Performance Analysis of PV Power Plants Across Norway

Developing a Practical Approach to Analyze Large-Scale PV Installations with Limited Metadata

MARTIN KREBS KRISTIANSEN

SUPERVISOR
Anne Gerd Imenes

University of Agder, 2023
Faculty of Engineering and Science
Department of Engineering and Sciences

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	Ja
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	Ja
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	Ja
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	Ja
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	Ja
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	Ja

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	Ja
Er oppgaven båndlagt (konfidensiell)?	Nei
Er oppgaven unntatt offentlighet?	Nei

Acknowledgements

This is the final project in my master's thesis. I want to turn my appreciation to the people who have helped me with guidance and answered questions. Firstly I would like to thank Anne Gerd Imenes, my supervisor from UIA. I am grateful for all the time and effort she has put off for meetings, answered questions, and read and corrected the report. I would also like to thank her for helping me find a relevant and exciting subject for my master's thesis and putting me in contact with Institute for Energy Technology (IFE). With her and IFE, the subject of this thesis was possible.
I am also grateful that IFE has used its time to contact Solecllespesialisten and get ahold of their data. I therefore also thank Solcellespeisalisten for sharing their data. Finally, in regards to IFE, I would specially thank Christoph Seiffert, in regards to answering questions about the dataset.

Abstract

This thesis examines hourly aggregated data from 501 photovoltaic (PV) installations, builds a better knowledge foundation about the geographical performance of PV systems in Norway, and provides a groundwork for how PV datasets with limited metadata can be analyzed. Metadata is supplemented with inferred tilt and azimuth by analyzing the power and irradiance relationship at different orientations, with 1° intervals. When tested with a known PV installation, the result shows a median accuracy of 12.2° and 14.1° for tilt and azimuth, respectively. To analyze the performance of PV installations, the power output data is filtered with a linear filter (RANSAC) and a polynomial non-linear filter. The latter shows promising results, as long as specific requirements regarding the number of available timestamps are available. Unknown capacity units are inferred by selecting highly probable units (W_p, kW_p, and MW_p) and finding highly probable specific yields. Installations, where highly probable specific yields are not found using these units have been removed from further analysis.

Sammendrag

Denne oppgaven undersøker timebaserte data fra 501 solcelleanlegg (PV) og bygger et bedre kunnskapsgrunnlag om den geografiske ytelsen til solcelleanlegg i Norge. Oppgaven gir også et grunnlag for hvordan solcelledatasett med begrenset metadata kan analyseres. Metadata er supplert ved å beregne tilt og asimut ved å analysere effekt- og solinnstråling i forskjellige orienteringer, med $1°$-intervaller. Resultatet er en median nøyaktighet på $12,2°$ og $14,1°$ for henholdsvis tilt og azimut. Resultatene er testet med en kjent PV-installasjon. For å analysere PV-installasjonene filtreres effektdataene med et lineært filter (RANSAC) og et polynomisk ikke-lineært filter. Sistnevnte viser lovende resultater, så lenge spesifikke krav til antall tilgjengelige tidsstempler er tilgjengelige. Ukjente kapasitetsenheter utledes ved å velge svært sannsynlige enheter (W_p, kW_p og MW_p) og finne svært sannsynlige spesifikke utbytte. Installasjoner der svært sannsynlig spesifikk utbytte ikke er funnet ved bruk av disse enhetene, er fjernet fra videre analyse.

- $E_{ac}(t)$ AC energy over time period Δt [kWh]
- $P_{ac}(t)$ AC power at time t [kW]
- $E_{dc}(t)$ DC energy over time period Δt [kWh]
- $P_{dc}(t)$ DC power at time t [kW]
- E_{cum} Cumulative energy output over time [kWh]
- t Time [s,h,min,M,Y]
- N Number of time steps [-]
- Y_r Reference yield [h]
- H_G Measured on-site irradiation [kWh/m^2]
- E_{STC} Reference irradiance at standard test conditions [kW/m^2]
- Y_f Final yield [h]
- P_{STC} Power produced under standard test conditions [kWp]
- Y_a Array yield [h]
- PR Performance Ratio [-]
- E_{poa} Irradiance on the plane of array [W/m^2]
- $NOCT$ Nominal operating cell temperature [°C]
- $temp_{cell}$ Cell temperature [°C]
- $temp_{air}$ Air temperature [°C]
- E_{NOCT} Irradiance at NOCT condition [W/m^2]
- P Power output [W]
- $TC(P_{(MPP)})$ Temperature coefficient at maximum power point [%/°C]
- GHI Global Horizontal Irradiance [W/m^2]
- BHI Direct (Beam) Horizontal Irradiance [W/m^2]
- DHI Diffuse Horizontal Irradiance [W/m^2]
- DNI Direct Normal Irradiance [W/m^2]
- K_d Daily diffuse fraction [-]
- DHI_{daily} Daily cumulative diffuse horizontal irradiation [kWh/m^2]
- GHI_{daily} Daily cumulative Global horizontal irradiance [kWh/m^2]
- R_b Geometric factor of direct irradiance on the tilted surface to the direct irradiance on the normal surface [-]
- a variable for incident angle of sunlight on the surface [-]

- b variable for solar zenith [-]
- F_1 Circumsolar brightness coefficient [-]
- F_2 Horizon brightness coefficient [-]
- α tilt angle [Degrees]
- β Azimuth angle [Degrees]
- θ Incident angle of the sun [Degrees]
- θ_z sun zenith angle [Degrees]
- δ Brightness sky condition [-]
- f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23} Numbers based on empirical data for the specific location [-]
- ρ albedo [-]
- $P_{norm}(t)$ Normalized power at time t [-]
- $P(t)$ Power at time t [W]
- $P(t)_{max}$ Maximum power at time t [W]
- $E_{poa,norm}(t)$ Normalized plane of array irradiance at time t [-]
- $E_{poa}(t)$ Plane of array irradiance at time t [W/m^2]
- $E_{poa}(t)_{max}$ Maximum plane of array irradiance at time t [W/m^2]
- $RMSE(\alpha, \beta)$ Root mean square error for given tilt (α) and azimuth (β) angles [-]
- i Time step [Variable]
- T Number of time steps [-]
- $AC_{norm}(\alpha, \beta, i)$ Normalized AC power for given tilt (α), azimuth (β), and time step (i) [-]
- N_u optimal number of iterations [-]
- p Probability for a successful fit in RANSAC [-]
- ω Probability of inliers in the data for RANSAC [-]
- n Required amount of data points to make an acceptable fit in RANSAC [-]
- F_{value} F-value for ANOVA [-]
- MSS_b Mean sum of squares between groups [-]
- MSS_w Mean sum of squares within groups [-]
- Q_1 Lower quartile [Variable]
- Q_3 Upper quartile [Variable]
- IQR Interquartile range [Variable]
- HSD Turkey Honestly Significant Difference [Variable]

- M_i Mean of group i [Variable]
- M_j Mean of group j [Variable]
- MS_W Mean square within groups [Variable]
- H Kruskal-Wallis H-value [-]
- U Mann-Whitney U-value [-]
- w_g Wilcoxon signed-rank test statistic [-]
- α_s Significans level [-]

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The surplus of energy production is forecasted to diminish up until 2030 in both Norway and the Nordic region. A deficiency in energy production causes the need for import during peak periods. There is already a power deficit in tight situations today in the Nordic region. There are significant uncertainties in the growth of demand up until 2030, but an expectation made by NVE is 2-6 GW. The forecasted increase in supply is expected to come mainly from solar and hydropower. However, the increase is only expected to be 0.6 GW in the winter months by 2030. This leads to an expectation of increased power deficiency in the time to come [1]. Table 1.1 shows the expected increase in grid-connected solar photovoltaic (PV) energy in Norway and the Nordic region.

Table 1.1: Expected solar PV capacity increase in Norway and the Nordic region. Source: NVE [1]

Area	**Year**	**Installed Capacity [GW]**
Norway	2021	0.3
	2025	0.7
	2030	1.8
Nordic region	2021	3.8
	2025	9.4
	2030	12.6

As the grid-connected installed capacity is forecasted to grow six-fold by 2030, and solar PV energy mainly depends on available solar irradiance and location, the knowledge of expected power output is essential for investors, owners, and grid regulators. Much extensive data analysis has been done on the performance of PV installations in Europe; however, a gap remains in the literature regarding large-scale PV analysis using real-world data for the Norwegian climate. Norway is located in the northern parts of Europe. Sunlight is therefore received at a steeper angle, and fewer sunlight hours and less irradiance are received; as a cause of this, the snow amount is also higher.

Solcellespesialisten is a large supplier of complete solar systems in Norway and delivers systems to housing, industry, agriculture, and a solar park. They have provided facilities with a yearly estimated production of up to 860,000 kWh [2]. Production records from these facilities have been saved and stored by Solcellespesialisten. Consequently, they have a vast amount of real-world solar data from the Norwegian climate. This project has been conducted in collaboration with Solcellespesialisten and The Institute for Energy Technology (IFE), a leader in solar PV research in Norway. IFE is looking for more information on the solar industry in Norway and has therefore identified Solcellespesialisten's database as a valuable resource.

1.2 Motivation

Critical challenges face the energy supply network in the near future, with a forecast of a 2-6 GW increase in demand by 2030 and only a 0.6 GW increase in production during the winter months. Together with the growth of grid-connected PV installations expected to be six-fold over the next decade, as much information as possible is needed to predict the demand/supply of the power grid at the end of this timeline. Therefore, understanding the performance of PV based on geolocation is a critical factor for the demand/supply prediction and economical bankability of projects. With Solcellespesialisten offering its dataset for further studies, the motivation for this master thesis becomes to analyze a real-world dataset and develop a practical procedure for data analysis.

1.3 Problem Statement

This thesis aims to provide a method for analyzing real-world datasets. Therefore, this thesis addresses the following problem: How can a large number of PV installations be analyzed with limited data? This includes solving challenges regarding missing data information and analyzing the data. To address the overarching problem, the following sub-questions are made:

- What are the challenges in working with real-world data containing unspecified or inconsistent measurement units, and how can they be addressed?
- How can shortcomings in data and metadata be overcome?
- Are there any regional differences in the performance of PV installations across Norway?

1.4 Limitations and Assumptions

The data from Solcellespesialisten lacks information on installation tilt and azimuth, leading to an investigation into determining tilt and azimuth from production data. The utilized solution includes curve fitting the power output of a selection of optimal days over the course of a year and the irradiance for every possible plane in 1° increments. However, this method assumes that all panels in a PV installation have the same orientation, which may lead to inaccurate results in cases where this assumption does not hold.

The data contains unspecified units of measurement, such as power, timezone, temperature, and capacity. The timezone has been determined by comparing the sunrise and sunset times with the power data's start and end times and other methods. The temperature measurement appears to be an offset, likely representing inverter temperature versus air temperature. A non-linear method has been utilized to filter the power data to include decreased efficiency at higher temperatures. The capacity unit is theorized to be in Wp, kWp, or MWp; the correct value is determined by calculating the specific yield for the different units and utilizing the most likely result. Capacity units that are not logged in Wp, kWp, or MWp have not been adjusted to the correct unit of measurement and will give false results if not detected and removed from the dataset. Another dataset limitation is that each PV installation is limited to a maximum duration of one year. This means some PV installations are not included in the analysis due to lack of time. The geographical distribution is also uneven, with most PV installations near major cities. Local irradiance measurements are also not included, leading to satellite data usage.

The data from Solcellespesialisten was made available on 13.03.2023, which limited the available time for the analysis. As a result, a majority of the time was spent in the initial research phase, including testing methods on known datasets, such as data from the installation at the University of Agder and reading literature. This allowed progress to be faster once the data was made available. In addition, the knowledge that tilts and azimuth would not become available came on 21.03.2023, with tilt and azimuth being absent from the dataset; an addition of a procedure to locate the tilt and azimuth was included in the theses, resulting in less time in other parts of the thesis.

1.5 Thesis Structure

This thesis is structured as follows: Chapter 2 presents the theory used to process the result. Chapter 3 details the previous research done in this field. Chapter 4 explains the method used to gather and analyze results. Chapter 5 includes various ways the data has been manipulated to ensure good quality. Chapter 6 shows the results and discusses some specific results, and Chapter 7 contains a broader discussion of the methods used, challenges, and some comparisons to the literature review. Finally, to conclude the thesis, chapter 8 presents the conclusion.

Chapter 2

Theory

2.1 PV Energy Output

2.1.1 Energy

Energy is defined as the integral of power over time, as described in equation 2.1 for AC energy, and 2.2 for DC energy [3, p. 3-5].

$$E_{ac}(t) = P_{ac} * t = \int P_{ac}(t)dt \tag{2.1}$$

$$E_{dc}(t) = P_{dc} * t = \int P_{dc}(t)dt \tag{2.2}$$

A PV system's cumulative energy output over time is defined by equation 2.3.

$$E_{cum} = \sum_{t=1}^{N} E(t) \tag{2.3}$$

E_{cum} is the cumulated energy over the given period, it can either be AC or DC power, depending on if the momentary measurement of energy E is in AC or DC, N is the duration of the period [4].

2.1.2 Yield

The yield of a PV plant can be measured in multiple ways, quantified in terms of reference yield, final yield, and array tiled as specified below.

Reference Yield (Y_r)

Reference yield compares the measured on-site irradiation with the irradiation at standard test conditions (STC), as described in equation 2.4. Y_r is the reference yield and describes the theoretical maximum convertible energy available [3, p.278-280], [4], [5].

$$Y_r = \frac{H_G}{E_{STC}} \tag{2.4}$$

In equation 2.4, H_G is the measured on-site irradiation (in kWh/m^2), and E_{STC} is the reference irradiance at standard test condition (1 kW/m^2).

Final Yield (Y_f)

The final yield describes the energy produced at the AC side, divided by installed peak capacity, as described in equation 2.5. It represents the installation's hours at STC conditions to generate the recorded energy. The final yield includes the generator losses(L_C). Generator losses can be caused by factors such as high module temperature, shading, ohmic losses, and not operating at maximum power point [3, p.278-280], [4]

$$Y_f = \frac{E_{ac}(t)}{P_{STC}} \tag{2.5}$$

In equation 2.5, Y_f is the final yield, $E_{ac}(t)$ [kWh] is the energy produced on the AC side, and P_{STC} [kWh] is the power produced under Standard test conditions.

Array Yield (Y_a)

Array yield is similar to the final yield, except that it refers to the energy produced at the DC side of the inverter; therefore, the generator losses(L_C) are not included. The array yield is described in equation 2.6 [3, p.278-280].

$$Y_a = \frac{E_{DC}(t)}{P_{STC}} \tag{2.6}$$

In equation 2.6, Y_A is the generator yield, $E_{DC}(t)$ [kWh] is the energy produced on the DC side, and P_{STC} [kWh] is the power produced under standard test conditions.

2.1.3 Performance Ratio

Performance ratio (PR) measures how efficiently the PV plant utilizes the available irradiation. Equation 2.7) describes the relationship between equation Y_f (from equation 2.5) and Y_r (from equation 2.4) [3, p.279-281].

$$PR = \frac{Y_f}{Y_r} \tag{2.7}$$

2.1.4 Solar Irradiance and Power Output Relationship

The module's temperature correlates with the air temperature and irradiance. Other factors affecting the temperature include windspeed, available cooling, and construction. This section demonstrates the nonlinearity of power output with increased cell temperature and irradiance. Nominal operating cell temperature (NOCT) is a standard to assess PV panels. The conditions are $E_{poa} = 800W/m^2$, ambient temperature of 20°, and windspeed of 1 m/s^2. The NOCT temperature is described in the datasheet of the PV module and varies depending on the technology and module. Equation 2.8 is a simplified estimation of the cell temperature that assumes a linear increase in temperature with irradiance. $temp_{cell}$ in equation 2.8 is the cell temperature and, $temp_{air}$ is the air temperature [3, p.147-149].

$$temp_{cell} = temp_{air} + (NOCT - 20°) \cdot \frac{E_{poa}}{E_{NOCT}} \tag{2.8}$$

With $temp_{cell}$ the actual power can be estimated using equation 2.9, where P is power, P_{STC} is power at STC, $TC(P_{(MPP)}$ is the temperature coeficient (TC) at maximum power point (MPP) [3, p.147-149].

$$P = P_{STC} \cdot [1 + TC(P_{(MPP)}) \cdot (temp_{cell} - 25°)] \tag{2.9}$$

2.2 Solar Irradiance and Resource Data

Solar irradiance data is needed for the calculation of PR. Irradiance can be measured locally with equipment such as pyranometers. When such equipment is unavailable, other options, such as measurements with satellite data, can be used.

2.2.1 Solar Irradiance: GHI, BHI, DHI, and DNI

Multiple factors are influential in how much irradiance reaches the PV panel. As the irradiance hits the atmosphere, some will not enter due to reflection on the atmosphere's boundary. Another reason for lower irradiance is the absorption of light by molecules. The irradiation that enters and does not get reflected or absorbed may change direction due to scattering effects. Scattering effects occur when the irradiation hits dust particles and other aerosols. When the irradiance changes direction, it is classified as diffuse irradiation. This diffuse irradiation can unevenly distribute the irradiance. Due to these factors, there are multiple classifications of irradiance measurements per surface unit. The difference between them is the travel path of the irradiance and the impact angle. The most commonly used classifications are Global Horizontal Irradiance (GHI), Direct (Beam) Horizontal irradiance (BHI), Diffuse Horizontal Irradiance (DHI), and Direct Normal Irradiance (DNI) [3], [6].

