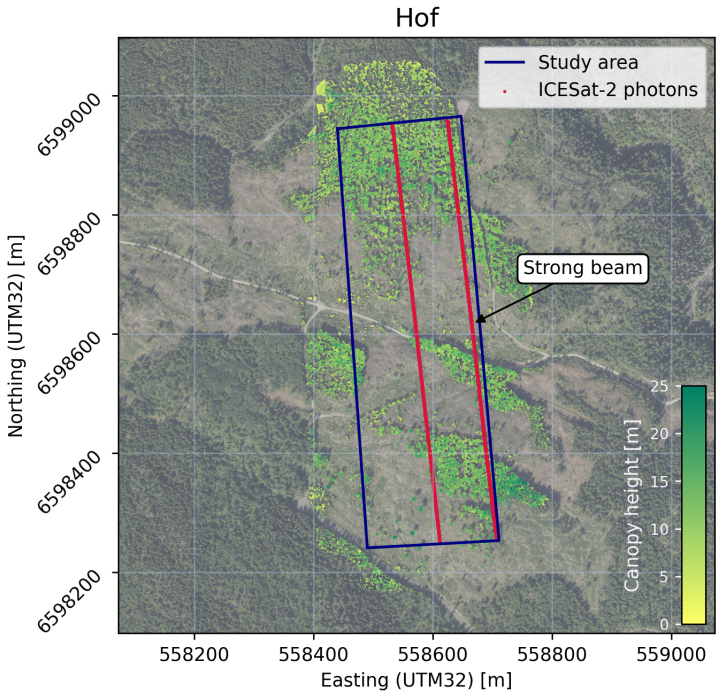

Figure 2.4: The site in Hof municipality.

Describe Hof

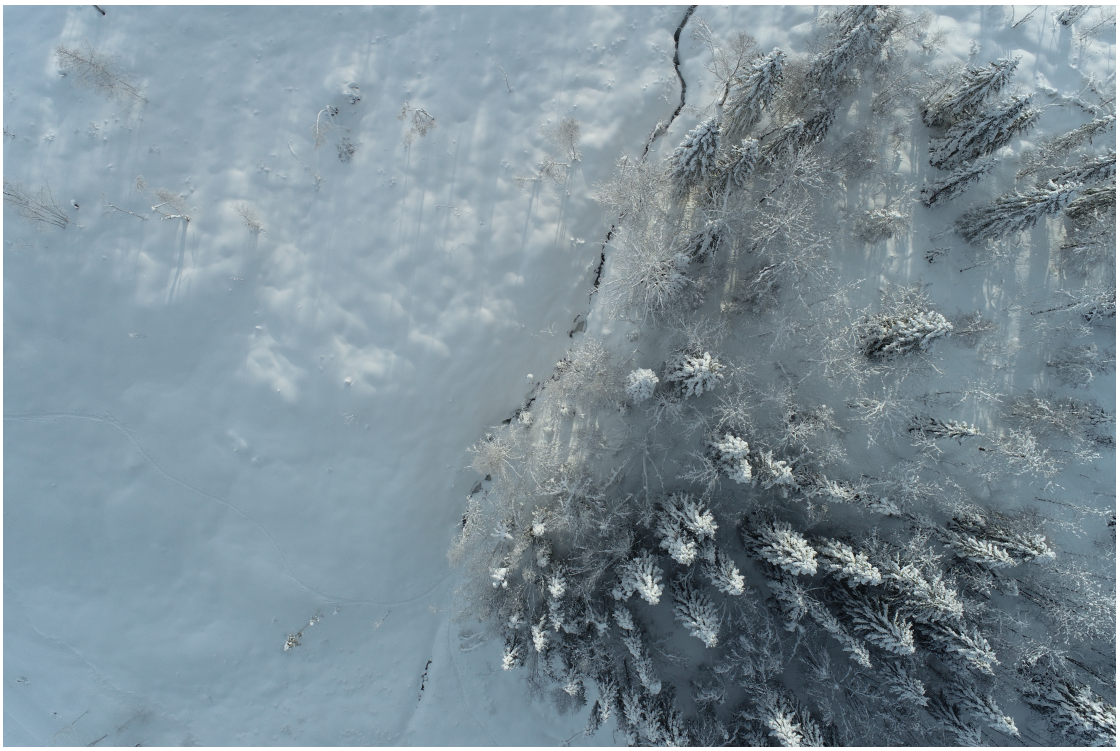

Figure 2.5: Sample image from the UAV flight, showing the canopy structure, covered in snow

3.3 Vikerfjell

Describe Vikerfjell

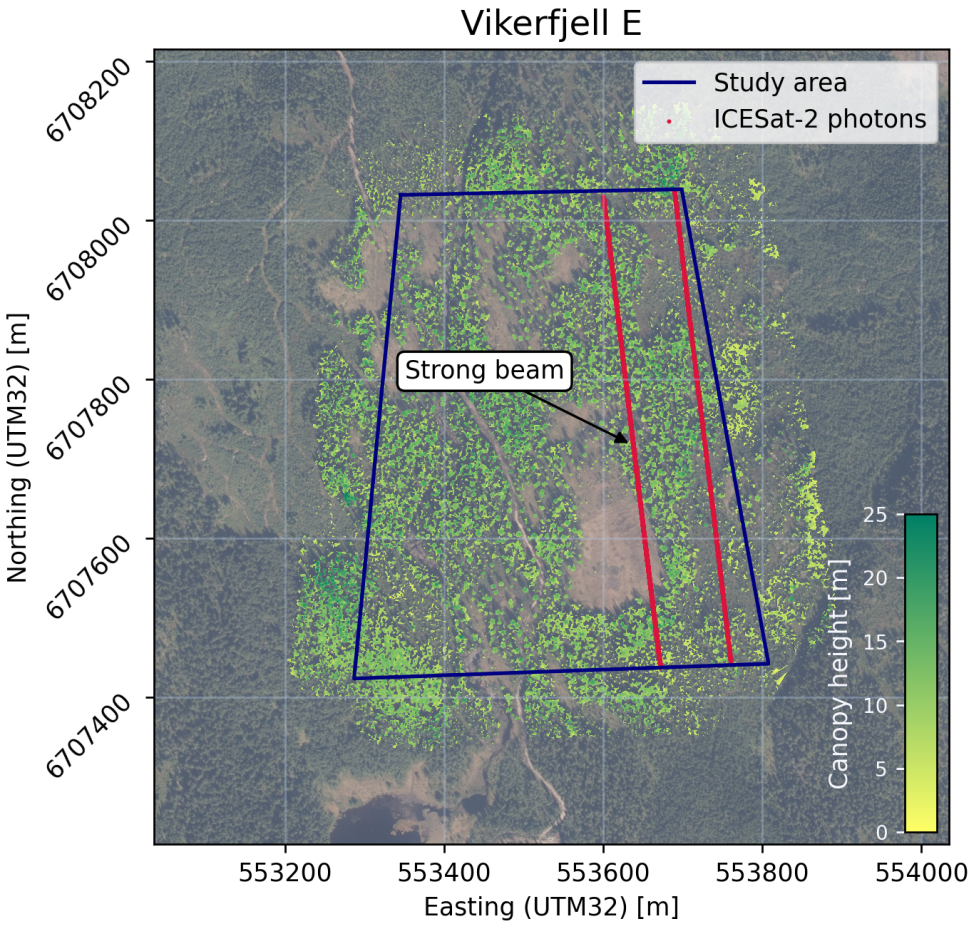

Figure 2.6: The eastern part of the site in Vikerfjell.