Diffuse horizontal irradiance has interacted with some form of aerosol and changed direction from a straight path from the sun. In some locations, like Glasgow, the DHI might contribute more to the total irradiance than direct irradiance for a year [3]. Direct normal irradiance is irradiance that has traveled in a straight path; It is measured on a normal plane (perpendicular) to the sun. Direct Horizontal Irradiance (BHI) is similar to DNI, except that it is measured in the perpendicular plane. Global horizontal irradiance is the combined effect of direct and diffuse irradiance measured on a horizontal surface [3], [6].

2.2.2 Albedo and Ground Reflection

In addition to direct and diffuse irradiation, there is the effect of albedo. Albedo is a reflective property of materials. As irradiance hits the ground, the material of the ground decides how much of the irradiance is reflected. Albedo can therefore impact the total irradiance on a given surface. The tilt of the panel decides how much this affects the total irradiance—a steeper angle results in more irradiation due to the albedo effect. Typical values for different surfaces include grass at 0.25, lawn changing between 0.18 to 0.23, forest altering between 0.05 and 0.18, tarmac at 0.15, concrete within the range of 0.2 to 0.3, fresh snow from 0.8 to 0.9, and aged snow at 0.45 [3, p.37-38].

2.2.3 CAMS Radiation Service

CAMS (Copernicus Atmosphere Monitoring Service) gathers and provides information on atmosphere conditions, including but not limited to CO2, CH4, pollen, and irradiance. CAMS radiation service's goal is to fulfill the needs of national policy developments and the requirements of third-party commercial use [7]. The quality of the data is assured with tests against independent observations. CAMS radiation services offer two primary services: CAMS all-sky radiation services and CAMS clear sky radiation service. Only CAMS all-sky radiation services have been utilized in this thesis. CAMS all-sky radiation service's newest model is Heliosat-4. It can generate data from 2004 up until two days ago. The data can be delivered with a time resolution of one min, 15 min, hourly, one day, and one month. Heliosat-4 generates data for the latitude and longitude between $-66°$ and $66°$. The data is interpolated to the chosen location. The data is calculated using aerosol, water vapor, and ozone data from CAMS global forecasting system and satellite observations, together with ground elevation and albedo. The calculation process mainly consists of look-up tables, where all aforementioned data is used. The output data includes two main categories, clear sky, and horizontal measurements, including GHI, BHI, DHI, and DNI measurements [6]–[10].

Satellite Data

Satellite-derived irradiance data is created differently based on the method used. The basics are, however, similar. A satellite in orbit takes pictures that are analyzed. The pictures are often taken at different wavelengths to distinguish different features, such as visible light ($\approx 0.65\mu$m) and infrared ($\approx 11.0\mu$m). Infrared images can be used to detect water vapor. A combination of reactance on visible light images and infrared brightness temperature can be used to detect clouds; combining these images allows for height and density detection. Photos taken at different times can also be compared, as a baseline of non-cloudy environments is beneficial [11], [12]. The equations for calculating the irradiance differ for different methods; Heliosat-4 mainly uses look-up tables [9].

2.3 Inference of Tilt and Azimuth

2.3.1 Daily Diffuse Fraction

The daily diffuse (K_d) is a fraction defined by the DHI and GHI at a specific location and time. The DHI and GHI values are integrated values over a day. The daily diffuse fraction is a factor that ranges from 0 to 1, describing the sky's clarity, 1 being full cloud cover, while 0 is a no-cloud environment. Equation 2.10 describes the mathematical expression of the daily diffuse fraction when the DHI and GHI are the integrated sums of the day [13].

$$K_d = \frac{DHI_{daily}}{GHI_{daily}} \tag{2.10}$$

Where K_d is the daily diffuse fraction. DHI_{daily} is the daily cumulative diffuse horizontal irradiation [kWh/m^2], and GHI_{daily} is the daily cumulative Global horizontal irradiance [kWh/m^2] [13].

2.3.2 Plane-of-Array Irradiance Calculation

Weather data from satellites or other off-site methods are often recorded in GHI, DHI, and DNI components. As the performance calculations for PV require the irradiation in the plane of tilt and orientation, the Perez model transposes the components into the plane of array irradiance (E_{poa}), and is implemented in the pvlib library [14]. Equation 2.11 is the mathematical model used by `pvlib.irradiance.get_total_irradiance` [14] . The Perez anisotropic sky model was developed in 1990 and has been widely used for its accuracy and efficiency [15], [16].

$$E_{poa} = DNI \cdot R_b + GHI[(1-F_1)(\frac{1+\cos\beta}{2}) + F_1\frac{a}{b} + F_2\sin\beta] + DHI \cdot \rho(\frac{1-\cos\beta}{2}) \quad (2.11)$$

In equation 2.11, R_b is a geometric factor of direct irradiance on the tilted surface to the direct irradiance on the normal surface. a is the incident angle of sunlight on the surface variable, and b is the solar zenith variable. a is defined in equation 2.12, and b is defined in equation 2.13. F_1 is the circumsolar brightness coefficient, and F_2 is the horizon brightness coefficients; they are defined in equation 2.14 and 2.15 respectively. β is the tilt angle measured from the horizon [15], [16].

In the equation 2.12, θ is the incident angle of the sun. While in equation 2.13, 2.14, and 2.15, θ_z is the zenith angle [15], [16].

$$a = max(0°, cos\theta) \quad (2.12)$$

$$b = max(\cos 85°, \cos\theta_z) \quad (2.13)$$

In equation 2.14 and 2.15, f_{11}, f_{12}, f_{13}, f_{21}, f_{22} and f_{23} are numbers based on empirical data for the specific location, δ is the sky brightness condition, The original presentation [15] of the model has two different datasets for these empirical data [15], [16].

$$F_1 = max[0, (f_{11} + f_{12}\delta + \frac{\pi\theta_z}{180})f_{13}] \quad (2.14)$$

$$F_2 = f_{21} + f_{22}\delta + \frac{\pi\theta_z}{180}f_{23} \quad (2.15)$$

2.3.3 Normalization

Normalization is a process that adjusts data amplitude by dividing each data point by a fixed and known variable. This is particularly useful when comparing two datasets with correlated changes but different amplitudes. Normalizing the data transforms the amplitude into a value between 0 and 1, allowing for easier comparison between datasets with different amplitudes. Equation 2.16 shows the power normalization, and Equation 2.17 demonstrates the plane of irradiance normalization. In both cases, the values are normalized using the maximum value of the corresponding variable during the respective day [13].

$$P_{norm}(t) = \frac{P(t)}{P(t)max} \quad (2.16)$$

$$E_{poa,norm}(t) = \frac{E_{poa}(t)}{E_{poa}(t)max} \quad (2.17)$$

2.3.4 Root Mean Square Error (RMSE)

The RMSE is a widely used metric to evaluate differences between two datasets. Equation 2.18 calculates the root of the average difference between the normalized plane of array irradiance ($E_{poa,norm}$) and the normalized AC power (AC_{norm}) data, for different tilt (α) and azimuth (β) angles, i being the timestep, and T being the number of timesteps [13].

$$\text{RMSE}(\alpha,\beta) = \sqrt{\frac{1}{N}\sum_{i=1}^{N_u}(E_{poa,norm}(\alpha,\beta,i) - AC_{norm}(\alpha,\beta,i))^2} \quad (2.18)$$

2.4 RANSAC

Random Sample consensus (RANSAC) is a method of finding inliers and outliers in a dataset. The algorithm selects an arbitrary data point within the dataset and fits the model. It then determines the number of outliers and repeats for selected iterations. Parameters in the analysis include the minimum samples needed (n) to make up a fit. This is a minimum of 2-datapoints for a 2D plot and 3 for a 3D plot. The optimal number of iterations (N_u) to get the correct inliers can be estimated based on the type of data used and its expected probability that a given datapoint is an inlier (ω) using equation 2.19 [17]–[19].

$$N_u = \frac{\log(1-p)}{\log(1-\omega^n)} \tag{2.19}$$

In equation 2.19, N_u is the number of iterations needed, p is the probability for a successful fit, ω is the probability of inliers in the data, and n is the required amount of data points to make an acceptable fit [19].

2.5 Statistical Analysis

To detect statistical differences between two groups, there are two main categories of tests; parametric and nonparametric. The main difference is the assumption of the underlying data. The normality of the dataset can be confirmed in two ways: First, if the filesize is small, multiple sample sets are needed. Alternatively, if there is enough information in one dataset, a conclusion can be made that the data is normally distributed, and the underlying data is said to be normally distributed. In these cases, parametric tests are best suited, as these are made with this data in mind. On the other hand, nonparametric tests do not look at the mean data, as in parametric tests but consider a magnitude made from the data. This causes information to be lost and is therefore seen as inferior in the use case if the data is normally distributed. Still, it is an effective procedure if the normal distribution criteria are not met. The significance level (α_s) describes how certain one should be before disregarding the null hypothesis. The equation to determine α_s is formulated in equation 2.20, where the confidence level is presented as a decimal [20].

$$\alpha_s = 1 - \text{confidence level} \tag{2.20}$$

2.5.1 One-way Analysis of Variance (ANOVA)

One-way Analysis of Variance (ANOVA) is used to compare multiple datasets and is a parametric test. The null hypothesis of ANOVA is; The samples in the groups are from the same population. If the null hypothesis is proven wrong, the data came from different populations, and the data is considered statistically different. The F-value decides the statistical difference. The F-value is calculated using equation 2.21. The assumptions that have to be met for the analysis to be valid are; independent samples, equal variance in the sample population, data measured on an interval or ratio scale, the data must be distributed normally, independent errors and errors that are normally distributed, and the variance in the different groups have to be equal. It is important to note that perfect scenarios rarely happen in the real world and that ANOVA is robust in cases where the normality assumption is somewhat disobeyed [21, p. 221-234], [22].

$$F_{value} = \frac{MSS_b}{MSS_w} \tag{2.21}$$

MSS_b is the mean sum of squares between the groups, and MSS_w is the mean sum of squares within groups. F-values being higher indicates differences between the groups. To abandon the null hypothesis, the calculated F-value has to be higher than the critical value. The critical value can be found using lookup tables but is usually automated with software [21, p.221-234].

2.5.2 Turkey's Method

Turkey's Method, also known as Turkey's fences, is a widely used filtering technique for identifying and removing outliers. It performs best if the data follows a normal distribution [23]. Outliers can alter results in modeling and statistical analysis; Turkey's rule addresses this issue by identifying data that falls outside a multiple of the interquartile range (IQR). IQR is defined by data between Q_1 (25^{th} percentile) and Q_2 (75^{th} percentile) and represents 50% of the data. The upper and lower limits in equation 2.22 describe the point at which outliers start [23].

$$\begin{aligned}
&\text{Upper limit} = Q_3 + 1.5 \cdot IQR, \\
&\text{Lower limit} = Q_1 - 1.5 \cdot IQR, \\
&\text{Inter Quartile Range (IQR)} = Q_3 - Q_1, \\
&Q_1 = 25^{\text{th}} \text{ percentile}, \\
&Q_3 = 75^{\text{th}} \text{ percentile}
\end{aligned} \tag{2.22}$$

2.5.3 Turkey HSD

Turkey Honestly Significant Difference (Turkey HSD) is a standard posthoc procedure after a one-way ANOVA test. Thurkey HSD is a pairwise comparison, where the knowledge of what pairs differ is found. The test's criteria are the same as for the one-way ANOVA test. The null hypothesis is that there is no difference between the groups. The method uses equation 2.23 to determine the HSD [24].

$$HSD = \frac{M_i - M_j}{\sqrt{\frac{MS_W}{N}}} \tag{2.23}$$

where the difference between the tested pairs is $M_i - M_j$, number of groups is N, and MS_W is the mean square Within. To reject the null hypothesis, the absolute difference between the means of the two groups must be greater than the calculated HSD value [24].

2.5.4 Kruskal-Wallis H-test

Kruskal-Wallis H-test is often seen as the nonparametric alternative to one-way ANOVA. Kruskal-Wallis H-test is, therefore, also a statistical test to determine if at least two groups differ. In cases with more than two groups, the result does not reveal which one is different. To use Kruskal-Wallis H-test, a couple of parameters must be met. The groups must be independent and should, therefore, consist of two or more categories. There should be no relationship between the observation in each group, and one participant can not be present in another group. The result of the data also needs to be analyzed according to the data distribution. If the data in the groups are similar, the groups' median should decide what groups might deviate. If the groups are not equally distributed, the Kruskal-Wallis H test should compare the mean. The P-value is then found using look-up tables or software. If the P-value is less or equal to the chosen α value, the null hypothesis can be proven wrong [25], [26, p. 216-217].

$$H = \frac{12N}{N(N+1)} \sum_{i=1}^{k} \frac{R_i^2}{I_i} - 3(N+1) \tag{2.24}$$

In equation 2.24, N is the sum of all samples, k is total samples, and R_i is the sum of ranks. The rank is found by merging all data and ranking by size. Finally, I_i is the sample size in the i group [26, p. 216-217].

2.5.5 Mann-Whitney U-test

Mann-Whitney U is a nonparametric test; it tests for significant differences between two groups. Mann-Whitney U-test is computed with Equation 2.25, and 2.26. The lowest U value of the two equations is used, it is compared to a look-up table to determine if the U value indicates a significant difference, but software is also often used [27].

$$U_1 = n_1 n_2 + \frac{1}{2} n_1(n_1 + 1) - R_1 \tag{2.25}$$

$$U_2 = n_1 n_2 + \frac{1}{2} n_2(n_2 + 1) - R_2 \tag{2.26}$$

In equation 2.25 and 2.26, n_1 and n_2 are the sample sizes (n_i) of each group. R_1 and R_2 are the number of ranks (R_i) in each group.

2.5.6 Dunn's Test

Dunn's test is an appropriate procedure following the Kruskal-Wallis H-test. Dunn's test allows for checking more than two groups and studying which groups differ. Dunn's test utilizes Mann-Whitney U-test to test each pair of groups. Dunn's procedure allows the comparison of the results. Equation 2.28 illustrates the equation for Dunn's method [28].

$$z_i = \frac{y_i}{\sigma_i(1)} \tag{2.27}$$

In equation 2.27, z_i is the z-score; as in earlier tests, this value is used to find the p-value by a look-up table or software. $y_i = \overline{W}_A - \overline{W}_B$, where $\overline{W}_A$ and $\overline{W}_B$ is calculated by $\overline{w_{g_i}} = R_i/n_i$ for each group. σ_i is defined in equation 2.28

$$\sigma_i = \sqrt{\{\frac{N(N+1)}{12} - \frac{\sum_{s=1}^{r} \tau_s^3 - \tau_s}{12(N-1)}\}(\frac{1}{n_1} + \frac{1}{n_1})} \tag{2.28}$$

Where N is the sum of all samples, τ_s is the number of tied values for the specific value in the current rank (s), and r is the number of tied ranks. n_1 and n_2 is the sample sizes (n_i) for each group [28].

2.5.7 Error Rate Control

Multiple pairwise comparisons increase the probability of a type 1 error (falsely rejecting the null hypothesis). Multiple procedures have been developed to address this issue; Bonferroni and Benjamini-Hochberg's procedures both address this issue. Bonferroni is a conservative procedure severely limiting the chance of type 1 error. The Bonferroni divides the α by the number of groups; this dramatically decreases the P-value needed to reject the null-hypotheis [28, p. 292-299]. Benjamini-Hochberg procedure is less strict and contols the FDR (false discovery rate). The goal of the ajustment is the make the probability of a type 1 error less than α [29].

Chapter 3

Literature Review

3.1 Performance

Performance analysis of PV installations is a widely available topic in the research literature, where different studies have looked at different climates and technology. There are multiple metrics to compare system performance. Some of the most common methods are energy output [kWh], final yield (Y_f), performance ratio (PR), specific yield [kWh/kWp], energy density (E_d), system efficiency (η_{sys}), and array capture losses (LC) [4], [30]–[34].

3.1.1 Performance in Norway

Norway is located in northern Europe at a primary latitude and longitude of 62°N and 10°E, respectively. As an effect of this, the radiation has to pass through a relatively thick atmosphere, compared to locations closer to the equator, which is not beneficial. As it is in the northern hemisphere, the Southern direction of the panels is beneficial. The northern parts of the county receive the least amount of irradiance, where annual horizontal irradiance is typically measured from 700 kWh/m^2 to 900 kWh/m^2. The photovoltaic potential is higher in the southern parts of the country, where the measurement can reach as high as 1100 kWh/m^2 [35], [36]. The variance in seasonality is significant; this goes for both the northern and southern parts. The best locations in the country during the summer with up to 5500 Wh/m^2 each day may not see more than 350 Wh/m^2 each day during the winter [35], [37].