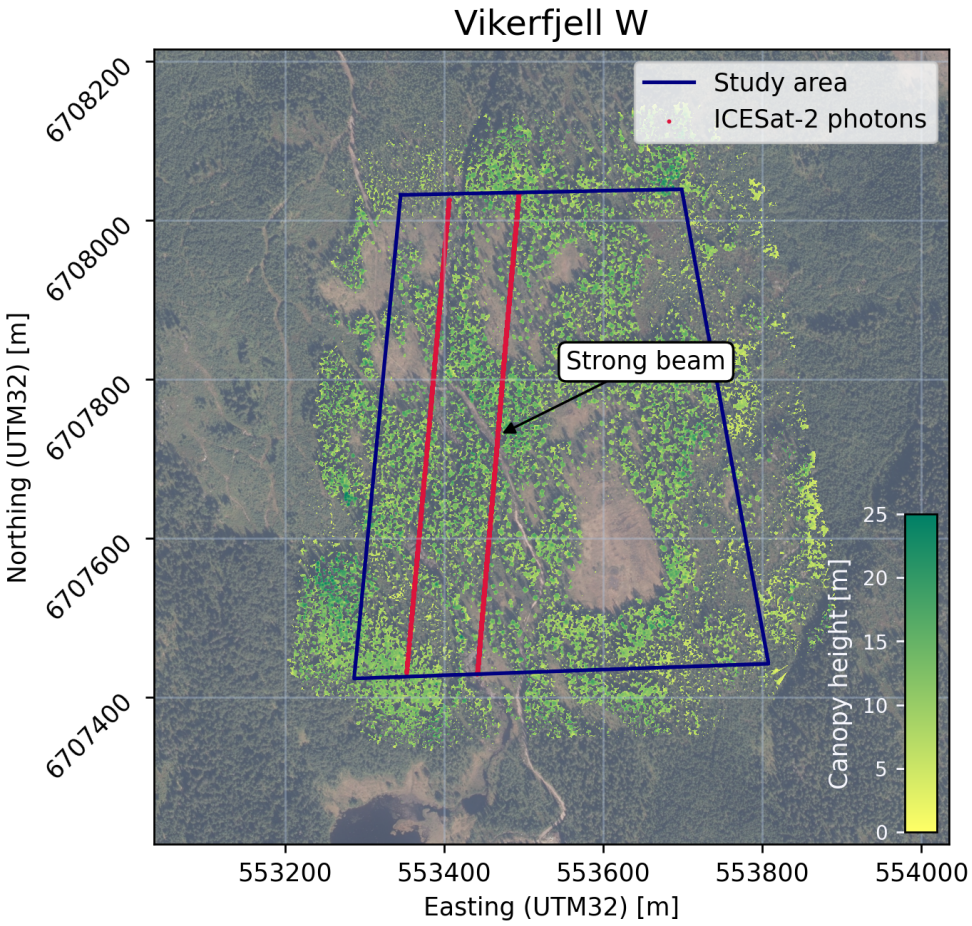

Figure 2.7: The western part of the site in Vikerfjell.

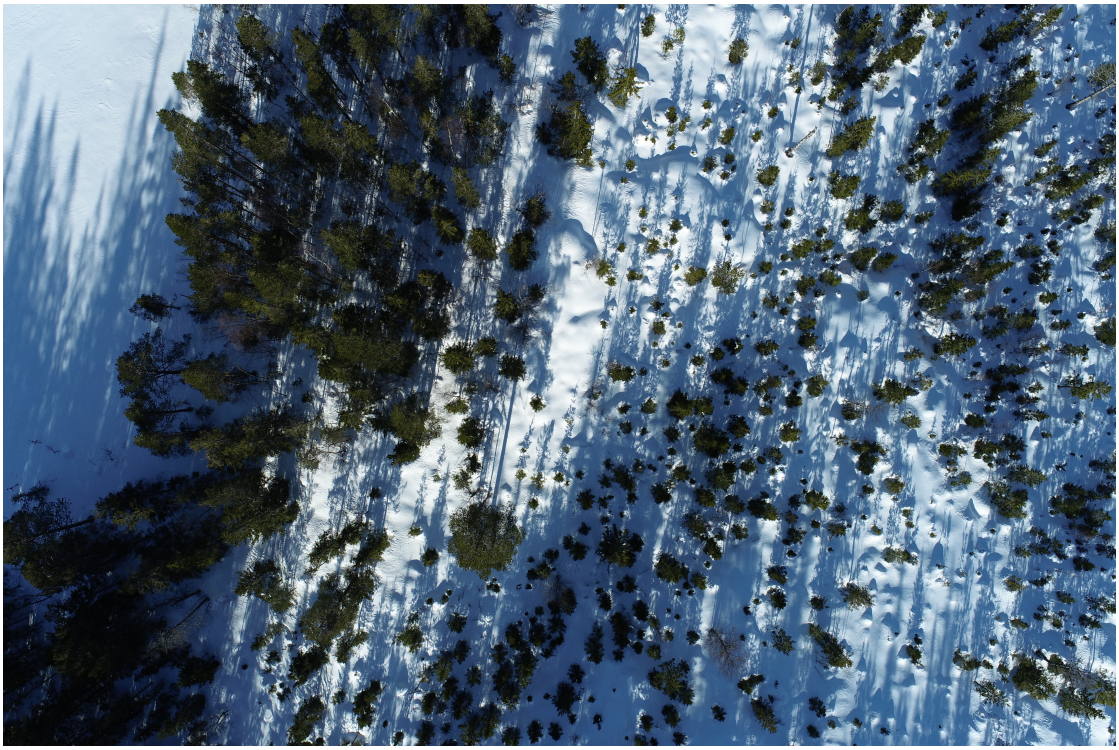

Figure 2.8: Sample image from the UAV flight in Vikerfjell, showing no snow cover on the canopy structure.

4 Data

4.1 ICESat-2 products

The available ICESat-2 mission generates a wide range of products from the photon returns, although the main purpose of the mission is to measure polar sea ice and land ice. A total of 23 different product indicators area listed at the *Data Products* section of the ICESat-2 mission site. This thesis is based on the *Global Geolocated Photon Data* (ATL03) and *Land Water Vegetation Elevation* (ATL08) products. This applies both to the data downloaded directly from the NSIDC and from Sliderule. Both products are available in the *Hierarchical Data Format* (HDF).

Global Geolocated Photon Data — ATL03

As the name suggests, this product contains the spatio-temporal location of each photon returned to the ATLAS instrument. The photons are "classified by signal vs. background, as well as by surface type (land ice, sea ice, land, ocean), including all geophysical corrections (e.g. Earth tides, atmospheric delay, etc...)" (NASA 2024). The photon products are provided in granules, each spanning several minutes.

Land Water Vegetation Elevation — ATL08

The ATL08 product is more specialised, and contains ground height and canopy surface. If the data permits it, this product also includes "canopy height, canopy cover percentage, surface slope and roughness, and apparent reflectance" (NASA 2024).

4.2 UAV-borne LiDAR point clouds

4.3 Norwegian elevation data

Digital Terrain Model

Geoid model

Chapter 3

Methods

Regime:	Low-orbit
Altitude:	496 km
Inclination:	92°
Period:	94.22 min
Velocity:	6.9 km/s
Repeat period:	91 days

Table 3.1: Orbital parameters of ICESat-2

5 Equipment

Necessary to describe the equipment? UAV yes, but IS2?

5.1 ICESat-2

All platform and instrument information in this subsection is referenced from Thomas A. Neumann et al. (2019), unless otherwise specified. The ICESat-2 mission is a part of *NASA's Earth Observing System* (2024). It consists of a single satellite in a near-polar orbit, carrying an The main instrument aboard the ICESat-2 satellite observatory is the ATLAS instrument, which is a photon-counting laser altimeter. The photons

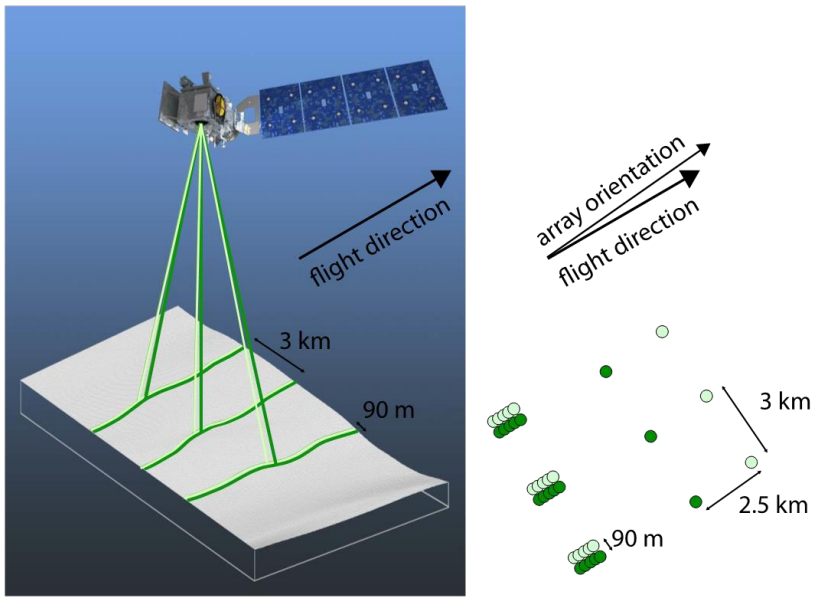

Figure 3.1: Spatial pattern of the ATLAS beams and footprints. From T. A. Neumann et al. (2023)

- Sensor properties
- Platform properties

5.2 DJI Matrice 300 RTK & DJI Zenmuse L1

Figure 3.2: DJI Matrice 300 RTK drone with DJI Zenmuse L1 on gimbal mount. (Image from DJI.com, accessed on 2024-04-09)

- Sensor properties
- Platform properties
- Why the M300

Table 3.2: DJI Zenmuse L1 sensor properties (From dji.com)

Operating temperature	-20° C to 50° C
Maximum Returns Supported	3
Ranging Accuracy (RMS 1σ)[2]	3 cm @ 100 m
Point Rate	Single return: max. 240,000 pts/s Multiple return: max. 480,000 pts/s

6 UAV data acquisition

Describe how the data was acquired. Considerations? (Weather, terrain, accessibility, etc.)

7 Software

The software used in this thesis is described in this section. All processing, except the initial post-processing of the UAV point clouds, was handled with open source programming libraries in either Python or R languages. The only exception is the initial post-processing of the UAV point clouds, which had to be done in the DJI Terra software due to the sensor outputs proprietary format. They were then saved to the LAS file format, which is open source and easy to work with (ASPRS 2019).

The actual processing steps, and parameters used, is further described in section 8.

7.1 DJI Terra

DJI Terra is a licensed software provided by the same manufacturer as the drone and Zenmuse L1 LiDAR instrument. Its simple user interface makes it easy to learn and use, but as it is not open-source software, the user has less control over — and insight into — what happens with the data. The software can generate secondary products like DEMs, but using them would make the results of this thesis harder to reproduce.

7.2 Jupyter

Project Jupyter (Kluyver et al. 2016) is responsible for the development of a suite of applications, like *JupyterLab*. *JupyterLab* is a coding interface which includes support of several file formats and also has a file explorer. It integrates well with *GitHub*, which makes version control simple. JupyterLab is especially suitable for working with *Jupyter* notebook files. These are interactive documents which can use both *R* and *Python* languages in dedicated cells, in combination with pure markdown content for annotation and longer texts. Each cell can show its output directly, while variables and functions are remembered across the entire document. All of the code in this thesis, be it *Python*, *R* or *bash* scripts (GNU 2007), is written using JupyterLab.

7.3 R

R is a programming language which is particularly suitable for statistical computing and data visualisation, and is available on many platforms (R Core Team 2023). R was also available, along with necessary libraries, on the university's high performance server, which significantly reduced processing time of the relatively heavy LiDAR point clouds.

lidR

The R library *lidR* is a powerful alternative for working with point clouds in the LAS/LAZ format, which makes it possible to use parallel processing to create — among other things — canopy height models (CHM) and digital elevation models (DEM) (Roussel et al. 2020).

terra

The *terra* library (Hijmans 2024) is useful for working with spatial raster files, like DEMs. Saving the DEM and CHM products to file was done with *terra*.

7.4 Python

Most of the processing in this thesis is written in Python. The language is easy to learn, and uses a vast amount of packages, for almost any use case. With a simple, English-like syntax and cross-platform basis it provides a great starting point for working with data science and visualisation (Van Rossum and Drake 2009). Several libraries are used in this thesis:

pandas

pandas (The pandas development team 2020) is the most widely-used library for data analysis in Python. It handles a wide range of tabular data formats, as well as time series and other structured data sets. Data is structured in DataFrames, which in turn consist of Series. pandas contains a multitude of classes and functions to analyse big datasets in a fast and flexible way. Pandas also makes it easy to export tables in LaTeX format, which are used in the writing of this thesis.

GeoPandas

GeoPandas is based on Pandas, but adds support for geospatial data and operations (Bossche et al. 2023). Examples include adding buffers and selecting data based on location, which is done in this thesis. GeoPandas uses GeoDataFrames as its basic data structure, which essentially is a Pandas DataFrame with an additional GeoSeries containing geometries (points, lines, polygons, etc.).

xDEM

Although in its early stages of development, xDEM is helpful when working with digital elevation models (DEM). In this thesis it is mainly used to extract elevation data from DEMs at the photon centre points, but it also provides functions for co-registration of DEMs with a method developed by Nuth and Kääb (2011).

SlideRule

Photons with the *Yet Another Photon Classifier* (YAPC) algorithm applied are downloaded from a web service API, through this *SlideRule* (Swinski et al. 2024) python client. YAPC (Sutterley 2022) is the basis of one of the filtering methods evaluated in this thesis.

NumPy

The *NumPy* library forms the basis of a lot of the computing operations in *Pandas*, but some operations in the processing in this thesis is done using NumPy functions directly. The library is one of the most widely used in the Python universe, built upon by a multitude of other libraries (Van Rossum and Drake 2009).

matplotlib

The visual presentations of the findings in this thesis is made with the library *matplotlib* (The Matplotlib Development Team 2023). *matplotlib* makes it easy to create simple data visualisation plots in Python, while also allowing users to modify their plots in more complex ways (Hunter 2007).

rasterstats

The rasterstats library (Perry 2024) makes it possible to extract raster information in several ways, including inside a polygon (zonal statistics). The ICESat-2 photons are defined with centre point geometry — while the photons have a surface footprint diameter of approximately 13 meters — and this library was helpful in getting DEM and CHM statistics inside the entire footprint.

8 Processing

Finish describing each processing step

8.1 Preprocessing of UAV data

In order to create the ground truth elevation and canopy height models, the drone data had to be reworked from point clouds into suitable raster formats for later use with the ICESat-2 and Norwegian Mapping Authority (NMA) data.

Point clouds

The data collected from the UAV for this thesis was initially in a proprietary point cloud format. To process it, the first step involved using the DJI Terra software for pre-processing. This software takes the point cloud data and other relevant mission files as inputs and converts them into a LAS point cloud file. In addition, the software can also perform ground point classification, which was applied during this process. Ideally, open-source software would have been used for this classification, but attempts with such software yielded difficult-to-manage results. The final output was a single LAS file, which was further converted to the LAZ format for a higher compression level of approximately 90 %. The conversion process utilised LASzip, which is included in the LAStools distribution (Isenburg n.d.).

DTM and CHM generation

The point clouds for all locations were then used to generate Digital Terrain Models (DTM) and Canopy Height Models (CHM). Point clouds are very useful for storing information about the LiDAR laser returns, but they can be large and cumbersome to work with. As the comparable datasets from NMA — like the geoid model and DTMs — are in raster format, it is useful to create data models in the same format.