A study done in 2015 used a PV installation from the southern parts, specifically Ås, Norway. The authors of [31] found an expected annual specific yield of 931.6 kWh/kW$_\mathrm{p}$ and an average daily final annual yield of 2.55 kWh/kW$_\mathrm{p}$ together with a performance ratio of 0.83. The system's tilt was 37°, and orientation was South [31]. Another study published the same year used a similar technique to find the expected performance of a PV installation in Agder, Norway. The authors of [38] found a similar annual specific yield of 950 kWh/kW$_\mathrm{p}$ and a performance ratio of 0.79 in the year 2014. The systems tilt was 20°, and azimuth was 200° (nearly South) [38].

The author of [39] has also done specific yield analysis on multiple PV installations in Norway. The author found specific yields such as; 800 kWh/kW$_\mathrm{p}$ (Hordanland; close to Bergen, azimuth 190°, tilt $0° - 70°$), 937 kWh/kW$_\mathrm{p}$ (Agder; Kristiansand, azimuth 200°, tilt 20°), 895 kWh/kW$_\mathrm{p}$ (Hedemark; Evenstad, azimuth 161°, tilt 34°), 810 kWh/kW$_\mathrm{p}$ (Akershus; Vestby, azimuth 90°/270°, tilt 10°), 723 kWh/kW$_\mathrm{p}$ (Oslo, azimuth 90°/270°, tilt 10°/20°), 710 kWh/kW$_\mathrm{p}$ (Oslo, azimuth 90°/270°, tilt 10°).

Solargis has simulated maps over the solar potential in Southern Norway; the World Bank Group has published the data, which can be seen in Figure 3.1 and 3.2. The map shows the coastal area between Kristiansand to Fredrikstad as one of the most optimal places, together with the mountainous middle. On the other hand, the west coast from Sandnesjoen and upwards appear to be the least optimal area [37]. Overall their simulated results aligned with what is found from actual PV installations [31], [35], [36], [38], [39].

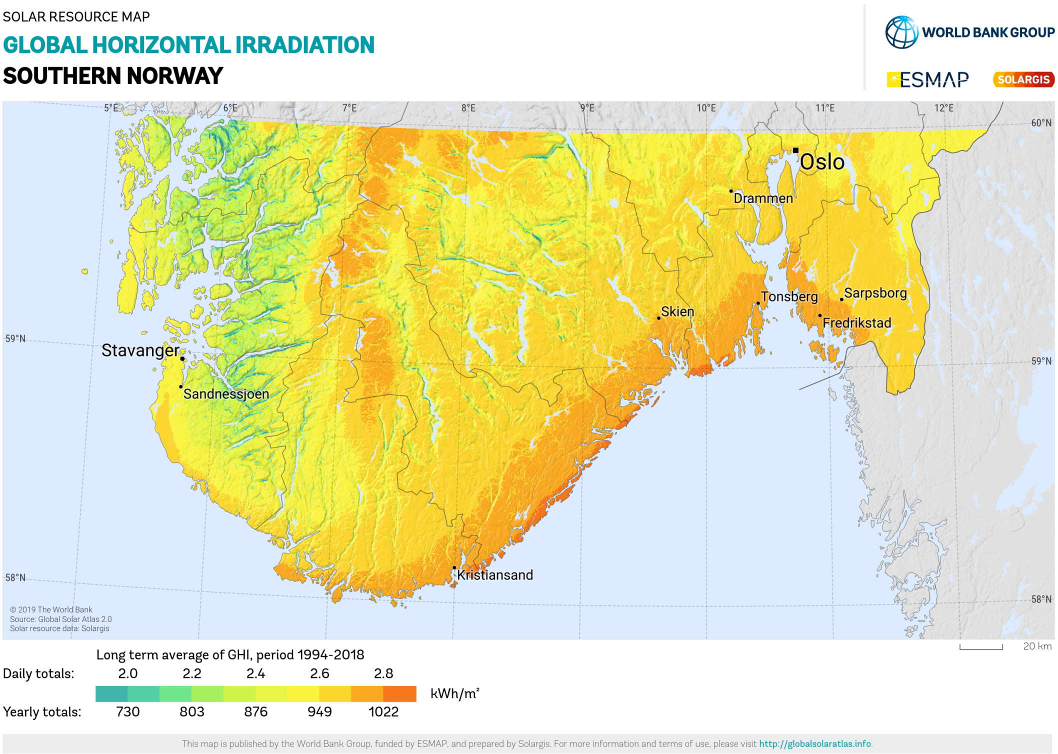


Figure 3.1: Partial global horizontal irradiation map in Norway: published by the World Bank Group, and visualized by Solargis. Source [37]

.

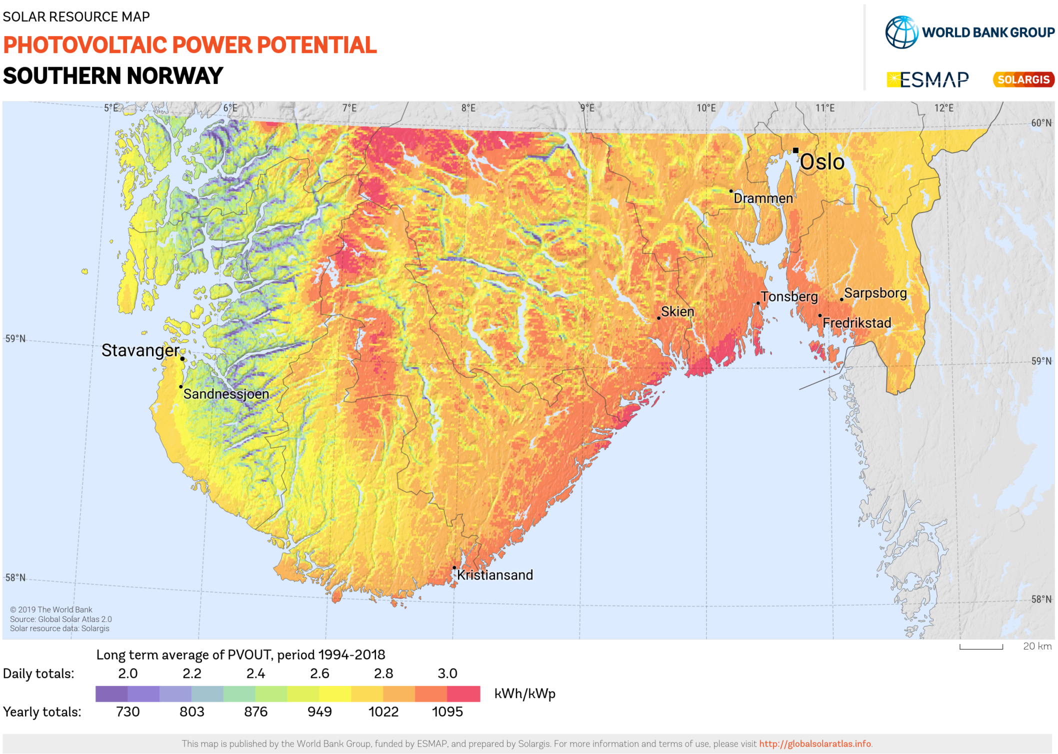


Figure 3.2: Partial photovoltaic electricity potential map in Norway: published by the World Bank Group, and visualized by Solargis. Source [37].

3.2 Data Filtering and Performance Analysis Methods

Existing outlier detection can roughly be categorized into three groups, rule-based, probability statistical theory, and artificial intelligence (AI). Rule-based filtering removes data that do not meet specific criteria; this could filter downtime, missing time intervals, power output, current output, voltage output, and faulty metadata [23], [40]. There have also been developed routines for grading the data, as has been used in [40]. The preliminary data quality grading was developed by IEA PVPS Task 13 [41], and grades power data depending on the number of outliers ($0 < P < P_{nom}$) and missing values. It defines missing values/data as 0 or NAN when irradiance is present. For the data to be accepted into the grading process, it has to contain at least 24 months of data, as this is a lower period for multiple analyzing tools [40].

Statistical probability theory is based on finding highly deviating values from a pattern and is quantified using different statistical tools. This is a low-cost way of outlier detection, where one could compare module or inverter output in cases with limited metadata, such as only current or power output. The authors of [42] used statistical methods, including the 3-Sigma rule, Hampel identifier, and Turkey's rule. The best indicator of how good these simple statistical methods are at detecting faults is their sensitivity to outliers. If too sensitive, outliers may not be detected as it is considered the norm. Therefore methods like the 3-Sigma rule are not recommended, as they break down at 10% contamination [42]. Tukey's rule and Hampel Identifiers are less sensitive to deviations and thus more suitable. The Hampel rule might give false negatives due to its insensitivity to outliers [42]. The near-linear power-to-irradiance relationship has also been utilized as a filtering technique. A study from 2019 [43] has developed a near-linear method for detecting un-normal operating conditions. Their method has been used with simulated and neighboring solar module data. The working conditions behind the model are that the compared reference data (simulated or neighboring PV installations) and the measured data are nearly linear. Therefore, a polynomial fit should allow temperature change with increasing power/irradiance. This method makes it possible to detect outliers and, thus, a loss in energy due to the surrounding area (clouds, snow, reflection) and a malfunctioning system. However, separating the different fault conditions still needs to be implemented and further studied. Data filtering appears minimal in multiple studies when calculating performance indices like PR and yield [4], [38], [40].

Also, a commonly used method is clear sky filtering. Clear sky filtering filters out data to only include data during low cloud cover. Clear sky filtering has the benefit of being consistent over time due to low fluctuations in irradiance. Therefore, this filtering is preferred when comparing modeled power to recorded data. For these reasons, the clear sky filter is a common filtering technique when calculating degradation and soiling loss [5], [44]–[47]. Common implementations are the PVlib library [48], and the RdTools library [49].

Typical daily profiles are also a form of statistical probability filtering. This method was developed to do a performance assessment where the loss in production due to failures and curtailment periods was not detected or recorded. As the name suggests, it calculates the production during a typical day, which can be used as a benchmark for comparing PV instalations. This technique also neglects the need for irradiance data, as only the power production log is needed. The model gives results on a typical day as the 50th percentile of the data and a clear sky day as the 90th percentile of the data [34].

Another way is to compare neighbors using peer to peers(P2P) analysis. This is an approach to analyzing the PV performance where either one [43] or multiple [50] peers are compared to the focus facility. This method can be more stable than the performance ratio without peer comparison, especially when less metadata is available or faulty [50]. In a perfect scenario, one would compare an installation on a neighboring roof with a similar tilt and azimuth, as these would have the same irradiance conditions. As this is impossible in most scenarios, some compromises must be made between distance and similarity. Therefore the basic steps of this method are 1) to quantify the best peers. 2) compare their energy production. 3) use fault detection on the compared data. One of the most significant benefits of this kind of model is its flexibility in adding metadata. Some studies have successfully analyzed data using only power generation data and location, although metadata such as peak capacity improves the result [50].

Support Vector Machine (SVM) is an AI model used to find anomalies in power data. SVM is a machine learning technique and, therefore, needs a training dataset clean of errors; Previous research has shown that a one-diode model for PV behavior is good enough to simulate this [51]. Machine learning methods like SVM is then used to find deviances between the prediction model and actual values. SVM is a useful method for detecting short circuit and shading faults [51], [52]. Autoencoders are another machine learning technique similar to that of SVM. Autoencoders have previously been used, like SVM, where a clean dataset is preferred to train the model. The training data could include data such as electrical parameters, solar irradiance, and temperature [53]. Isolation Forest is another machine learning technique, a benefit if this method is that it can handle more unbalanced datasets such as unbalance between anomaly and normal operation in the dataset [54].

3.3 Detecting Tilt and Azimuth

Big data studies that have examined the most common tilt and azimuth in Europe [55] found that a significant portion of the studied PV modules pointed south, with outliers in the range of $+/-100°$ from the south. The authors also found that tilt is most common in $0° - 50°$. The study used datasets in the latitude of $30° - 50°$. Datasets that rely on the user manually entering the orientation can include false standard values (e.g., 0°) when the user has not set any, or the user may set an incorrect value. An Australian dataset included 39% such values, which are likely wrong [55]. An newer big data analysis in Europe that used data gathered from private PV installations found that 30% of the installations had at least 10% of the time intervals missing [40]. It is, therefore, highly possible that automatically logged information might be missing or logged incorrectly.

Detecting the tilt and angle of PV installations is a problem that has been solved in different ways. One approach is to use digital elevation models (DEM). These models are created using radar, lidar, or stereoscopic images. However, these methods rely on the knowledge of the precise location and are often impractical because of time consumption. These methods can detect the tilt with accuracy down to 3° mean absolute error [13], [56], [57].

Other methods can simulate the facility in various orientations and find the best fit. However, these methods often require accurate metadata about modules and inverters' technology. The authors of [58] calculated the orientation as quality control of a dataset. Their method relies on a nonlinear least squares solver and performs well at $\approx 4°$ error. However, the method requires module technology to acquire the parameters for the model. PV installation parameters for the panel and inverter specifications can also be derived from historical PV power measurements and meteorological data as done in [59]. They have not calculated individual parameters, such as DC-loss, efficiency, and characteristics, but the cumulative effect of multiple. Their method showed to not be sensitive to outliers due to outages; however, shading is a significant issue. In optimal cases, the orientation can be computed with an error as low as 2° [59]. Curve matching between the power output and irradiance is also a method. The most significant benefit of this method is its ability to calculate the orientation without any metadata. However, these methods also struggle with shadow [13].

In [13], the authors describe the relationship between the E_{poa}, tilt, and angle. Figure 3.3 how E_{poa} is affected by tilt and azimuth. Figure 3.3a shows a module in the northern direction; this panel has a decreased E_{poa} at steeper tilt angles. However, a northern installation azimuth is uncommon in the northern hemisphere due to suboptimal solar patterns. Figure 3.3b, Figure 3.3c, and Figure 3.3d depict the impact of tilt and azimuth on solar panels facing south, east, and west, respectively. These curves are normalized against the maximum value for the corresponding day. In the southern direction, the time of peak normalized E_{poa} is not affected by tilt, as shown in Figure 3.3b. However, the normalized E_{poa} before and after peak hours decreases as the angle of tilt increases, resulting in a narrower curve for the irradiance. As an effect of the sun's path from east to west, the panels facing east reach peak normalized E_{poi} earlier in the day than panels facing west. An increase in tilt for east and west-facing panels results in a narrower curve for the normalized E_{poa}, indicating a shorter duration of peak irradiance compared to panels with a direct southern orientation [13].

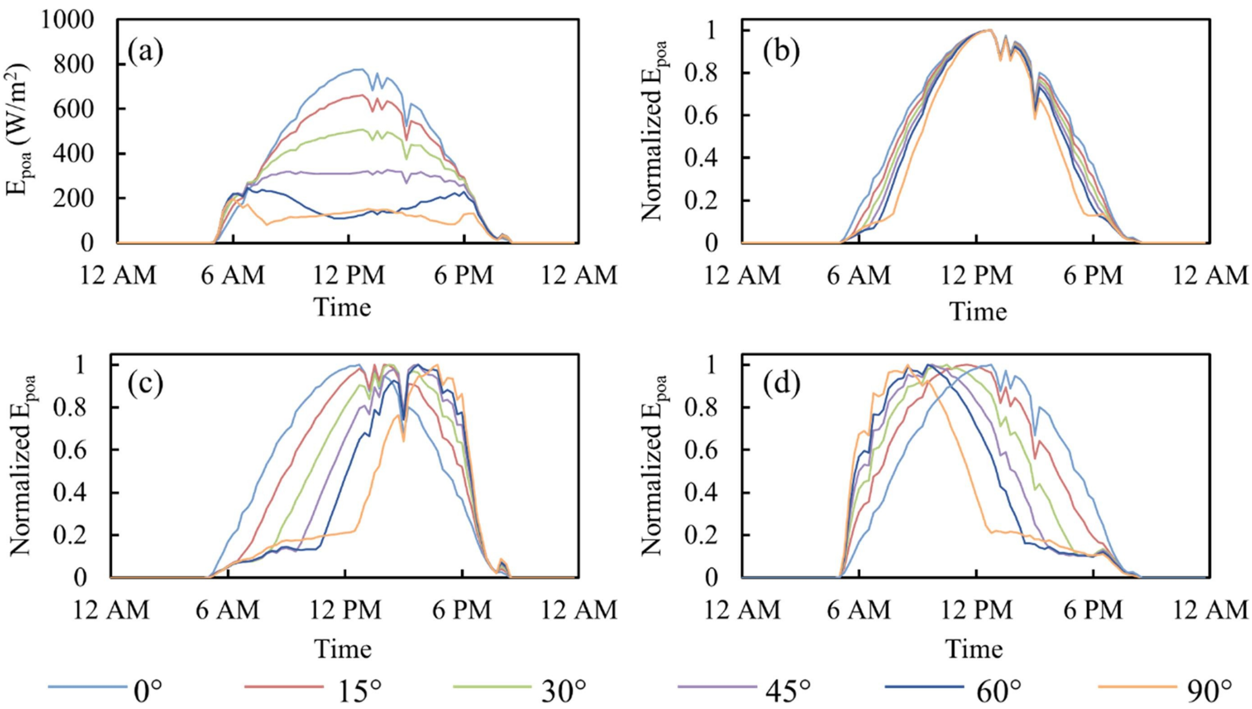


Figure 3.3: Effect of tilt and azimuth on E_{poa} for solar panels in different orientations: north (a), south (b), east (c), and west (d). Source: Meng et al., 2020 [13]

3.4 Off-site Irradiance Measurement

There is multiple off-site (implying that the recording did not occur at the particular location of interest) sources of irradiance data. CAMS solar radiation services [60], [61] is a part of the Copernicus program [62], a component of the European Union's space program. They can offer irradiation data in the longitude and latitude of $-66°$ and $66°$. National Renewable Energy Laboratory (NREL) also has a free service that provides high-resolution irradiance data for the entire globe. The data is in 4 km grid resolution between 1998 to 2017 at 30-minute intervals, with higher resolution of 2 km and 5-minute intervals after 2017 [63]. PVGIS, run by the European Commission, offers typical meteorological year data in hourly resolution. The data is available in the timeframe 2005 to 2020. As well as satellite-derived (SARAH, SARAH2, and NSRDB PSM3) and re-examine data (ERA5) [64], [65].

3.4.1 CAMS Accuracy

Larger inaccuracies at the edge of the satellite view have been registered. The cause of this is the satellite viewing angle, which causes cloud detection to be less accurate. For the Heliosat-4 model, this has been registered at latitudes above 60°. Snow can also cause problems as it can be mistaken as clouds. The Heliosat-4 model can offer surface solar irradiance that changes accurately over time. The 15-min interval measurements have a correlation coefficient of $0.67 - 0.87$ when compared against ground measuring stations for DNI and $0.68 - 0.87$ for DHI, and $0.90 - 0.96$ for GHI [66].

Quarterly reports are made to ensure the accuracy of the data. The report from March-May 2022 is publicly available [67]. The relative biases for all-sky global irradiance were low at under 5% for 24 of 32 stations, with an average bias of 16 $W/m^{-}2$. The biases were primarily positive, meaning the modeled values were higher than the measured. Mountain tops had the weakest performance. All-sky diffuse irradiance also mainly overestimated the model compared to the measurements and had a relative bias at less than 5% for 10 out of 17 stations, with an average of 15 W/m^2 (absolute value). All-sky direct normal irradiance had an average of 5.7% relative bias(27 W/m^2), where 10 of 15 measurement stations were under 5%. The results were overall concluded to be satisfactory, even in northern locations [67].

Chapter 4

Method

This chapter describes the methods used in this study and how they are implemented. The first section 4.1 describes how irradiance data has been gathered using the Heliosat-4 model from CAMS. The CAMS service is chosen because of its accessibility and having data for the necessary dates. Next, reverse geocoding is utilized to find the geographical names of the PV installation location; this is described in section 4.2. As all files have had some adjustment/control to their time format, the following section 4.4 describes the method used during time adjustment/control. As orientation(tilt and azimuth) was not included in the data from Solcellespesialisten, the method to estimate this is included in section 4.5 After this, the method for finding inliers is described in section 4.6. Finally, the PV system performance evaluation procedure is described in section 4.7. Figure 4.1 illustrates a high-level structure of the method.

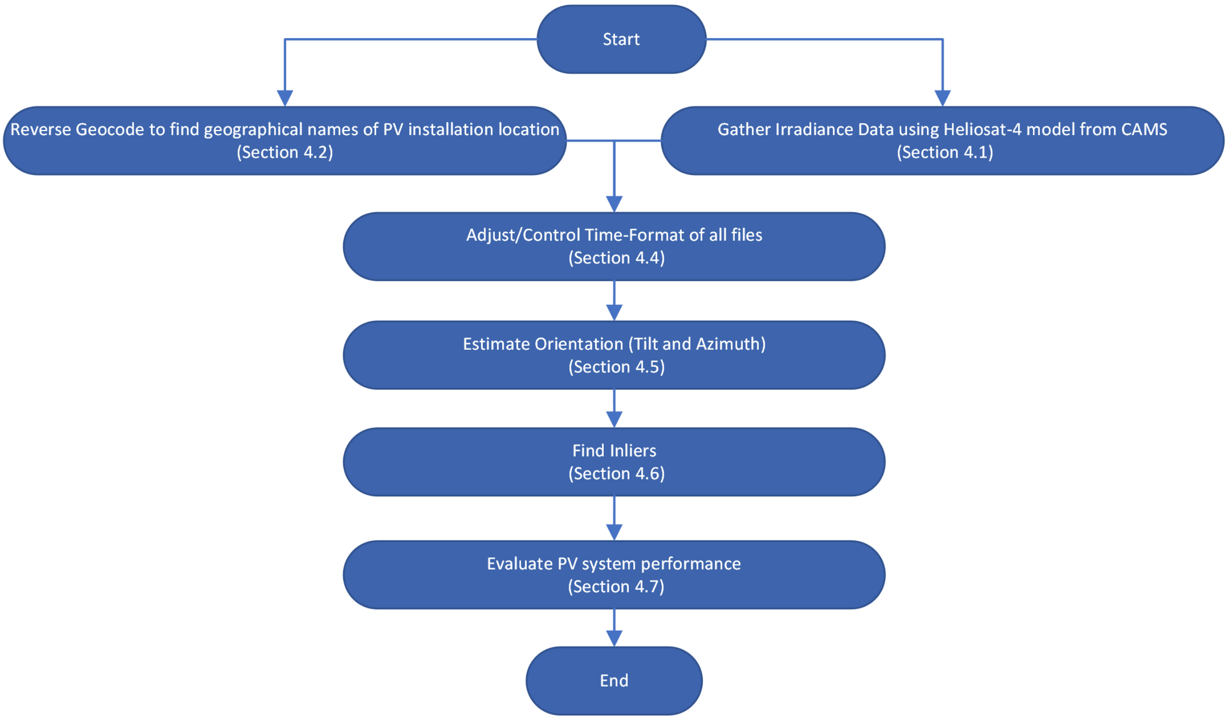


Figure 4.1: High-level flowchart of the utilized method

4.1 Irradiance Data - CAMS

Using the provided longitude and latitude, weather data is downloaded for all installation sites in their available period with a time resolution of 5-min. The service used to gather the weather data is CAMS. The data has been accessed using CAMS Radiation Automatic Access (SoDa) using pvlib's function `pvlib.iotools.get_cams` [10]. CAMS API is free; The only limitation is that a user profile has to be created, and a maximum of 100 requests can be sent each 24 hours. As the number of PV installations provided by Solcellespesialisten is 501, the corresponding irradiance data must be downloaded in smaller batches over multiple days. A Python script has been created to allow for this. A short description is shown in Table 4.1, and the complete code can be seen in Appendix K.

Table 4.1: Process of downloading irradiance data from CAMS

Step	Description
1	Manually create a download folder for the weather data
2	Loop over the different PV installations: a. Check if the plant ID from the PV installation folder matches the filenames in the weather data folder. b. If data has been downloaded previously, skip to the next file. c. If data has not been downloaded, download the weather data for that plant ID and save it in the weather data folder with the name of the ID.
3	Adjust time zone from UTC to CET. Then merge the weather data with the PV data and save the combined data in a new folder for further use.

4.2 Geolocation and Reverse Geocoding of PV Installations

The latitude and longitude of each PV installation have been reverse geocoded. The method of geocoding is the Python library `reverse_geocoder` [68], which includes cities with above 1000 in population size. The city, county, and municipality are located from this, giving a more descriptive placement.

4.3 Solar Position Algorithm

In the northern hemisphere, the sun rises in the east and sets in the west. The solar altitude during the day is linked to the latitude. NREL (The National Renewable Energy Laboratory) solar position algorithm is capable of calculating the zenith and azimuth between the years -2000 to 6000 with an accuracy of $+/-0.0003°$ [69], [70]. NREL solar position algorithm is utilized to calculate the solar zenith and azimuth for the available timestamps.

4.4 Time Zone

CET is used as the standard time format in this study. The PV production dataset and CAMS irradiance datasets are all adjusted to use this time format. The adjustment has been made using `pytz` Python library [71]. `pytz` cannot automatically detect the time format the data is given in. Manual detection of time format is therefore conducted. The beginning and end of the production data have been matched up to the sunrise and sunset given on the website `timeanddate` [72]. After that, the adjusted timezone is confirmed using computed solar zenith and azimuth angles with `pvlib.location.Location.get_solarposition` [73], which are compared to that of `timeanddate`.

4.5 Inference of Tilt and Azimuth

The tilt and azimuth of the studied PV panels are critical factors in the performance study. Since metadata and installation information is limited, a method should be found to determine the orientation using widely available data. Therefore, the method used is a data-driven inference approach that only uses logged power and irradiance measurements. The method uses curve-fitting on the most sky-clear days to determine orientation. The method was first developed and tested in 2020 and showed promising results at a maximum orientation inference error of 10% or less [13]. There is also an existing code implementation [74] that has been utilized and modified to this thesis use.

This method has been verified using a PV installation where the tilt and azimuth are known. The used PV installation is located in Grimstad, Agder, on the roof of UIA. The installation azimuth is in two directions; $\approx 83.2°$ and $\approx 263.2°$, and the tilt is $\approx 10°$.

4.5.1 Step 1. Data Loading and Preprocessing

The selected PV system and weather data are loaded and preprocessed. This includes adjusting the timezone of the data and labeling daytime saving as described in section 4.4. Finally, all data is resampled to hourly left-closed format before being merged.

4.5.2 Step 2. Solar Position and Irradiance Calculation

The latitude and longitude of the selected PV installation are considered by using `pvlib.solarposition.get_solarposition` [73] to calculate the solar position for each timestamp. Days, where the solar zenith does not have lower values than 70° have not been included. This is due to the low zenith angle increasing the chance for shadow [13]. This causes some winter months not to be included in the analysis.

Daily Diffuse Fraction

The weather data is used to determine which day has the lowest chance of clouds to occur. This is done for multiple reasons, the foremost being that small clouds are less likely to occur, and off-site irradiance measurements might need a higher resolution to capture these. This also allows for using the same weather data over greater distances. For each day in the selected year, the DHI and GHI are individually summed up over the day and then used in equation 2.10. After that, every month's clearest day (lowest answer for K_d) is filtered out for further use [13].

Transposing GHI, DHI and DNI

The GHI data is transposed to E_{poa} for each hour in the clearest days. Since this aims to find the tilt and azimuth of the PV module, every possible angle is calculated ($0 - 360°$ for orientation, $0 - 90°$ for tilt, with $0°$ included for tilt). This is done in $1°$ resolution; Each hour, the GHI value is transposed 32,760 times.

The method is implemented using pvlib and its various functions. The method used for transposing is the Perez model [15] and is calculated using the `pvlib.irradiance.get_total_irradiance` [14] function. In addition to the GHI, DHI, DNI, and orientation, the pvlib function takes multiple other variables, which are listed below, together with the procedure used to gather them.

1. **Solar Zenith and Azimuth:** These values are obtained using the respective period and the pvlib function `pvlib.solarposition.get_solarposition` [73].
2. **Extraterrestrial Direct Normal Irradiance:** This value is determined using the pvlib function `pvlib.irradiance.get_extra_radiation` [75], along with its default values and the relevant year.
3. **Airmass:** The airmass is calculated using the Solar Zenith and Azimuth obtained from the first point in this list, combined with the pvlib function `pvlib.atmosphere.get_relative_airmass` [76] and its default values.
4. **Albedo and Surface Type:** The albedo is set to 0.2, representing a typical value for various surface types [3, p. 37].

4.5.3 Step 3. Searching for Optimal Tilt and Azimuth Angles

Normalization and Curve Evaluation

The 12 days selected (one for each month) in chapter Daily Diffuse Fraction is normalized in this step. As the amplitude of the power and irradiance data differs, both are normalized with respect to their maximum values, with equation 2.16 and 2.17, for each day.
The normalized power and plane of array irradiance are then compared, using RMSE as the cost function. Each day has one normalized power curve and 32,760 normalized irradiance curves. A lower cumulative RMSE value indicates a better fit between the two curves. The top 15% E_{poa} curves with the lowest result are selected as the result of tilt and azimuth for that particular day and, therefore, the month.

Generating and Overlapping Monthly Results

The monthly result consists of 4914 values $(32,760 \cdot 15\%)$ of tilt and azimuth that deviate from one another. The other months might also get different results due to seasonal effects like temperature and wind speed. Therefore, each month is compared to one another, and the number of duplicate tilt and azimuth values is calculated. The tilt and azimuth are treated as two separate values. There can therefore be a maximum of 12 similar (for a 12-month dataset) tilts and azimuth angles. Linear interpolating is then performed to calculate the result, as shown in [13].

4.5.4 Code Implementation and Modification

The code used to perform the computations in this section is based upon the code of "Data-driven inference of unknown tilt and azimuth of distributed PV systems" by Meng et al. [13]. The source code can be found at [77]. The modifications done to the code are importing PV data and weather data and data manipulation, such as setting the exact time format and daylight saving. On an AMD Ryzen 5800H, the code took somewhere between 45 min to 1 hour to execute. This is mainly because of the number of times the E_{poa} calculation must be executed. The test data from UIA consists of 120 individually monitored panels (here treated as separate systems), and the data from Solcellespesialisten has 373 PV installations that passed filtering processes. Some optimization in runtime was necessary. Altho computing time of respectively 120 hours and 373 hours would be feasible, this would limit the usability of the code, especially regarding troubleshooting the result. The code has therefore been modified with the `ProcessPoolExecutor` from the `concurrent.futures` library [78]. This allows Python to run the same amount of processes as CPU threads available, 16 in this case. The code uses approximately 1-hour to run after the modification, and a batch of 16 results is thus calculated each hour, significantly increasing the speed.

4.6 Filtering by Clustering

The filtering process of the data points is mainly inspired by [43]. This procedure was chosen because of its promise of adjusting for the nonlinearity of the performance due to temperature change with limited metadata.

4.6.1 Step 1. Data Loading and Preprocessing

The data needed for the filtering process is the data that is being filtered (Y_f) and some reference data (Y_r). The only requirement for the reference and the data being filtered is that there is a strong relationship between them. Because of this, irradiance-irradiance, irradiance-power, and power-power are the combinations that can be used. After selecting the chosen data, a timezone adjustment is performed, as explained in section 4.4. This is essential to make sure that the correlation between the two data is maintained [43].

4.6.2 Step 2. Normalization and Error Calculation

Both the Y_f and Y_r datasets are normalized. Equation 2.16 is used to normalize when power data, and equation 2.17 is used for irradiance data. Datapoints recorded during the same time instance are then compared, and the deviation is calculated using equation 4.1 [43].

$$error(Y_f) = Y_f - Y_r \tag{4.1}$$

4.6.3 Step 3. Finding Inliers using RANSAC

A regression line is found using RANSAC from `sklearn.linear_model.RANSACRegressor` [79]. As the RANSAC model fits a regression line from a random sample of inliers, the result may alter when rerunning the calculation. A grid search is used with `sklearn.model_selection.GridSearchCV` [80] to find the most optimal parameters, table 4.2 shows the combination of parameters tested to get the best result.

Table 4.2: GridSearchCV Parameters

Parameter	**Values**
min_samples	Range from 10 to 149
max_trials	100, 200, 300, 500, 700, 1000, 1500
residual_threshold	Range from 0.07 to 0.15 with step size of 0.01
loss	absolute error

4.6.4 Step 4. Binning and Polynomial Regression

Only the data categorized as inliers from the RANSAC result in the last step is used for further calculations. The range of Y_r is divided into equally sized groups. Within each group, a histogram is created of the previously calculated error value from section 4.6.2. A polynomial fit in the 1st to the 10th-degree range is created within each group with `numpy.polyfit` [81], and the polynomial fit with the lowest mean squared error is selected. The global maximum and local minima are found using `scipy.signal.argrelextrema` [82]. In cases where the local minimum is under 0, it has been set to 0. Figure 4.2 illustrates a sample of a bin.

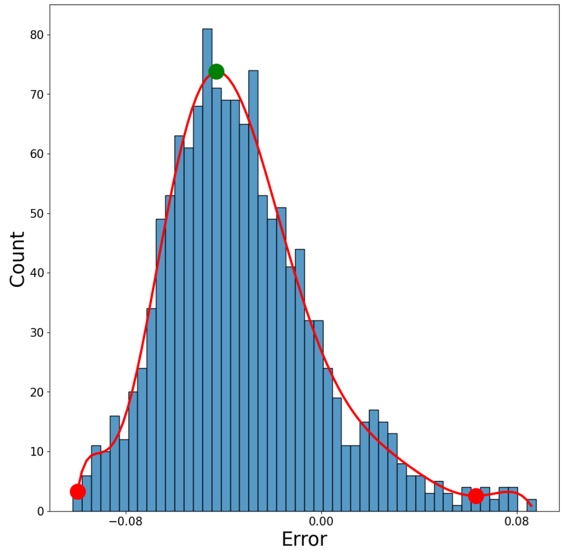


Figure 4.2: Example of a bin (eg., $Y_r = 0.4 - 0.5$) during the filtering process from one of Solcellespesialisten's PV installations. Green markers: global maximum. Red markers: local minima. Red line: Best fit polynomial curve.