The *R* library *lidR* was very useful to achieve this. It enables parallel-processing on several of its built-in calculation algorithms, and could therefore work through each file in relatively short time. Still, even when using one of the university's high-performance servers the processing of each file took approximately one hour. The lidR script first takes an input LAS/LAZ file, and each photon's x, y and z values — in addition to classification, number of returns, and return number — was selected for further processing. The script then returns the LAS file information. Example output is shown below, which describes a point cloud from the Drammen site:

```
class         : LAS (v1.2 format 3)
memory        : 10.7 Gb
extent        : 554808.1, 555267.5, 6628592, 6629570 (xmin, xmax, ymin, ymax)
coord. ref.   : WGS 84 / UTM zone 32N
area          : 299760 m²
points        : 318.3 million points
density       : 1061.85 points/m²
density       : 765.59 pulses/m²
```

The next step involves applying a rasterisation function on the data, which generates the DTM. This function allows the use of different algorithms to create DTMs, but the default triangulated irregular network (TIN) method was applied here. As the name suggests, this method creates a triangulated network of the points classified as ground

points, which makes it possible to get the height of any arbitrary point inside the network, using a bivariate function. The spatial resolution of the DTMs was set to 0.5 m.

The point clouds were also used to create the Canopy Height Models (CHM), which were integral for answering the research question in this thesis. The process is similar, but with a few important differences. Firstly, because some trees had been cut between the two data acquisitions (with and without snow), only the point clouds with snow cover could be used to create useful CHMs. Otherwise the models would not contain the same canopy structure as the ICESat-2 data. Second, three steps had to be completed before creating the output model:

1. The heights need to be normalised. In short, this process subtracts the ground height from each point, meaning that height values are now above-ground, instead of the ellipsoidal height.

2. The next step was to rasterise the canopy. This is similar to when rasterising the terrain, but a higher resolution of 0.2 m was used, in addition to another algorithm, more suited for the large height variability of a canopy model compared to a ground model.

3. Lastly, before exporting the canopy model, a lower height threshold of 2 m was applied. Any structure above that height should be counted as forest canopy, which the thesis is focusing on. (No man-made structures were present in the study sites.)

The CHM was finally exported. All files, both DTM and CHM, were stored in the *GeoTIFF* format, which is is a widely-used Open Geospatial Consortium standard format (OGC 2019).

The R processing script was applied to each point cloud file using a simple GNU bash script (see Appendix B), which also creates a log file with all console output and timings.

8.2 Data structuring and metrics extraction

The data is now ready for the main processing part. The data sources are in several different formats, making it beneficial to structure all relevant information in one dataset. *Pandas* DataFrames are easy to work with, Generate timestamp, merge info from ATL08, classify beams, interpolate data in gaps, apply pre-filtering (low signal conf, outside AOI, no heights)

8.3 Canopy height model

A function made to apply information from the canopy height model to each photon footprint. Implements the zonal_stats function from rasterstats, but also adds calculation of canopy density.

```
1 def chmStats(gdf, raster, diameter=12, stats=['mean'], density=False):
2     '''
3     Usage: chmStats(gdf, raster, diameter, stats, density)
4     Use a canopy height model (CHM) raster to calculate a set of stats
5     for the area within a photon footprint diameter. Defaults to
6     mean canopy height. With density=True, the canopy density is
      calculated
7     by also adding a 'count' and 'nan' column, which counts cells with
      values
8     and NaNs respectively.
```

```
9
10      gdf:        geopandas.GeoDataFrame with the photons. Must contain a
    geometry field
11      raster:     The CHM raster to use for calculations
12      diameter:   Photon footprint diamenter [m]
13      stats:      Which statistics to calculate. Alternatives derive from
    rasterstats.zonal_stats
14      density:    Whether to calculate canopy density. Will also add 'count
    ' and 'nan' to stats
15
16      Output:     The input gdf with new columns for stats
17      '''
18      from rasterstats import zonal_stats
19
20      if density:
21          stats += ['count', 'nan']
22      data = gdf.geometry.apply(
23          lambda pt: zonal_stats(pt.buffer(diameter/2), raster, stats=stats
    , all_touched=True)[0]
24      )
25      chdf = pd.DataFrame(list(data), index=data.index.tolist())
26      if density:
27          chdf['density'] = chdf['count'] / (chdf['count'] + chdf['nan'])
28      for col in chdf.columns:
29          chdf.rename({col: f'chm_{col}'}, axis=1, inplace=True)
30      gdf = gdf.merge(chdf, how='outer', left_index=True, right_index=True)
31      return gdf
```

8.4 Point classification methods

The following subsection will describe the three photon filtering methods used to remove the photons that likely do not represent the surface. The *threshold validation* and *Point grouping* methods are developed for this thesis, while the *YAPC* method uses the algorithm with the same name. The point grouping and YAPC methods are similar in principle, but YAPC is a far more complex algorithm.

Threshold validation

The threshold validation method is based on a function developed while working on this thesis. In short, the function uses incremental percentiles of the input snow depth values, to define a threshold for reasonable snow depths. The input is a *Pandas Series* containing snow depth values. The function then finds the corresponding value of incrementally higher percentiles — in steps of 5 % between 30 % and 100 %. A visualisation of this can be seen in fig. 3.3. When these values increase by more than a pre-defined margin per increment, this value is then set as the threshold. The function returns a new *Pandas Series*, containing boolean value *True* where the input value was below the threshold, and *False* if they are above it. The original Series' indices are kept, making it easy to concatenate the returned Series to the original DataFrame.

The margin is set to a default value of 0.1 m in the function, which was found by simple trial and error on the ICESat-2 snow depths from one of the study sites. The margin was changed until the mean snow depth corresponded closely to that of the UAV snow depths. After testing it on the other sites, this margin turned out to work equally well there, regardless of the differences in both terrain slope and canopy structure/density.

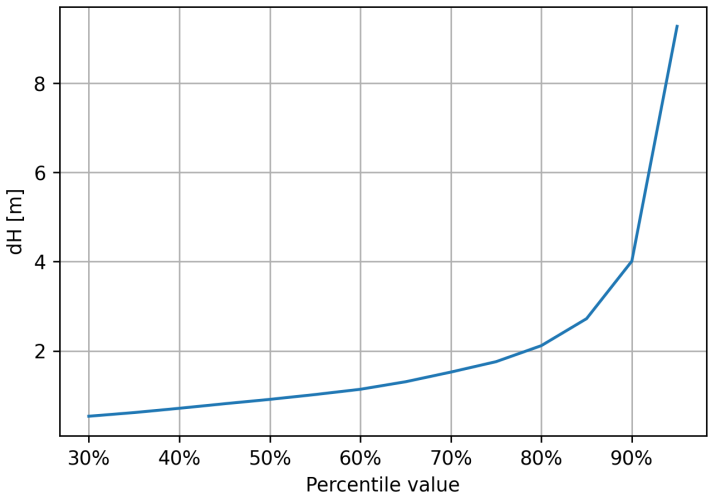

Figure 3.3: A visualisation of the change in the maximum acceptable snow depth value, as a function of the percentile value, as performed in *eval_ph*.

The principle behind this function is that when the snow depth values increase by a more than a certain margin per increment of percentile value, these snow depths are actually, more likely than not, something other than the surface. The function code is seen below:

```
def eval_ph(values, margin=.1):
    '''
    Usage: eval_ph(values, margin)
    Takes a set of dH (snow depth) values as input. Calculates a suitable
     percentile threshold to use for photon selection, and returns an
    equal-shape series with boolean True where photons are within
    threshold. (Preserves input indices.)

    values:    pandas.Series with dH values
    margin:    Max acceptable dH increase per 5 pct percentile increase

    Output:    pandas.Series with boolean True/False for valid/invalid
    snow depths
    '''

    valid = values
    x = np.arange(0.3, 1, .05)
    y = abs(values).quantile(x)
    y0 = y.iloc[0]
    for threshold in y:
        if threshold - y0 >= margin:
            valid[(values > threshold) | (values < 0)] = False
            return valid.astype('bool')
        y0 = threshold
    return values.astype('bool')
```

Point grouping

The principle of the point grouping method is that if there are enough photons in the immediate vicinity of each other, they are more likely to represent a surface than not.

The function lets the user define a lower threshold for the number of photon returns needed within a buffer, and the spatial extent of that buffer. This filtering is applied to the input photons with an inner function, before a rolling median is applied to the remaining photons. As with the threshold validation method, the values used in this function was found by trial and error until the results were optimal on one site, before applying the same values to all sites. The code for this function can be seen below:

```
def grpheights(gdf, hcol='h_ph', xybuff=1, zbuff=1, threshold=5, window
    =20):
    '''
    Usage: grpheights(gdf, hcol, xybuff, zbuff, threshold, window)
    For each photon in the data column hcol, use a 3D buffer of
    horizontal size xybuff and vertical size zbuff, to scan for other
    photons. If number of photons surpass threshold, the originial photon
    is assigned the mean height of the all photons in buffer. Otherwise
    np.nan is assigned. Finally, a a rolling median function is assigned
    to all photons.

    gdf:        geopandas.GeoDataFrame containing photon heights
    hcol:       The name of the photon height column
    xybuff:     Horizontal buffer size
    zbuff:      Vertical buffer size
    threshold:  Photon count threshold
    window:     Window to use for rolling median

    Output:     geopandas.Series with new photon height values
    '''

    def group(row):
        grp = gdf.loc[
            ( gdf.within(row.geometry.buffer(xybuff)) ) &
            ( abs(row.h_ph - gdf.h_ph) < zbuff )
            ].h_ph
        if len(grp) >= threshold:
            return grp.mean()
        else:
            return np.nan

    h_grp = gdf.apply(lambda row: group(row), axis=1)
    h_grp = h_grp[h_grp.notnull()]    # Drop NA rows
    h_grp = h_grp.rolling(window, closed='both', center=True).median()

    return h_grp
```

YAPC

This method uses the Yet Another Photon Classifier (YAPC) algorithm, which is developed by Jeff Lee at NASA's Goddard Space Flight Center (Sutterley 2023). It is a pure implementation of already-existing functionality, provided by the Python *SlideRule* library, described in section 7.4. The algorithm calculates a score for each photon, which is based on the number of nearest neighbours and a three-dimensional buffer. This is similar to what is done in the point grouping method above, but the calculation is more complex. It also allows the user to set a new threshold without recalculating anything, just by allowing a lower or higher score in further processing.

8.5 Snow depth calculations

8.6 Bias estimation

The accuracy of the point classification and ICESat-2 snow depth estimates are examined by comparing the ICESat-2 heights to the UAV heights.

8.7 Canopy vs. bias correlation

To assess whether the canopy density in the study sites affects the snow depth measurement errors (bias), a Pearson correlation coefficient r is calculated between these two variables. This coefficient can be used to indicate whether there is a linear correlation between two quantitative variables (Moore, McCabe and Craig 2012).

The Pearson correlation coefficient can be expressed as a the mean of the products of the standard scores of the two variables x and y, with n observations:

$$r_{xy} = \frac{1}{(n-1)} \sum_{i=1}^{n} \left(\frac{x_i - \overline{x}}{s_x} \right) \left(\frac{y_i - \overline{y}}{s_y} \right)$$

The returned values are

Chapter 4

Results

The processing results in a set of statistics and metrics, describing the ability of the ATLAS instrument to measure snow depths in vegetated terrain. This chapter will show each of the results from the processing, and explain how they are interpreted.

9 The effects of vegetation

Overall, the results of the analyses in this thesis show little to no correlation between canopy density and surface height (i.e. snow depth) measurement bias. Th

9.1 Overall

Explain the numbers in the results Overall subsection

After applying the methods for noise removal, the overall results show little to no correlation between canopy density and snow surface height measurement bias, with Pearson correlation coefficients for all sites and photon filtering methods are $|r| < 0.3$ (table 4.1). As seen in fig. 4.1, there are no indications that we have a non-linear correlation either.

Table 4.1: Statistics per method (both beams)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-0.968 m	2.095 m	3.906 m	-169%	662%	-0.18
Threshold	0.026 m	0.306 m	0.383 m	7%	60%	-0.10
Grouped	-0.221 m	0.305 m	0.722 m	-30%	115%	-0.07
YAPC	-0.077 m	0.438 m	0.594 m	-19%	78%	-0.00

Strong beam

Table 4.2: Statistics per method (Strong beam)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-0.996 m	2.093 m	3.881 m	-170%	637%	-0.18
Threshold	0.023 m	0.302 m	0.379 m	7%	58%	-0.08
Grouped	-0.226 m	0.335 m	0.696 m	-32%	113%	-0.17
YAPC	-0.077 m	0.440 m	0.593 m	-20%	77%	0.01

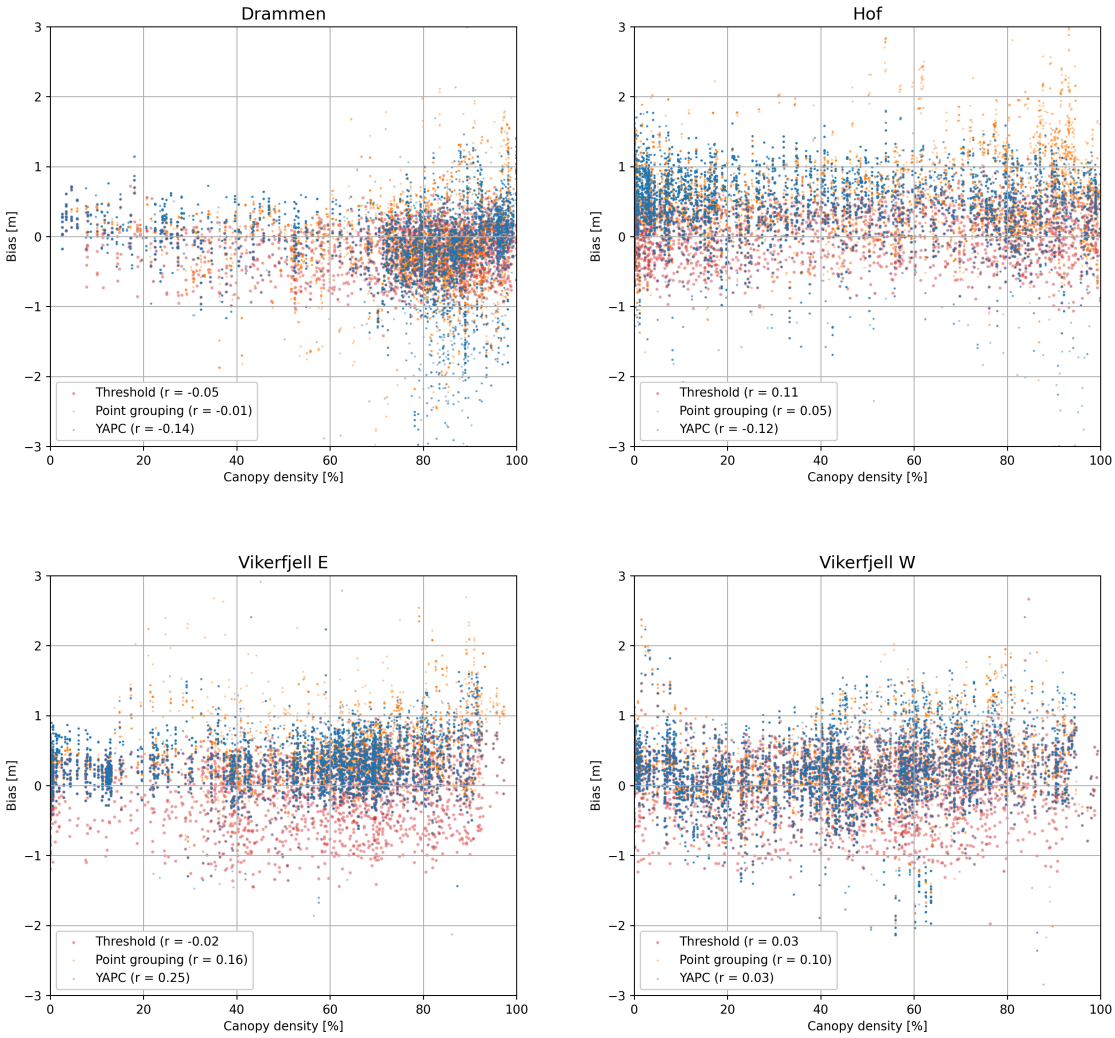

Figure 4.1: Correlation of height estimation errors (bias) in all the filtering methods used, versus the canopy density inside the photon footprint.