The error value where maximum/minima occur is set as the error value for that group. Figure 4.3 shows the global maxima, left minima, and right minima for all groups combined into one plot. A 4th-degree polynomial curve is fitted to each maximum and minima dashed line. For every point along the x-axis, the x-axis value of the polynomial line is added to the y-axis, as shown in equation 4.2 [43] resulting in moving the curve into a 45-degree angle (see Figure 6.4) in the first quadrant. The resulting position of the right polynomial line becomes the upper limit, and the left becomes the lower limit of inliers.

$$Y_f = error(Y_r) + Y_r \tag{4.2}$$

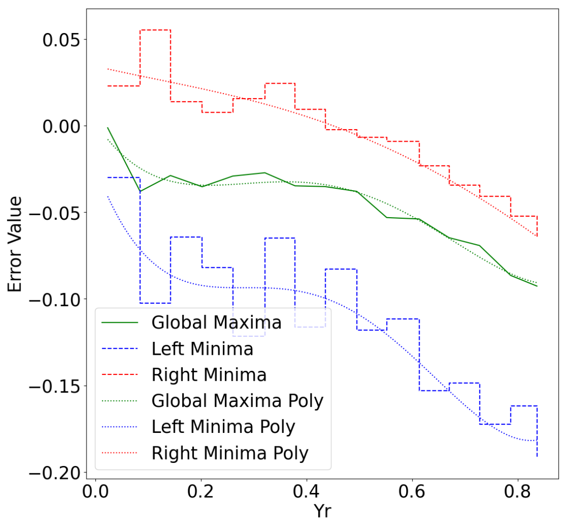


Figure 4.3: Global maximum, left and right minima result from all bins. Example from the filtering process of one of Solcellespesialisten's PV installations.

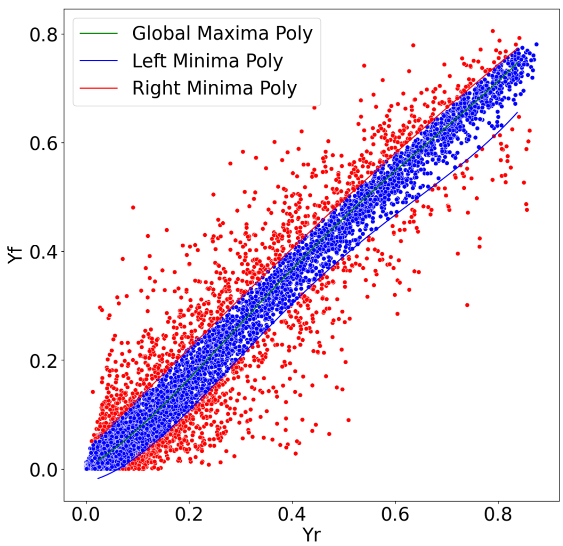


Figure 4.4: Filtered data result. Example from the filtering process of one of Solcellespesialisten's PV installations.

4.7 PV System Performance Evaluation

Three datasets are made for the PV system performance evaluation; 1) All data; contains all data in its original form. 2) inliers from the RANSAC regression, and 3) inliers from the polynomial fit; have 0 values removed before the RANSAC regression is fitted. This is due to two reasons; firstly, in some cases, the 0 values influenced the RANSAC regression line. And secondly, 0 values have been considered downtime and do not represent the PV installation at operational times. All three datasets only contain PV installations where the location was found in Norway. The procedure and result regarding this filtering can be seen in section 5. All datasets are limited to one year (365 days).

4.7.1 PR

Based on earlier research [23], [83] the PR is expected not to be normally distributed because a semi-natural limit is caused by values above 0.95 being extremely hard to achieve [83]. Therefore a Weibull distribution is expected to match better. With the underlying data being nonparametric, a Kruskal-Wallis H-test is utilized to test if there are any differences in the mean of the locations. Finally, the post-hoc analysis is performed with Dunn's, and Mann-Whitney U tests in cases with a statistical difference. As the uncertainty of a type 1 error increases as more groups are tested, a correction is made with the Benjamini-Hochberg procedure. The PR analysis is done with all three datasets, and Tukey's rule is applied to identify and remove any outliers in the remaining dataset.

4.7.2 Spesific Yield

Due to the removal of data points lowering the total kWh available, only dataset 1) has been utilized to calculate the specific yield. The data from Solcellespesialisten has been cleaned using statistical removal of outliers. The method used is Tukey's rule and is applied to identify and remove any outliers in the remaining dataset, improving the overall reliability and validity of the data analysis. Turkeys rule is represented in equation 2.22. The PR value is defined by identifying the peak value of a fitted Weibull distribution. Specific yield is expected to be somewhat normally distributed across the different PV-innstaltions [83]; therefore, a one-way ANOVA test is utilized together with a posthoc Tukey HSD.

Chapter 5

Data and Data Manipulation

This chapter describes the data given by Solcellespesialisten and the test data from UIA. The description of the data from Solcellespesialisten is provided in section 5.1. Section 5.1.1 includes information about the metadata and production data, and section 5.1.2 includes what data manipulation is done to the raw data. The description of the data given by UIA is in section 5.2.

5.1 Solcellespesialisten

5.1.1 Raw Data

The data has been provided by IFE, who obtained it from Solcellespesialisten. It consists of 501 distributed PV facilities, mainly small in the southern half of Norway. For each installation the data is separated into 2 files, metadata, and PV data.

Metadata

Metadata for each installation has been given in an Excel file. An example of a metadata file and its included information is shown in table 5.1. The location is given as longitude and latitude, with two decimal places. However, no information is given about the unit of measurement of the "Capacity" column. The capacity is most likely given in W_p, kW_p, or MWp. As a starting point, the capacity unit is set to kW_p and adjusted afterward. This is further discussed in section 5.1.2. The plant's creation date is also given; however, this seems to match the start date of the data more than the actual creation date. Finally, the number of errors is given as "error" and "no error," where no error has been detected for any data.

Table 5.1: Metadata information

Longitude	**Latitude**	**Capacity**	**Plant Created**	**Error Count**	**No Error Count**
10.56	59.92	5,000,000	2022-01-01T00:00:00	0	366

Solcellespesialisten was, unfortunately, unable to provide metadata about the installation tilt and azimuth. The tilt and azimuth, therefore, have been gathered by other means. IFE has access to a database of all rooftops in Norway, with information such as tilt and azimuth. However, as the accuracy of the data coordinates is in the range of two to five decimals, the correct roof might not be selected. For example, during IFE‘s testing of this procedure, some rooftops were selected, and offsets of approximately 50 meters were discovered by visual inspection using Finn Kart aerial photo between the actual installation and the selected roof. An alternative to this method would be using paid services; however, this does not remove the problem of varying degrees of coordinate accuracy. Another challenge is that roofs often consist of multiple parts with different angles. IFE has used the database solution on 380 of the PV installations to find a plausible angle and tilt and has been kind to share their result. However, due to the previously mentioned challenges, this file includes multiple orientations for several PV installations. An example of the tilt and azimuth for an installation is depicted in table 5.2. This shows a small part of the result of PV installation number 1005. An important notice is that table 5.2 does not convey the spread of the tilt and azimuth, as more variation in tilt and azimuth are included, making it difficult to estimate the actual tilt and azimuth.

Table 5.2: IFE rooftop metadata

Plant ID	**Tilt**	**Azimuth**
1005	79.75	341.35
1005	11.83	281.22
1005	84.99	80.78
1005	84.27	30.75

A solution here is to visually inspect every rooftop with satellite images; this has not been done due to time challenges. As an alternative to this, the orientation has been estimated using the power data, as described in section 4.5.

PV Production Data

The log of the recorded data for each installation is given in a JSON file. The JSON file consists of a list for each timestamp entry, which has been combined into a single list. An example of the list is shown in table 5.3. The information is logged in 5 min intervals For the performance evaluation in this thesis, only AC production was used.

Table 5.3: Information from JSON file containing PV data

Variable	Time Interval 1	Time Interval 2	Time Interval 3
Key	197	197	197
Timestamp	2022-03-02T09:15:00	2022-03-02T09:20:00	2022-03-02T09:25:00
Date	2022-03-02	2022-03-02	2022-03-02
Time	09:15:00	09:20:00	09:25:00
Delta	Instant	Instant	Instant
AC Production	4748	3686	4633
Daily Production	1.040	1.420	1.780
Total Production	18697.0	18697.400	18697.801
Month Total Production	0	0	0
Year Total Production	247.217	248.659	249.639
Vnom	142.6	143.3	143.7
Voltage L1	143.5	144.4	144.8
Voltage L2	142.6	143.5	144.4
Voltage L3	11.0	8.5	10.6
Current L1	11.1	8.6	10.7
Current L2	11.0	8.5	10.6
Current L3	50.07	50.04	50.02
Frequency	11592	11592	11592
Run Hours	28.4	31.8	34.8
Temperature	28.4	31.8	34.8
Mocked	False	False	False
MPPT	None	None	None

5.1.2 Refining Dataset

Data Aggregation and Conversion to Hourly Format

The metadata and PV data have .been combined using the filename, as both files have a unique number in the name. After these have been merged, the data consists of approximately 20 GB. They have, after that, been merged into an hourly format to make the data more manageable, both computational and visually. When merging the timestamps, it can be binned using the sum, mean, first, or last in the corresponding hour. Table 5.4 shows how the different variables have been merged.

The timestamp has been marked as "first," so the time format becomes left-closed, meaning that hour 12:00 contains values between 12:00 and 12:59. The line voltage and current were utilized to calculate the AC production to be logged in W. This is transposed to Wh by taking the mean of all values in an hour. Every variable marked as "First" (except Timedate) is utilized to keep a constant value. The aggregation method "Last" is used in cases where the original data is summed up for each column. Table 5.4 shows the aggregation methods used for the various variables.

Table 5.4: Aggregation method for Solcellespeialisten's dataset

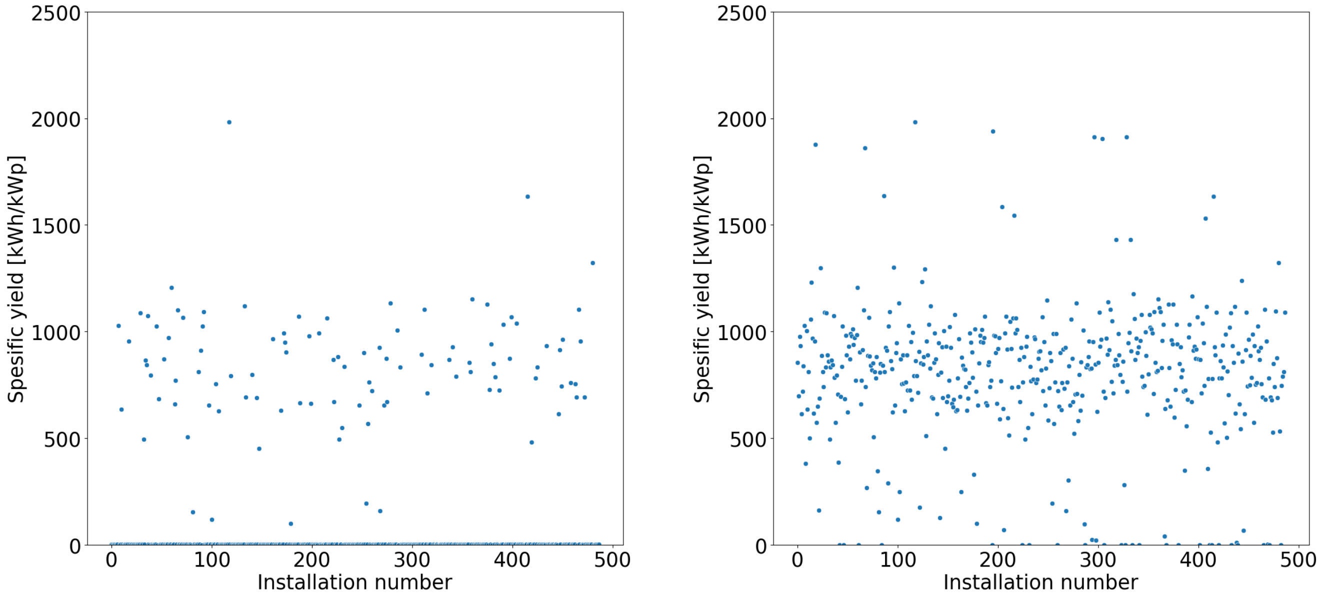
Variable	Aggregation
Key	First
Timedate	First
Capacity	First
AC Production	Mean
Daily Production	Last
Total Production	Last
Vnom	Mean
Voltage L1	Mean
Voltage L2	Mean
Voltage L3	Mean
Current L1	Mean
Current L2	Mean
Current L3	Mean
Frequency	Mean
Run Hours	Last
Temperature	Mean
Mocked	First
MPPT	First
Latitude	First
Longitude	First

Refining Capacity Data

Due to a possibility of wrongly logged capacity in the metadata file, this has been checked, the method and result are explained in this section.

Due to the data containing slightly more than a year, the data has been filtered to contain exactly one year (365 days). The yearly specific yield is thus calculated on the period 01-03-2022 23:00:00 to 01-03-2023 23:00:00. Furthermore, three types of data have been removed: 1) Locations outside Norway, 2) Instances where no location was found, and 3) PV installations that generated 0 kWh/kW$_p$ per kW$_p$ per year. The capacity in the original data has been presumed to be in kW$_p$. The result is visualized in Figure 5.1a, and statistical breakdown is shown in Table 5.5.

Figure 5.1a shows the specific yield of all installations when the capacity is estimated to be in kW$_p$. From visual inspection of Figure 5.1a, a lot of the data is within the expected range of approximately 500-1300 kWh/kW$_p$ per year, as found in the literature review. However, many data points are near the 0 kWh/kW$_p$ marker. Therefore, some data may have been logged in different units: W$_p$ and kW$_p$. Table 5.5 supports this hypothesis, as the 50th percentile of specific yield is 0.91 kWh/kWp. This indicates that half the data is below 0.91 kWh/kW$_p$. From visual inspection of the datafile, many of the specific yields were in the range of 0.5 to 1 kWh/kW$_p$, corresponding with the 50th and 75th percentile of the table. Due to this reason, the most likely cause of the skewed results has been determined as logged capacity in different units. The capacity has therefore been divided by 1000 where the yearly specific yield is below 5 kWh/kW$_p$. This value was selected to be higher than the expected specific yield based on the fact that later stages will filter out unexpected values regardless. After this step, the result still showed some anomalies, as some capacities are still logged with a capacity under 5 kW$_p$ resulting in a yearly specific yield of 300,000 kWh/kW$_p$ and above. In cases like this, the capacity has been multiplied by a factor of 1000. The finalized adjusted yearly specific yield, after these steps, the result is visualized in figure 5.1b and summarized in table 5.6 together with the capacity.



(a) Raw data

(b) Refined data

Figure 5.1: Yearly specific yield vs. installation number scatterplots. Figure (a) displays the raw data scatterplot. Figure (b) displays the refined capacity scatterplot. Note; y-axis has been limited for improved visibility.

Table 5.5: Summary statistics of yearly specific yield and capacity. Assuming metadata capacity in kW_p

Statistic	Yearly Specific Yield [kWh/kWp]	Capacity [kWp]
mean	3,047	6,714
std	48	7,690
min	0.0	0.006
25%	0.7	3,000
50%	0.9	5,000
75%	1.5	9,000
max	988,324	96,000

Table 5.6: Summary statistics of yearly specific yield and refined capacity

Statistic	Yearly Specific Yield [kWh/kWp]	Capacity [kWp]
mean	797	194
std	287	1,137
min	1.43	2
25%	690	5
50%	840	7
75%	952	10
max	3,086	11,000

In total, the capacity for 344 installations was divided by 1000, and 2 were multiplied by 1000 to bring all to the same unit of kW_p, as seen in table 5.6. The mean of the values is now within the range of the 25th and 75th percentile of data, indicating that most values are located where expected. By visual inspection, some outliers with a factor of 100 off expected values remain in the data. These have been left unaltered due to not altering the data for the worse. These values and other low and high anomalies will be filtered out in later stages.

Geographical Distribution

Figure 5.2 shows all the installations in Norway provided by Solcellespesialisten. Where multiple installations are present, they are shown as groups, with a heat-map overlay to show more accurate placement. All installations are below 66° latitude, the limit of CAMS weather data, and weather data for all locations have been collected. On visual inspection of the interactive map, some locations' coordinates do not correspond with the actual placement. One point, in particular, is placed in the sea, as can be seen in the middle-left part of figure 5.2. Points such as these have been removed from the dataset. Some are also placed just outside the coast; these placements may not be wrong, but a consequence of the two decimal coordinates. Table 5.7 shows the number of PV facilities in each county. All counties with less than 10 PV installations have not been prioritized for further study.