Weak beam

Table 4.3: Statistics per method (Weak beam)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-0.868 m	2.095 m	3.841 m	-152%	679%	-0.16
Threshold	0.065 m	0.336 m	0.419 m	10%	66%	-0.22
Grouped	-0.227 m	0.179 m	0.799 m	-30%	115%	0.00
YAPC	-0.047 m	0.466 m	0.660 m	-6%	100%	-0.03

Describe results for each site. How are they varying? Mention the influences on-site if relevant.

9.2 Drammen

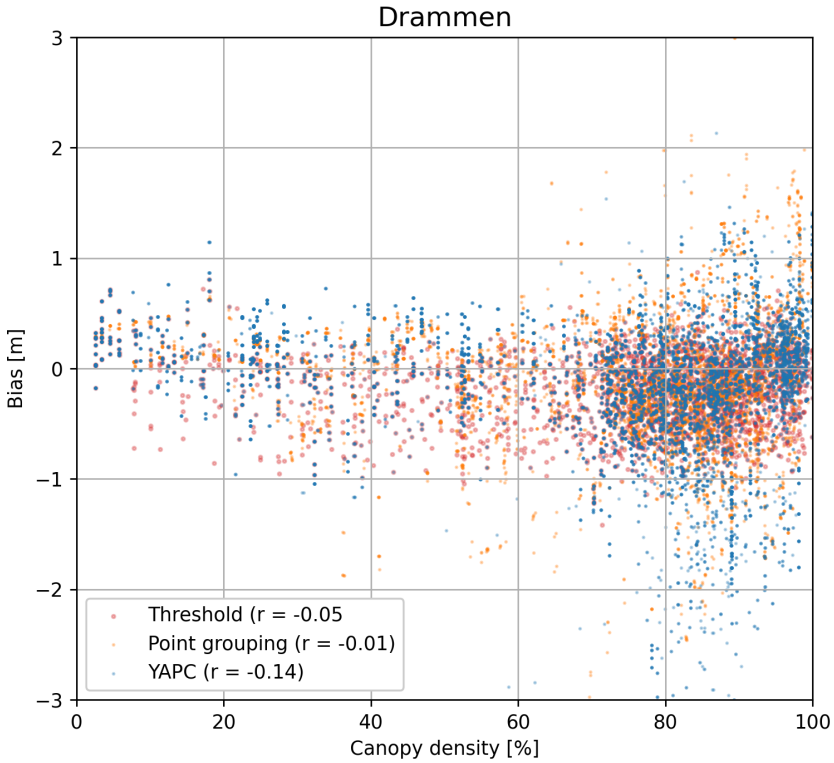

Figure 4.2: Correlation plot of all methods in the study site Drammen

Table 4.4: Statistics per method, Drammen (Both beams)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-1.773 m	3.914 m	6.379 m	-300%	1,081%	-0.13
Threshold	0.198 m	0.292 m	0.373 m	33%	63%	0.05
Grouped	0.123 m	0.227 m	0.677 m	21%	115%	0.01
YAPC	0.152 m	0.411 m	0.639 m	26%	108%	0.14

9.3 Hof

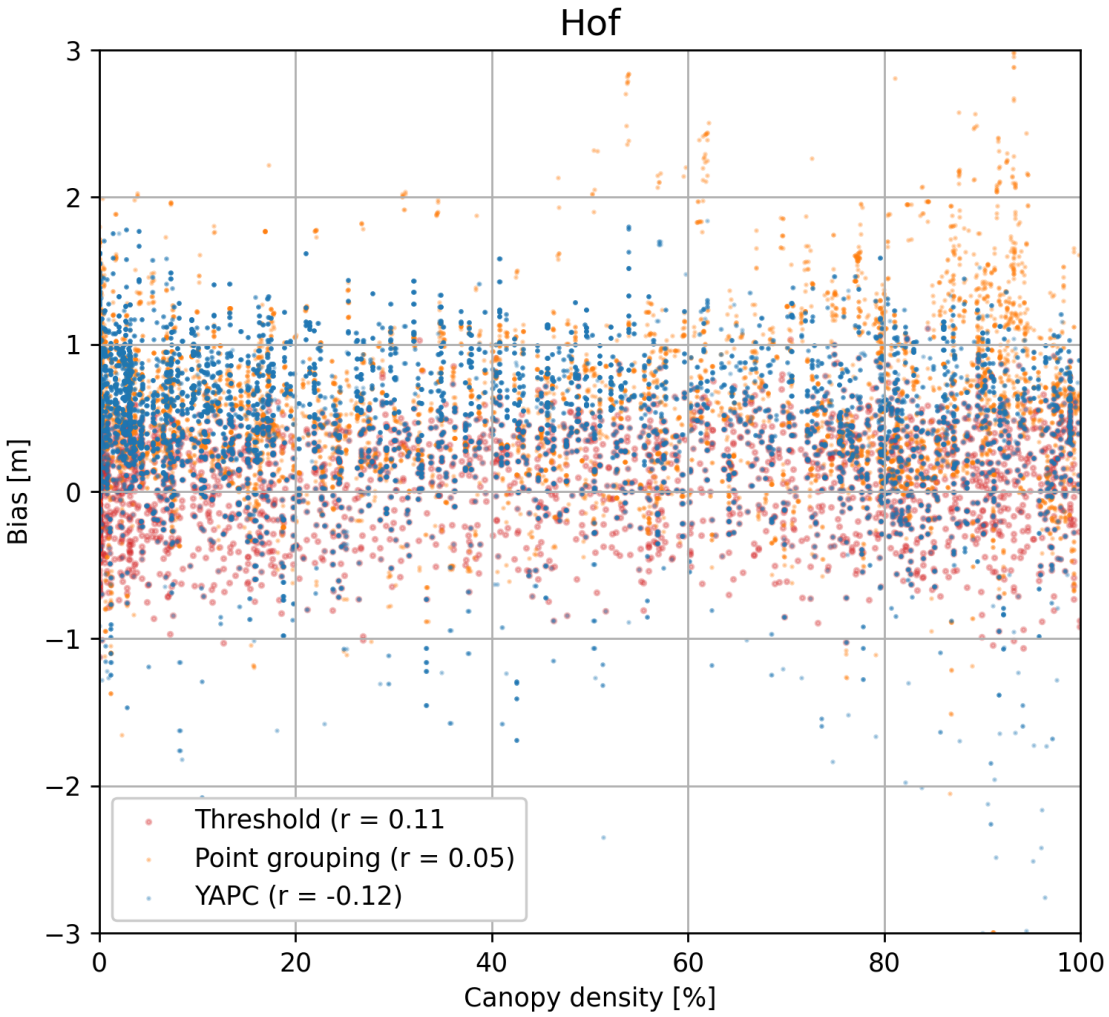

Figure 4.3: Correlation plot of all methods in the study site Hof

Table 4.5: Statistics per method, Hof (Both beams)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-1.226 m	2.343 m	4.087 m	-246%	820%	-0.21
Threshold	-0.088 m	0.306 m	0.365 m	-18%	73%	-0.11
Grouped	-0.622 m	0.465 m	0.802 m	-125%	161%	-0.05
YAPC	-0.483 m	0.579 m	0.682 m	-100%	141%	0.12

9.4 Vikerfjell East

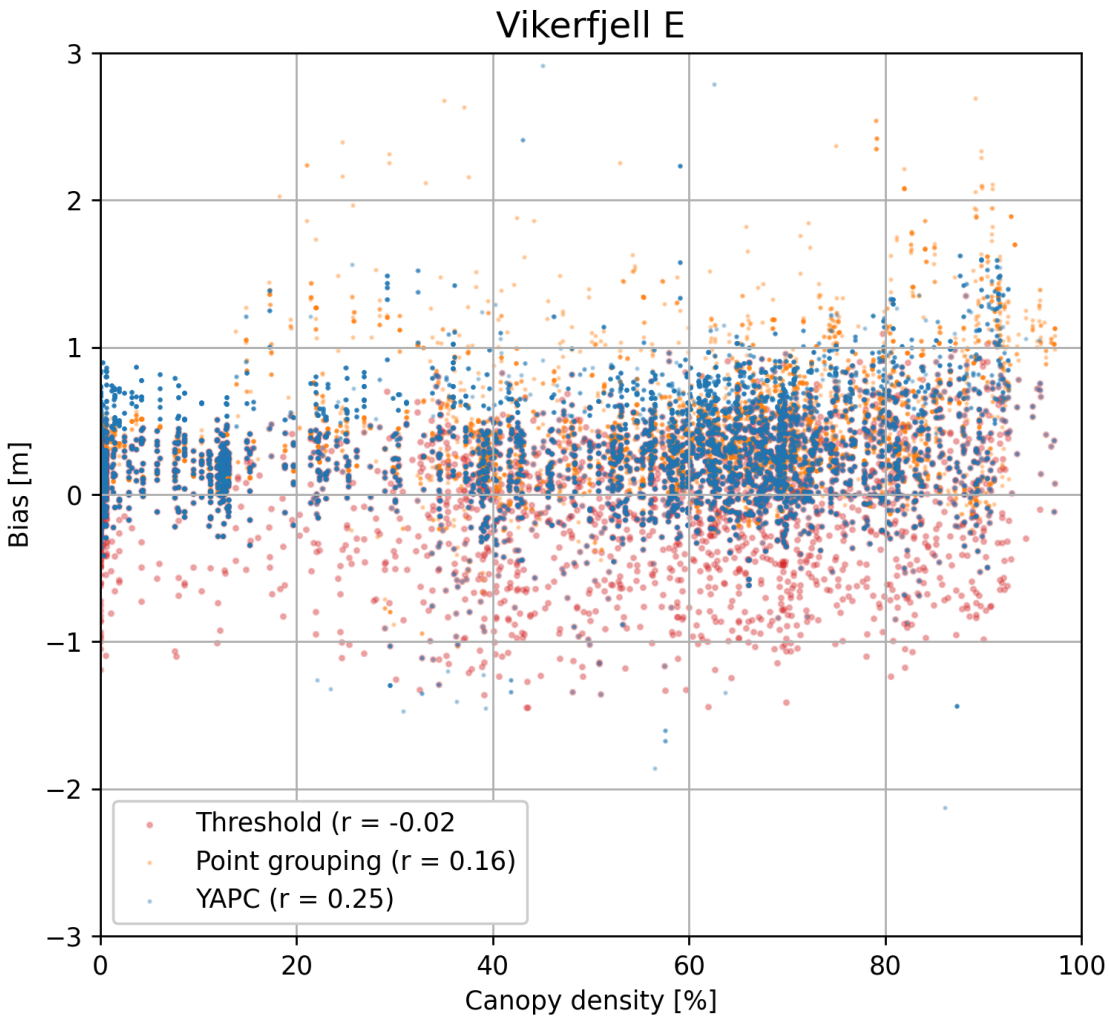

Figure 4.4: Correlation plot of all methods in the study site Vikerfjell East

Table 4.6: Statistics per method, Vikerfjell E (Both beams)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-0.638 m	1.189 m	2.311 m	-65%	235%	-0.16
Threshold	-0.063 m	0.302 m	0.386 m	-6%	39%	0.02
Grouped	-0.429 m	0.344 m	0.820 m	-44%	84%	-0.16
YAPC	-0.255 m	0.299 m	0.390 m	-26%	39%	-0.25

9.5 Vikerfjell West

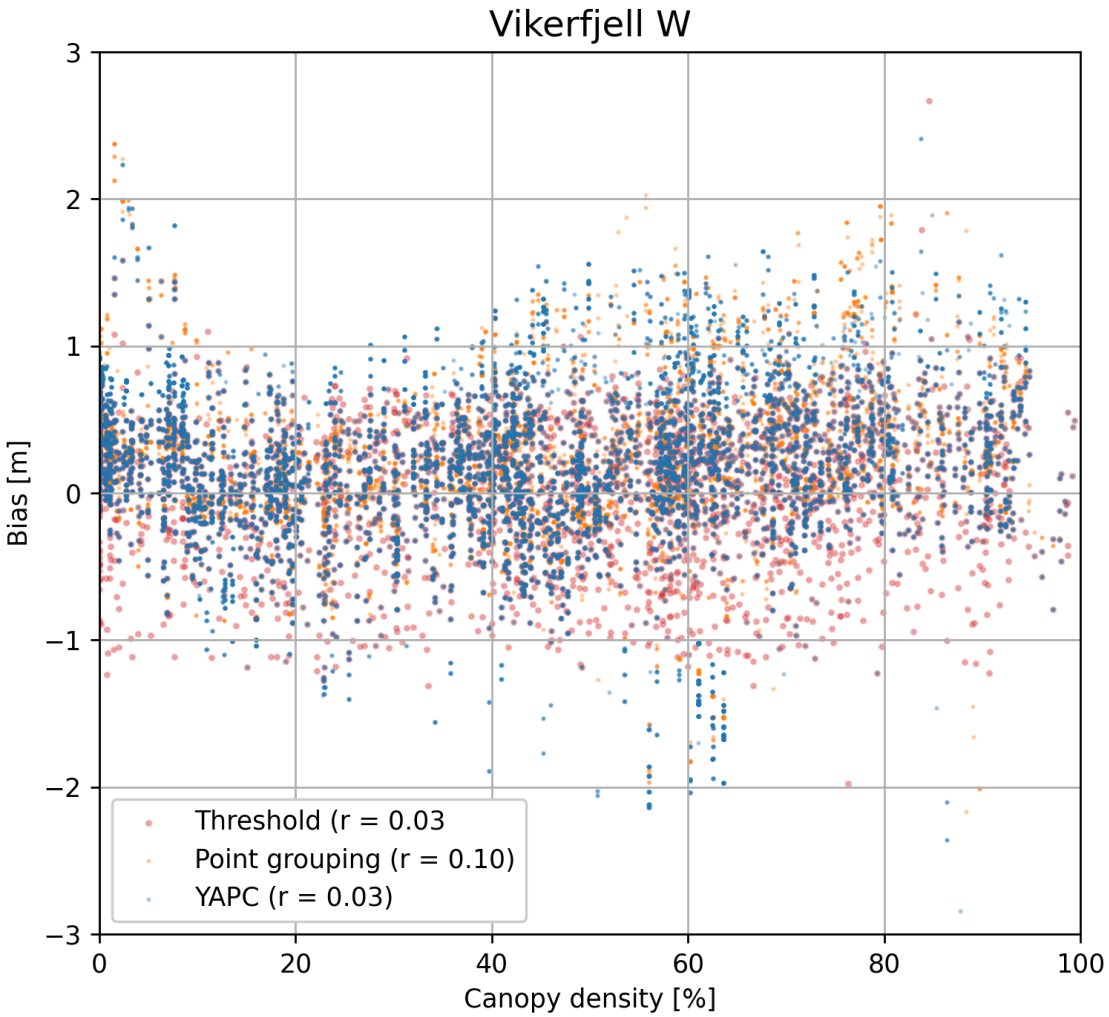

Figure 4.5: Correlation plot of all methods in the study site Vikerfjell West

Table 4.7: Statistics per method, Vikerfjell W (Both beams)

Method	Bias	MAE	RMSE	Rel. Bias	Rel. RMSE	Corr. veg.
Unfiltered	-0.316 m	1.244 m	2.874 m	-33%	299%	-0.10
Threshold	-0.063 m	0.347 m	0.434 m	-7%	45%	-0.03
Grouped	-0.297 m	0.353 m	0.659 m	-31%	69%	-0.10
YAPC	-0.186 m	0.409 m	0.555 m	-19%	58%	-0.03

10 Performance of filtering methods

Describe how each filtering method performs, and relevant influences on results

This section will contain plots and more detailed descriptions of how each filtering method performs.