Table 5.7: County location of PV installations from Solcellespesialisten‘s dataset

County	Number of PV installations
Rogaland	105
Ostfold	97
Akershus	62
Hordaland	54
Hedmark	45
Buskerud	32
Vestfold	25
Sor-Trondelag	18
Telemark	14
Oslo	11
Oppland	11
Vest-Agder	5
Sogn og Fjordane	3
Aust-Agder	2
Nord-Trondelag	2
More og Romsdal	1

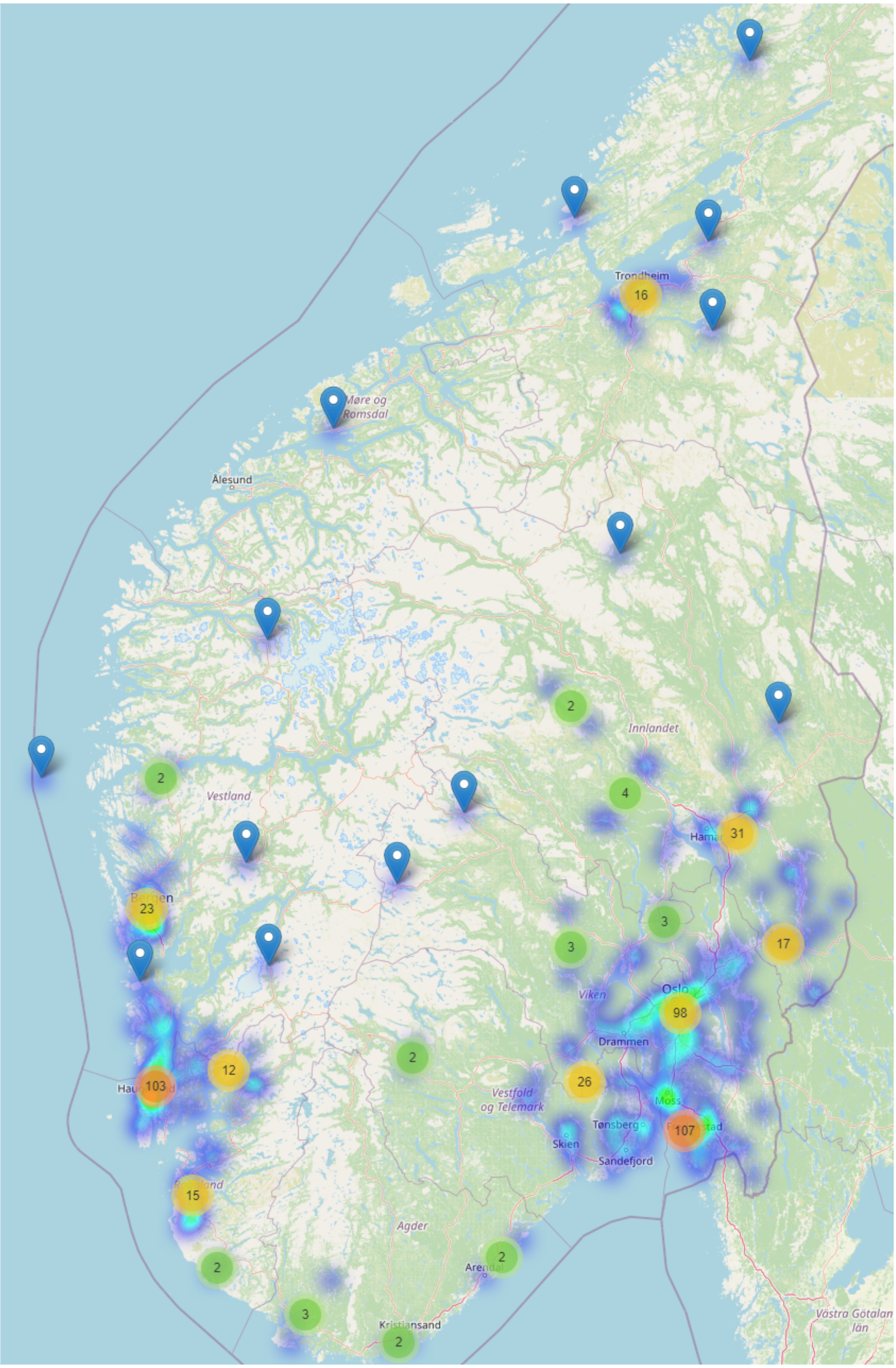


Figure 5.2: Raw data: Map of installations in Solcellespesialisten‘s dataset

Missing Data

Missing data points can change the result of time-sensitive calculations such as specific yield. The data has, therefore, been analyzed for missing timestamps to analyze the quality of the time series. Furthermore, the different installations have been analyzed separately. In addition, each month in the PV-facility datalog has been studied. The result can be seen in table 5.8, where the outcome is grouped into six bins. The vast majority of the data has no missing timestamps. However, 388 of the months contain less than 10% of the available month. All these sub 10% availability months are the month of 03.2023. This is the last month of the dataset and only contains the first day. There is also a low amount of data that is missing less than 5%. The installations with more than 90% of timestamps available have not been deleted as this has been set as the threshold for deletion has been set to 90%. Overall the result shows a low amount of missing timestamps.

Table 5.8: Available timestamps

Category	**Months**
99-100%	4652
95-99%	6
90-95%	1
50-90%	0
10-50%	1
0-10%	388

Due to the data being transformed from 5-min intervals to hourly intervals, there might be some missing intervals in each hour. The original data has therefore been checked for missing 5-min gaps. In the original data, the sum of all missing 5-min intervals for each month equaled full days, indicating that only full days are missing or some data manipulation has already occurred to fill in the missing time. Therefore, the missing times-intervals in the original 5-min data match the result of Table 5.8, and each hour is highly likely not to lack any data.

5.2 UIA

5.2.1 UIA Data

The PV installation from UIA is located on a flat roof. It consists of 120 PV modules with module-level monitoring (Tigo optimizers). The PV modules are a mix of IBC PolySol, IBC MonoSol, and SunPower mono-si panels. They are installed in two directions; east ($\approx 83.2°$) and west ($\approx 263.2°$), with a tilt of 10°. Refer to figure 6.10 and 6.11 for layout.

The data is recorded in 5 min intervals from January 2019 to January 2021. Visual inspection of the data revealed a lot of small negative power values during the night; These have been removed and replaced with the value 0. After removing the negative power values, the timestamp has been resampled to hourly format, using "sum" as the aggregation method.

Weather information has been collected on the roof of UIA using various instruments, including pyranometers, temperature sensors, and wind measurement devices. The data has been recorded in both 1-minute and 1-hour intervals. However, only the 1-hour interval data has been utilized for this analysis. Visual inspection reveals a lot of negative values in this dataset GHI, DHI, and DNI values during the night-time. These negative values have been adjusted to 0.

Table 5.9: Weather data columns from UIA with descriptions

Columns	**Description**
Timestamp	Time and date of the data point
Record	Record number
POA1 - POA5 Averages	Plane of array irradiance averages
GHI Average	Global horizontal irradiance average
DHI Average	Diffuse horizontal irradiance average
Albedo1, Albedo2 Averages	Albedo averages
DNI Average	Direct normal irradiance average
PVT1 - PVT20 Averages	PV module temperature
Wind Speed (Max, Min, Avg)	Wind speed (max, min, average)
Wind Direction	Wind direction from
Precipitation Total	Total precipitation
Atmospheric Pressure Avg	Average atmospheric pressure
Air Temperature1, Air Temperature2 Averages	Air temperature averages
Relative Humidity Avg	Average relative humidity

Chapter 6

Results

This chapter brings forth the results. First, section 6.1 describes the differences (UIA - CAMS) in local and CAMS irradiance for the location of UIA, Grimstad. After that, section 6.2 shows the result of the inference of tilt and azimuth; this chapter is separated into four subsections: where the tilt and azimuth are inferred with the local irradiance data from UIA, irradiance data from CAMS, the effect of shading on the result when CAMS data was utilized and lastly the result of the clearest day calculation. Furthermore, section 6.4 shows the capacity distribution of the utilized PV installations. The performance ratio of the systems is after that presented in section 6.5. Continuing with the data analysis, the found specific yield is in section 6.6. Finally, the RANSAC and polynomial filtering result is shown in section 6.7.

6.1 Comparative Analysis of Local and CAMS Irradiance Measurements

Figure 6.1 shows three histograms of difference for the Global Horizontal Irradiance (GHI), Diffuse Horizontal Irradiance (DHI), and Direct Normal Irradiance (DNI) between the measurements done locally at UIA, and the Inferred results from the CAMS service. Nighttime has been removed by removing data where both local and CAMS data are zero. This is done in order to emphasize the difference in daytime irradiance data. The data consists of 11 048 hours of data, where one hour is one measurement. The data is from the date 2019-08-04 to 2021-12-31. The leftmost plot shows the difference histograms for the Global Horizontal Irradiance (GHI). The absolute difference from table 6.1 shows that Q1 differs less than 4.51 W/m^2, median less than 15.14 W/m^2. And Q3 under 57.14 W/m^2 difference. This indicates that the data is well aligned, and most measurements correlate with an acceptable error range.

The middle figure shows the same comparison for DHI. Table 6.1 show that Q3 and max values are slightly more inaccurate, leading to a broader graph. This is not the case for the Q1, with slightly better accuracy.

The DNI difference depicted in the rightmost graph shows the least accuracy with Q1 under 9.829 W/m^2 and median under 44.4 W/m^2, and Q3 under 123.1 W/m^2 difference. This is the worst-performing measurement. Resons for the DNI difference to be

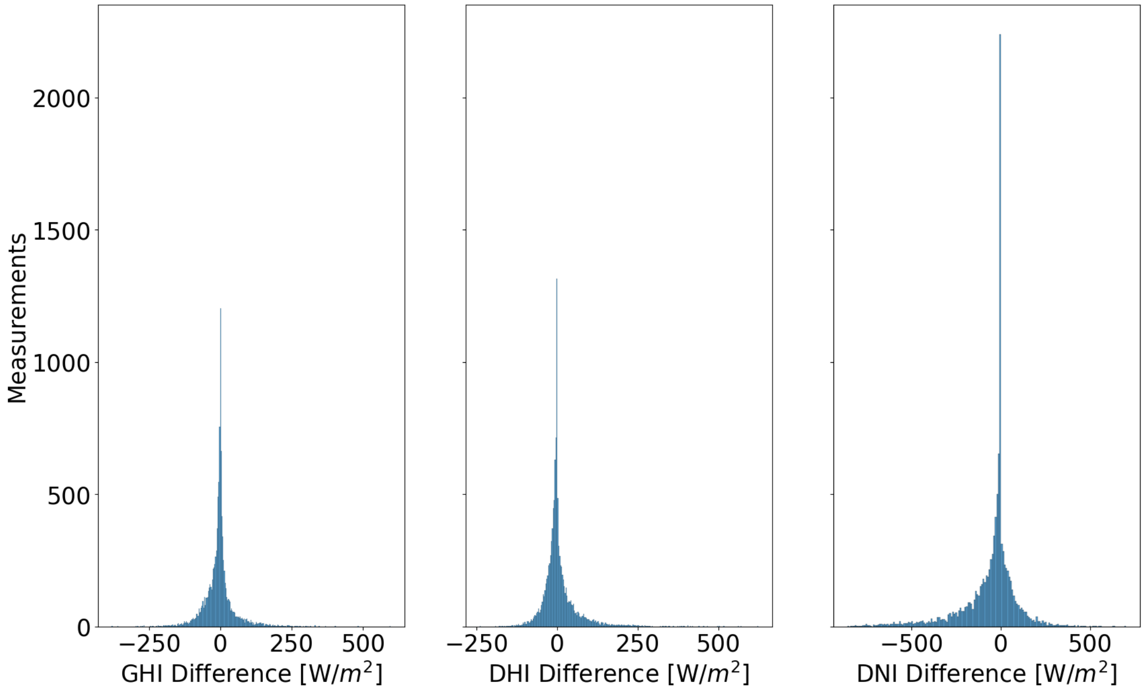


Figure 6.1: Comparison of GHI, DHI, and DNI absolute in histograms: This image consists of three separate histograms, with the count on the y-axis and the differences of GHI, DHI, and DNI on the x-axis (in W/m²). The left plot shows the GHI differences, the middle plot displays the DHI differences, and the right plot illustrates the DNI differences.

Table 6.1: Summary statistics of the absolute error in irradiance data

	GHI Difference [W/m^2]	DHI Difference [W/m^2]	DNI Difference [W/m^2]
Mean	31.1	30.5	93.9
STD	41.5	43.9	130.0
Min	0.0	0.0	0.0
Q1	4.5	5.1	9.8
Median	15.3	16.1	44.3
Q3	42.1	38.2	123.1
Max	596.2	623.7	858.4

Table 6.2 shows the monthly deviance between local and CAMS data irradiance measurements. Due to an increase in day length during the day, these months contain more data. January, February, November, and December appears to have less deviation in the mean, Q1, median, and Q3 values for the GHI and DHI measurement. However, the opposite seems true for the DNI measurements and higher error in general. A large part of the increase in DNI's inaccuracy might be due to measurement challenges. Cloud-causing shadow is likely one of these, as this is a commonly recorded problem. A larger error during the winter month could be described by snow being detected as shadow [66]. This is also visible in scatterplot 6.2 where many local DNI measurements measure close to 0 W/m^2, while CAMS data measured high values.

Table 6.2: Summary statistics for the difference between local and CAMS data for GHI, DHI, and DNI by month.

	GHI Difference [W/m^2]							
	Count	Mean	STD	Min	Q1	Median	Q3	Max
January	486	20.72	23.91	0.01	3.57	12.64	29.18	145.32
February	576	25.88	28.61	0.00	4.66	15.81	39.15	180.77
March	779	29.78	36.12	0.01	4.21	15.72	43.02	278.96
April	911	25.41	42.21	0.00	2.94	9.60	28.34	596.26
May	1090	36.29	46.20	0.01	4.06	14.74	51.81	298.87
June	1140	39.23	53.23	0.00	4.65	17.12	52.21	332.43
July	999	43.34	54.97	0.00	6.33	22.71	62.13	404.34
August	1440	35.99	50.38	0.00	5.02	16.13	47.18	483.02
September	1223	29.88	34.70	0.00	5.35	15.85	43.69	270.52
October	1022	27.59	29.22	0.01	6.16	16.87	41.23	272.28
November	724	22.18	22.48	0.00	4.35	14.71	35.52	160.87
December	658	18.85	17.30	0.01	4.37	13.70	29.20	100.23
	DHI Difference [W/m^2]							
	Count	Mean	STD	Min	Q1	Median	Q3	Max
January	486	20.07	24.66	0.01	2.80	9.87	28.42	147.92
February	576	19.15	21.38	0.00	3.27	11.01	27.77	137.11
March	779	23.23	24.68	0.00	5.34	14.94	32.34	149.13
April	911	22.09	26.35	0.01	4.34	14.07	28.89	191.08
May	1090	32.28	36.40	0.00	7.21	19.34	43.34	222.64
June	1140	35.06	44.31	0.04	6.55	18.73	46.40	328.75
July	999	34.63	42.33	0.00	6.26	17.81	46.83	239.89
August	1440	40.26	60.71	0.01	6.34	19.24	49.08	623.71
September	1223	43.12	65.07	0.00	6.97	20.36	50.27	467.23
October	1022	32.32	47.17	0.00	5.92	16.23	40.56	386.99
November	724	19.71	25.29	0.00	3.38	10.60	27.00	197.58
December	658	16.53	18.06	0.00	2.93	10.34	23.89	108.15
	DNI Difference [W/m^2]							
	Count	Mean	STD	Min	Q1	Median	Q3	Max
January	486	154.84	174.67	0.00	16.69	94.16	229.67	769.98
February	576	78.78	112.37	0.00	4.40	33.76	105.17	707.73
March	779	85.28	104.55	0.00	13.73	50.51	117.37	754.62
April	911	63.82	100.69	0.00	5.32	26.10	79.82	789.20
May	1090	70.11	99.43	0.00	5.97	32.49	92.56	727.54
June	1140	66.81	93.80	0.00	5.56	29.77	82.17	672.00
July	999	69.45	93.32	0.00	6.35	35.52	93.93	634.46
August	1440	74.02	104.44	0.00	5.71	32.15	93.18	765.62
September	1223	69.45	98.68	0.00	5.97	28.12	87.33	771.22
October	1022	69.89	96.07	0.01	6.47	32.56	87.91	757.01
November	724	76.65	104.12	0.00	7.51	34.87	98.36	732.28
December	658	130.32	148.99	0.01	16.78	65.30	186.69	812.34

Figure 6.2 shows a scatterplot of CAMS versus local irradiance data for DHI, DHI, and DNI. Table 6.3 shows the Pearson and Spearman relationship. Both show a strong correlation. However, The DNI measurement has a visual anomaly at 0 W/m^2 measurements done locally. This is likely due to the shadow not being detected by the CAMS data. Short-duration clouds could create such shadow, although the local measurement of 0 W/m^2 might indicate that something was blocking the pyranometer or a measuring error.