10.1 Threshold validation

10.2 Point grouping

10.3 YAPC

Chapter 5

Discussion

11 Application

Do the results provide answers to the research question? Would other areas yield comparable results?

12 Workflow Review

Review the processing workflow and software used. What works, and what could have been better/more efficient?

13 Software Review

Chapter 6

Conclusion

14 Main findings

Is the research question answered?

15 Applications and future work

How can these findings be used, and what needs to be improved or further explored/tested

Bibliography

ASPRS (9th July 2019). *LAS Specification 1.4 - R15*. The American Society for Photogrammetry & Remote Sensing.

Bossche, Joris Van den et al. (June 2023). *geopandas/geopandas: v0.13.2*. Version v0.13.2. DOI: 10.5281/zenodo.8009629.

GNU, P (2007). *Free Software Foundation. Bash (3.2. 48)[Unix shell program]*.

Hijmans, Robert J. (2024). *terra: Spatial Data Analysis*.

Hunter, J. D. (2007). 'Matplotlib: A 2D graphics environment'. In: *Computing in Science & Engineering* 9.3. Publisher: IEEE COMPUTER SOC, pp. 90–95. DOI: 10.1109/MCSE.2007.55.

Isenburg, Martin (n.d.). *LAStools - efficient LiDAR processing software*. Version 171030, unlicensed.

Kluyver, Thomas et al. (2016). 'Jupyter Notebooks – a publishing format for reproducible computational workflows'. In: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. IOS Press, pp. 87–90. DOI: 10.3233/978-1-61499-649-1-87.

Luftfartstilsynet (2024). *Fly drones safely*. Luftfartstilsynet - Fly drones safely. URL: https://luftfartstilsynet.no/en/drones/veiledning/fly-drones-safely/ (visited on 17/01/2024).

Moore, David S., George P. McCabe and Bruce A. Craig (2012). *Introduction to the practice of statistics*. 7. ed., internat. ed. New York: Freeman [u.a.] 101-105. ISBN: 978-1-4292-8664-0.

NASA (2024). *Data Products | ICESat-2*. Data Products | ICESat-2. URL: https://icesat-2.gsfc.nasa.gov/science/data-products (visited on 11/04/2024).

NASA's Earth Observing System (2024). URL: https://eospso.gsfc.nasa.gov/ (visited on 09/04/2024).

Neumann, T. A. et al. (2023). *ATLAS/ICESat-2 L2A Global Geolocated Photon Data, Version 6*. DOI: 10.5067/ATLAS/ATL03.006.

Neumann, Thomas A. et al. (1st Nov. 2019). 'The Ice, Cloud, and Land Elevation Satellite – 2 mission: A global geolocated photon product derived from the Advanced Topographic Laser Altimeter System'. In: *Remote Sensing of Environment* 233, p. 111325. ISSN: 0034-4257. DOI: 10.1016/j.rse.2019.111325.

Nuth, C. and A. Kääb (29th Mar. 2011). 'Co-registration and bias corrections of satellite elevation data sets for quantifying glacier thickness change'. In: *The Cryosphere* 5.1. Publisher: Copernicus GmbH, pp. 271–290. ISSN: 1994-0416. DOI: 10.5194/tc-5-271-2011.

OGC (14th Sept. 2019). *GeoTIFF Standard*. URL: https://docs.ogc.org/is/19-008r4/19-008r4.html (visited on 14/04/2024).

Perry, Matthew (26th Mar. 2024). *perrygeo/python-rasterstats*. original-date: 2013-09-18T06:44:29Z.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Roussel, Jean-Romain et al. (15th Dec. 2020). 'lidR: An R package for analysis of Airborne Laser Scanning (ALS) data'. In: *Remote Sensing of Environment* 251, p. 112061. ISSN: 0034-4257. DOI: 10.1016/j.rse.2020.112061.

Sutterley, Tyler (June 2022). *tsutterley/yapc: v0.0.0.8*. Version 0.0.0.8. DOI: 10.5281/zenodo.6717591.

— (2023). *YAPC Overview — pyYAPC 0.0.0.8 documentation*. Rev. 87023d6e. URL: https://yapc.readthedocs.io/en/latest/getting_started/Overview.html#yapc-goals (visited on 17/04/2024).

Swinski, J. P. et al. (Mar. 2024). *ICESat2-SlideRule/sliderule: v4.3.1*. Version v4.3.1. DOI: 10.5281/zenodo.10797886.

The Matplotlib Development Team (Nov. 2023). *Matplotlib: Visualization with Python*. Version v3.8.2. DOI: 10.5281/zenodo.10150955.

The pandas development team (Feb. 2020). *pandas-dev/pandas: Pandas*. Version latest. DOI: 10.5281/zenodo.3509134.

Van Rossum, Guido and Fred L. Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace. ISBN: 1-4414-1269-7.

Appendix A

Python scripts

Include processing scripts? R script included now, as an example, but Python will be longer

1 **Data download**

2 **Data processing**

Appendix A. Python scripts

Appendix B

Pre-processing script

3 GNU bash script

4 R code

```
library(lidR)
library(tools)

# Enable verbose mode and (try to) remove progress bars for logging
options(lidR.verbose=TRUE, lidR.progress=FALSE)

# lidR uses 50 % of CPU threads by default. Reduce to 25 %
set_lidr_threads(get_lidr_threads()/2)

# Fetch command arguments
args = commandArgs(trailingOnly=TRUE)

# Check if correct number of arguments is provided
if (length(args) != 2) {
    stop("Please include input filename => 'las2dem.R <inFile> <outFile>'
    ")
    }

# Set variables from command arguments
inFile = args[1]
outFile = args[2]

# Start processing
cat("    a.  Open LAS file: ", file_path_sans_ext(basename(inFile)))
las <- readLAS(inFile, select="xyzrnc")

cat("\n    b.  LAS file information:\n")
print(las)

cat("\n    c.  Generating DTM\n")
dtm <- rasterize_terrain(las, res = .5, algorithm = tin())
terra::writeRaster(dtm, paste(outFile, "_DTM.tif", sep=""), filetype = "
    GTiff", overwrite = TRUE)

if (grepl("SnowOn", inFile, fixed=TRUE)) {
    cat("\n    d.  Generating canopy height model\n")
    nrm <- normalize_height(las, algorithm = tin())
    chm <- rasterize_canopy(nrm, res = .2, p2r(.2, na.fill = tin()))
    chm <- terra::clamp(chm, 2, values=FALSE)    # Set cell values < 2 m
    to NaN
    terra::writeRaster(chm, paste(outFile, "_CHM.tif", sep=""), filetype
    = "GTiff", overwrite = TRUE)
    }

cat("\n    e.  las2dem.R terminating successfully!\n")
```