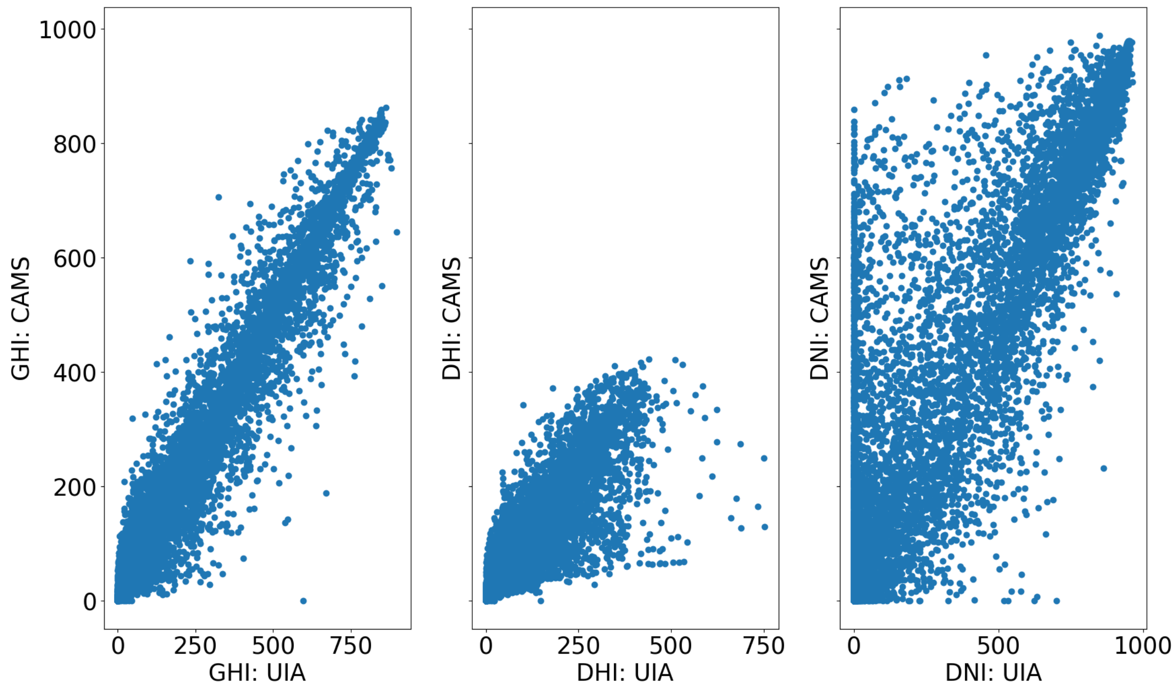


Figure 6.2: Scatterplots comparing CAMS and UIA (local) data for GHI, DHI, and DNI: This image features three separate scatterplots with UIA data on the x-axis and CAMS data on the y-axis. Left shows the GHI comparison, middle illustrates the DHI comparison and right presents the DNI comparison.

Table 6.3: Summary of correlation coefficients and linear regression parameters for GHI, DHI, and DNI scatterplots

Statistic	**GHI**	**DHI**	**DNI**
Pearson's r	0.9419	0.8412	0.8417
Pearson's p-value	0.0000	0.0000	0.0000
Spearman's rho	0.9330	0.9198	0.7822
Spearman's p-value	0.0000	0.0000	0.0000

6.2 Inference of Tilt and Azimuth

Figure 6.3, 6.4, 6.6 and 6.7 presents a whisker-boxplot of the results from the inference of tilt and azimuth based on the clearest day each month. Figure 6.3 and 6.6 being tilt, and Figure 6.4 and 6.7 being azimuth for the UIA and CAMS data respectively. The error is the absolute value of the actual value minus the calculated value. The x-axis is the percentile of best fit, used to calculate the resulting tilt/azimuth, as described in section 4.5.3. The mean is depicted as a horizontal line within each box. The 25th and 75th percentile are marked as the bottom and top parts of the colored box. The whiskers on either end are $1.5 \cdot IQR$ on their respective end. To not cause any confusion between the percentile along the x-axis and the 25th and 75th percentile of the boxplot, they are henceforth in this section called percentile, Q1, and Q3, respectively.

6.2.1 UIA: Local Data

The result from the locally measured irradiance data shows a promising result. The spread of the calculated tilt decreases as the percentile increases to about the 6th percentile; simultaneously, the mean decreases, and the median increases slightly. The Q1, median, and Q3 values get larger when using more than the 10th percentile of the data. As a result, the best tilt predictor is in the group 6th to 10th percentile. The 9th and 10th percentile of data has a mean tilt error of 10.9° and 10.8° respectively, and a median error of 8.7° and 9.4° respectively. However, the 7th and 8th percentile has a slightly lower mean and median error while the max error is larger. It is worth mentioning that these results may alter with different data.

The azimuth's mean, Q1, and Q3 results follow the same trend as that of the tilt, where the deviation of the results decreases as more data is used in the calculation process. However, a larger gap exists between the 9-10th percentile. The 10th percentile has the lowest Q3 but a slightly higher max value than the 9th percentile. 9th and 10th percentile has a mean values of 24.1° and 23.7°, median of 11.6° and 11.5°, Q1 of 5.2° and 5.2°, and Q3 of 35.4°, 29.9°, respectively. Table 6.4 shows a more extensive and exact summary of the results. The count value in Table 6.4 is the number of PV systems tested.

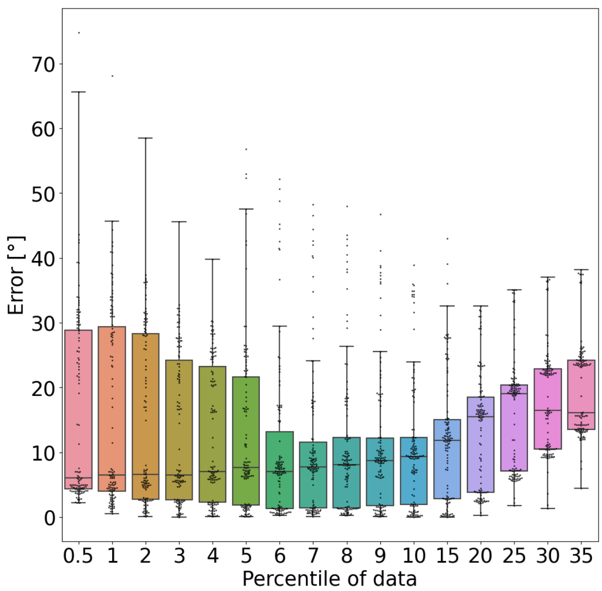


Figure 6.3: Inference of tilt: results from UIA with local data

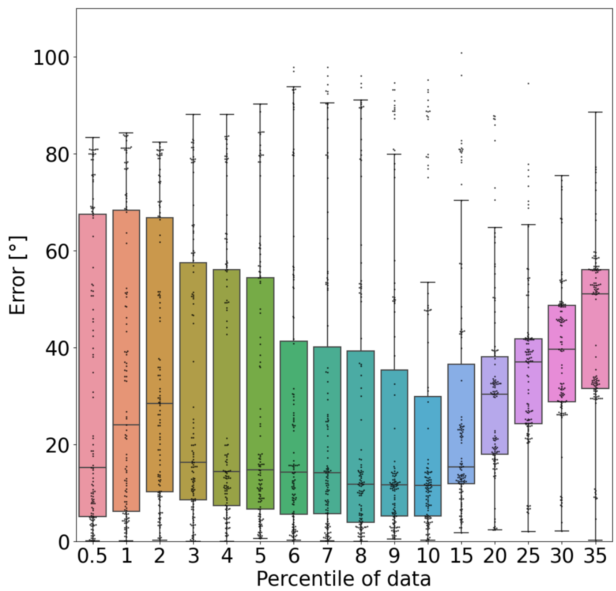


Figure 6.4: Inference of azimuth: results from UIA with local data

Table 6.4: Summary of azimuth and tilt errors for different percentile variations using local irradiance data

				Absolute azimuth error [°]				
Percentile	Count	Mean	STD	Min	Q1	Median	Q3	Max
0.5	119	31.977	30.464	0.078	5.112	15.153	67.514	83.345
1.0	119	34.280	29.906	0.057	6.157	23.995	68.266	84.200
2.0	119	35.408	27.971	0.191	10.172	28.446	66.730	82.383
3.0	119	30.541	28.245	0.001	8.524	16.300	57.409	88.057
4.0	119	29.733	28.393	0.016	7.395	14.422	56.001	88.050
5.0	119	28.505	28.329	0.522	6.598	14.788	54.383	90.200
6.0	119	27.027	29.428	0.212	5.509	14.216	41.283	97.815
7.0	119	26.992	29.619	0.052	5.739	14.080	40.140	97.804
8.0	119	25.222	29.636	0.012	3.918	11.736	39.297	95.991
9.0	119	24.124	28.975	0.422	5.152	11.601	35.365	94.614
10.0	119	23.714	28.425	0.203	5.189	11.529	29.891	95.177
15.0	119	26.872	24.957	1.756	11.908	15.353	36.522	100.785
20.0	119	31.618	21.087	2.308	17.997	30.277	38.048	115.493
25.0	119	36.247	20.218	1.981	24.224	36.973	41.736	159.864
30.0	119	41.003	23.503	2.053	28.800	39.573	48.599	182.978
35.0	119	45.692	23.619	0.185	31.507	51.085	56.073	148.851
				Absolute tilt error [°]				
Percentile	Count	Mean	STD	Min	Q1	Median	Q3	Max
0.5	119	15.732	14.585	2.176	4.306	6.000	28.856	74.800
1.0	119	15.963	14.862	0.500	4.021	6.500	29.403	68.139
2.0	119	14.559	13.409	0.076	2.741	6.542	28.312	58.500
3.0	119	13.152	11.410	0.013	2.621	6.526	24.216	45.557
4.0	119	12.433	10.566	0.039	2.274	7.065	23.243	39.742
5.0	119	12.774	12.801	0.059	1.863	7.604	21.652	56.833
6.0	119	10.326	12.014	0.231	1.331	7.065	13.192	52.161
7.0	119	10.542	11.245	0.078	1.379	7.709	11.555	48.225
8.0	119	10.797	11.070	0.229	1.438	8.083	12.283	48.000
9.0	119	10.942	10.758	0.117	1.765	8.701	12.212	46.757
10.0	119	10.818	9.991	0.013	1.982	9.380	12.268	38.894
15.0	119	11.943	9.710	0.031	2.853	11.849	15.068	42.986
20.0	119	13.269	8.794	0.247	3.842	15.498	18.539	32.544
25.0	119	15.772	8.571	1.766	7.148	19.042	20.369	35.057
30.0	119	17.825	8.064	1.352	10.498	16.433	22.873	37.046
35.0	119	19.588	7.203	4.444	13.494	16.097	24.214	38.152

Figure 6.5 shows the resulting azimuth and tilt combinations for all the panels when the 15th percentile is used. The panels with an east direction have all calculated a tilt between $20 - 30°$. As the actual tilt is $10°$, this aligns with the results from table 6.4 where the mean error is $11.943°$. Noticeably, the panels with a west direction have a smaller tilt angle, corresponding well with the actual tilt. There also seems to be a small cluster of panels with an azimuth of approximately $180 - 190°$. These are highly likely caused by shading, as further discussed in section 6.3

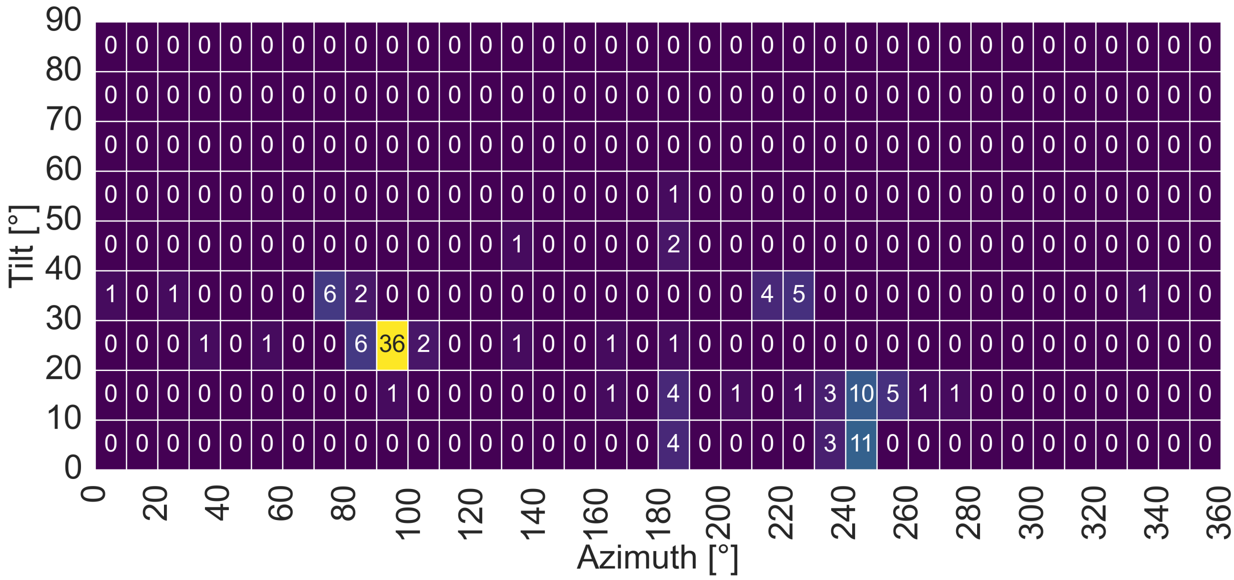


Figure 6.5: Tilt and Azimuth matrix: UIA, irradiance measurement recorded locally on UIA. The matrix shows the resulting tilt and azimuth of the inference of tilt and azimuth using local irradiance measurements from UIA and the available PV systems on UIA. The 15th percentile results are shown.

6.2.2 UIA: CAMS Data

This section includes the result for calculating tilt and azimuth when irradiance measurements from the CAMS service Heliosat-4 have been used. As a result of the CAMS data possibly being more inaccurate than the local pyranometer measurements, a higher error deviation was presumed to be found. However, the result in Figure 6.7 is similar to that where local irradiance data has been used. The result in the 0.5 to 3rd percentile has a wide IQR range with respect to the other percentiles, similar to the previous results. Furthermore, the most optimal range for calculating the tilt is in the 4th-10th percentile, similar to when local irradiance data were used. Overall the result is quite similar. The mean, Q1, and Q3 at 10.1°, 2.8°, and 14.0° respectivly at 9th percentile.

Regarding Figure 6.7 and the azimuth error, the percentile range of 6th-15th is among the data with the lowest mean, Q1, and Q2. 9th percentile has a mean error of 24.4°, Q1 of 8.1°, and Q2 of 35.3° and max of 98.6°. However, the 15th percentile improved from the result using local irradiance measurements, with a mean error of 22.4°, Q1 of 8.4°, and Q3 of 24.1° and a max of 93.9°. Table 6.5 shows a more extensive and exact summary of the resulting percentile groups. The count value is the number of PV systems tested.

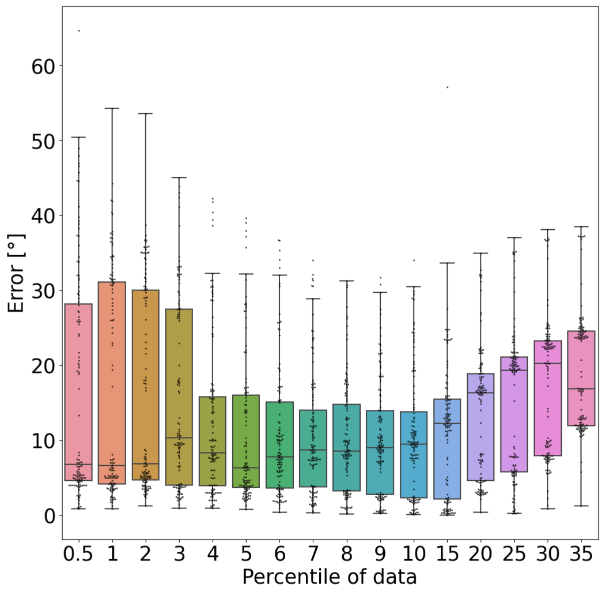


Figure 6.6: Inference of tilt: results from UIA with CAMS data

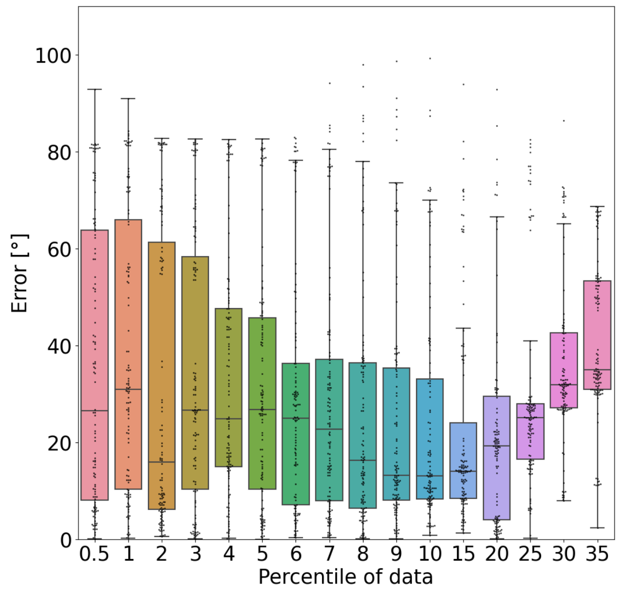


Figure 6.7: Inference of azimuth: results from UIA with CAMS data

Table 6.5: Summary of azimuth and tilt errors for different percentile variations using CAMS data

				Azimuth error				
Percentile	Count	Mean	STD	Min	Q1	Median	Q3	Max
0.5	120	35.495	28.999	0.117	8.112	26.477	63.721	92.800
1.0	120	37.519	28.279	0.200	10.289	30.955	65.965	90.943
2.0	120	31.152	30.152	0.560	6.186	15.924	61.313	82.650
3.0	120	34.495	27.492	0.099	10.286	26.693	58.292	82.574
4.0	120	33.126	24.576	0.200	14.925	24.848	47.552	82.485
5.0	120	30.882	24.532	0.026	10.305	26.788	45.640	82.639
6.0	120	28.966	24.484	0.359	7.115	24.964	36.255	82.966
7.0	120	28.433	24.838	0.300	7.974	22.740	37.161	94.075
8.0	120	25.743	25.188	0.061	6.399	16.321	36.424	97.927
9.0	120	24.429	24.318	0.033	8.075	13.137	35.331	98.621
10.0	120	23.973	23.212	0.774	8.286	13.026	33.089	99.232
15.0	120	22.360	21.698	1.335	8.445	14.068	24.070	93.890
20.0	120	26.490	32.576	0.063	4.041	19.233	29.456	161.204
25.0	120	30.766	27.622	0.231	16.465	25.069	27.964	236.600
30.0	120	39.259	29.411	7.960	27.136	31.897	42.587	252.200
35.0	120	44.670	30.663	2.388	30.949	34.924	53.271	253.087
				Tilt error				
Percentile	Count	Mean	STD	Min	Q1	Median	Q3	Max
0.5	120	16.389	15.173	0.848	4.594	6.744	28.125	64.664
1.0	120	16.418	14.470	0.854	4.131	6.565	31.065	54.219
2.0	120	15.592	13.390	1.200	4.641	6.867	30.004	53.545
3.0	120	15.664	12.375	0.892	3.976	10.331	27.438	45.000
4.0	120	12.155	10.415	0.870	3.882	8.258	15.786	42.214
5.0	120	10.756	9.818	0.727	3.669	6.301	16.017	39.640
6.0	120	10.579	8.857	0.398	3.631	7.751	15.059	36.729
7.0	120	10.658	8.116	0.279	3.750	8.662	14.028	33.976
8.0	120	10.041	7.359	0.108	3.238	8.488	14.772	31.250
9.0	120	10.130	7.563	0.211	2.760	9.014	13.950	31.731
10.0	120	10.476	8.198	0.091	2.318	9.486	13.747	34.000
15.0	120	11.390	8.989	0.003	2.165	12.210	15.448	57.078
20.0	120	13.712	8.637	0.404	4.565	16.289	18.861	34.892
25.0	120	14.894	9.755	0.190	5.727	19.283	21.038	36.975
30.0	120	17.156	9.186	0.841	7.880	20.242	23.230	38.098
35.0	120	19.142	7.867	1.213	11.943	16.874	24.500	38.507

Figure 6.8 shows the resulting angle and tilt combinations for all the panels. Again, the result is similar to where local irradiance data is used, as the east-oriented panels are estimated with a steeper tilt angle than the west-oriented panels and a cluster of wrongly estimated angles at around 200°.

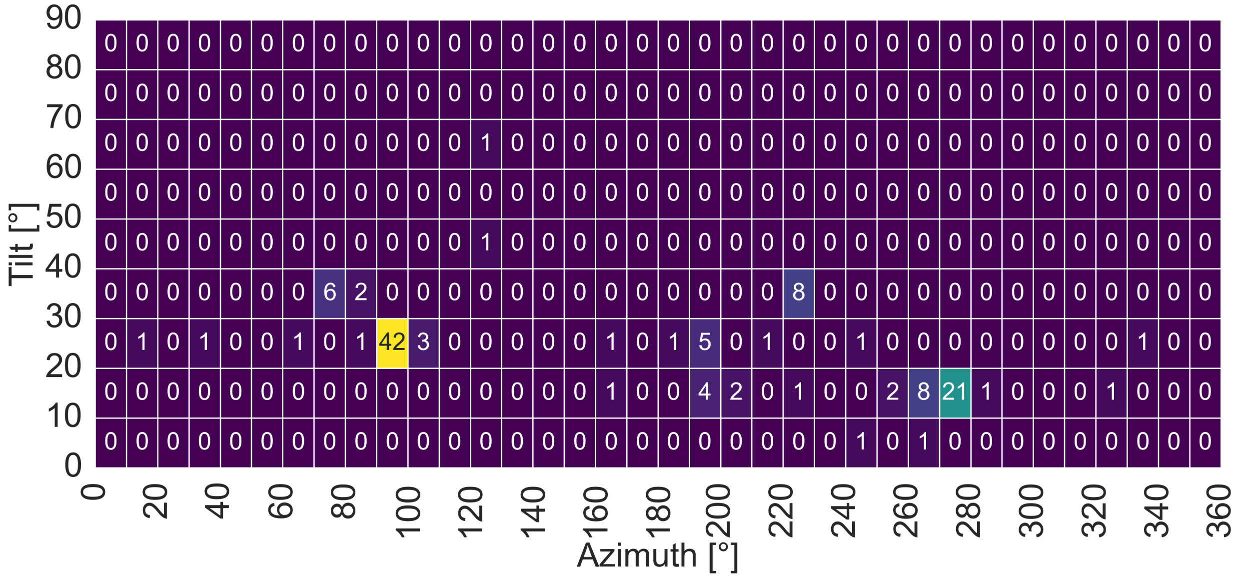


Figure 6.8: Tilt and azimuth matrix: UIA, irradiance measurement from Heliosat-4. The matrix shows the resulting tilt and azimuth of the inference of tilt and azimuth using CAMS irradiance measurements for the location of UIA and the 15th percentile of data.

6.2.3 Solcellespesialisten‘s Data

Figure 6.9 shows the result from the inference of tilt and azimuth method from section 4.5 on Solcellespesialisten's data. Most PV installations appear to have an azimuth between 100 and 270 degrees, corresponding with optimal azimuth. However, there is a noticeable gap in the azimuth range of 180° to 200°, with fewer PV installations. Instead, the installations are clustered on either side of this range.

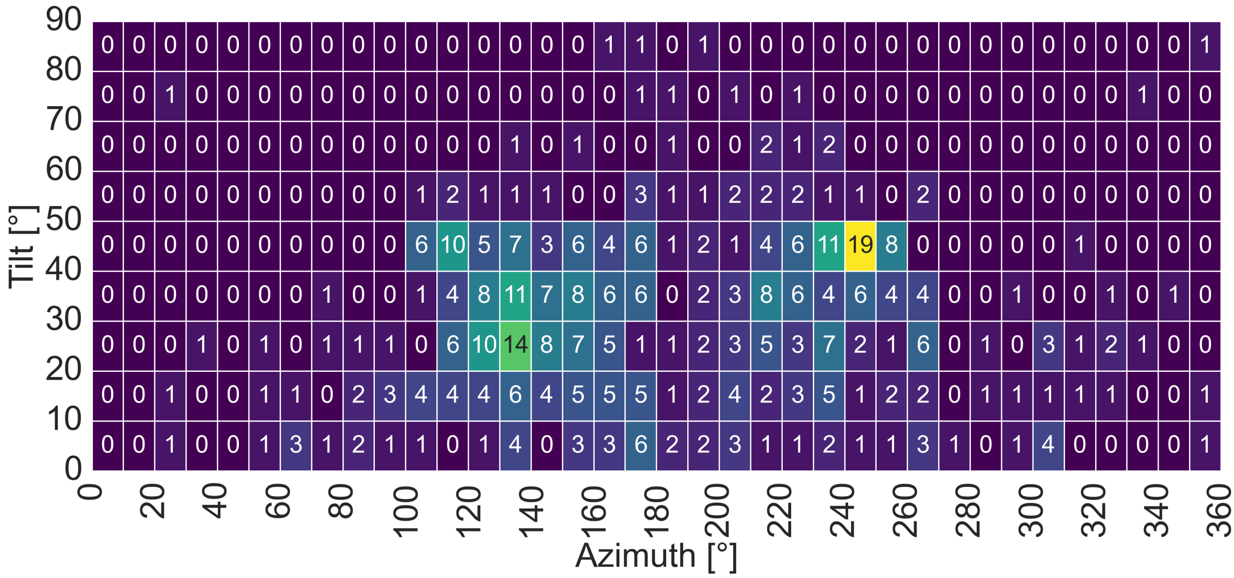


Figure 6.9: Tilt-Azimuth heatmap of PV distribution in Solcellespesialisten‘s data: The heatmap shows the distribution of the tilt-azimuth of the PV installations. The x-axis represents the azimuth, and the y-axis represents the tilt. Both axes are in 10-degree intervals. North is 0°, and south is 180°

6.3 Effect of Shading

Figure 6.10 presents an image from the roof of UIA, where the solar panels are installed. As seen in the figure, a fence is installed around the roof's perimeter, leading to a small amount of shading. In addition, global horizontal irradiance (GHI), diffuse horizontal irradiance (DHI), and direct normal irradiance (DNI) pyranometers are installed on a solar tracker in the foreground on the southern side of the installation. Figure 6.11 illustrates the placement and numbering of the panels and their number. The red circle depicts the pyranometers on the southern side. Where tilt or azimuth is calculated with an error greater than 20° is marked with the color black. The black marking mainly follows the installation's perimeter, corresponding to shading from the fence and the panels closest to the pyranometers. Hence there seems to be a correlation between shading and more significant errors in the azimuth and tilt estimation.

Figure 6.10: Northwest-oriented view of the PV installation on the roof of the University of Agder. GHI, DHI, and DNI pyranometers are installed in the image's foreground. Source [84]

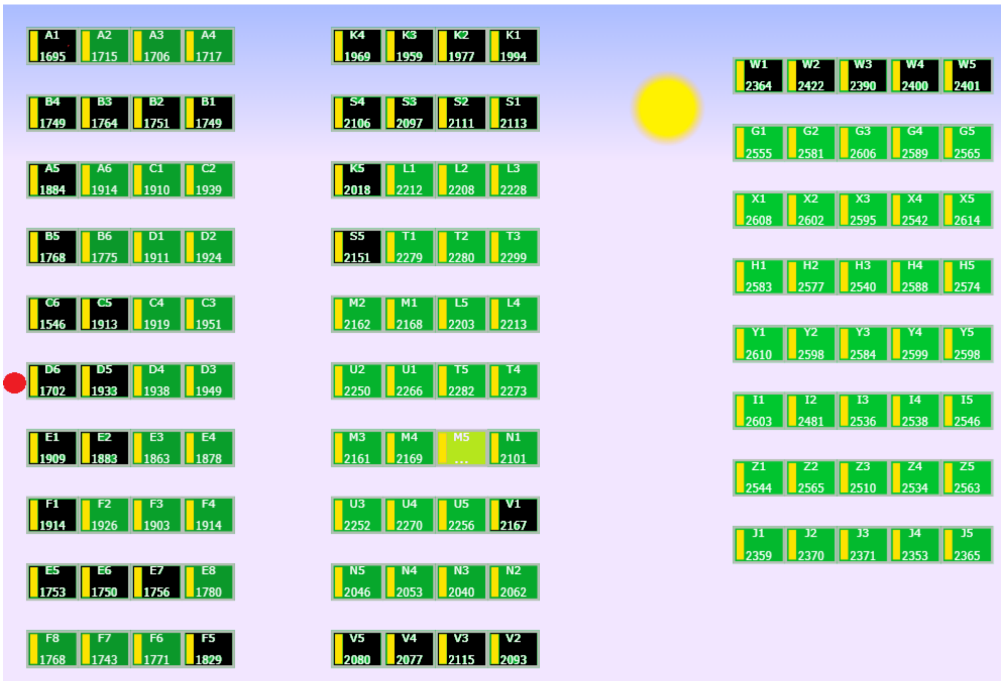


Figure 6.11: Illustration of the PV installation on the roof of the University of Agder. The orientation of the image is; top (west ≈ 263.2°), left (north), bottom (south ≈ 83.2°), left(south). The red dot roughly illustrates the position of the pyranometers, a source of shading. Modules, where tilt or azimuth is calculated with an error greater than 20°, are marked in black. Source: [84]

Figure 6.12 shows the logged power data for 2020-07-22. Panel A1-A4 is the top left row, and A5 is the leftmost panel on the third row from the top in figure 6.11. A1-A5 is facing east. A decrease in produced power is visible after 14:00. The panels V2-V5 are installed at the bottom in the second row from the left. They have a reduction in power before 12:00. The shading of the railing likely causes both reductions, as it cannot be seen in the C5-C6 or Y1-Y5 panels. The C6 panel does, however, have a decrease midday; this is aligned with the sun being in the southern direction and is, therefore, most likely caused by shading from the pyranometer. The reason this is not reflected in C5 is likely the high altitude of the sun during July, causing a shorter shadow from the pyranometer that only affects C5. The short-duration spikes in power reduction during the day might be short lasted shadows by clouds. However, the timing does not match exactly for all of the shown panels due to this power reduction might also be caused by someone walking on the roof. Appendix A contains a table of the modules where the inference was affected by shading. This shows that the power curve is influenced differently depending on the level of shading and, therefore, the inference of tilt and azimuth, where significant inaccuracies may occur.

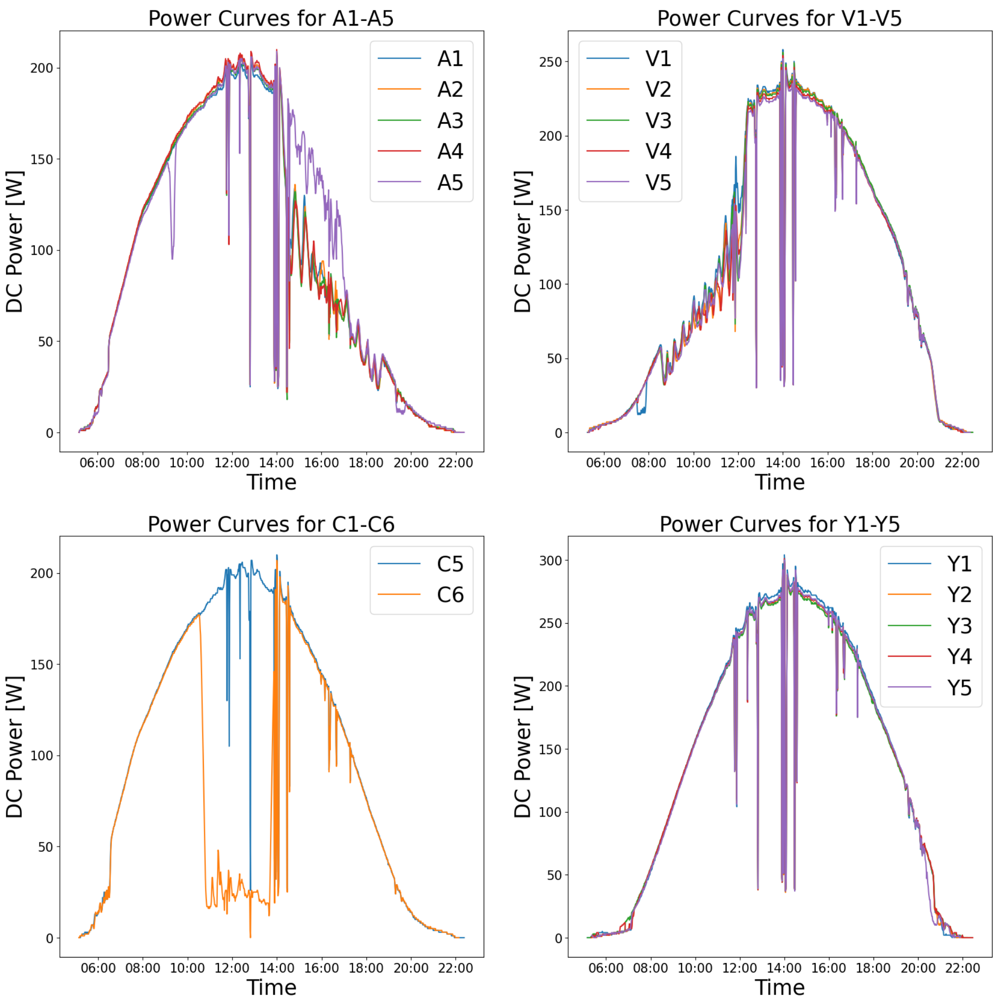


Figure 6.12: Recorded power curve in 5-min data for panels A1-A5, V1-V5, C5-C6, and Y1-Y5. The panels in groups A, V, and C had a calculated tilt or azimuth degree greater than 20°. Y1-Y5 is shown as a reference, as close to no shading was present and inferred a low tilt error below 10°.

6.3.1 Clearest Day

The clearest day has been found using equation 2.10. The result can be seen in table 6.6 where they are separated into two columns; Clearest day with the use of CAMS weather data, and with the use of local weather data, both columns shows the result from UIA.

Table 6.6: Result of the clearest day each month using irradiance data

Location: UIA Year: 2020 Data: CAMS		Location: UIA (Local) Year: 2020 Data: Local	
Month	**Clearest day**	**Month**	**Clearest day**
1	29	1	29
2	19	2	19
3	5	3	21
4	22	4	22
5	31	5	25
6	15	6	24
7	22	7	22
8	16	8	14
9	2	9	15
10	14	10	16
11	6	11	6
12	25	12	24

6.4 Capacity Distribution

The majority of the capacities are below 10 kW_p. There are 462 facilities in total, where the Q1, median, and Q1 is 5, 7, and 10 kW_p, respectively. There are also some very large installations at 5000 kW_p to 11000 kW_p.

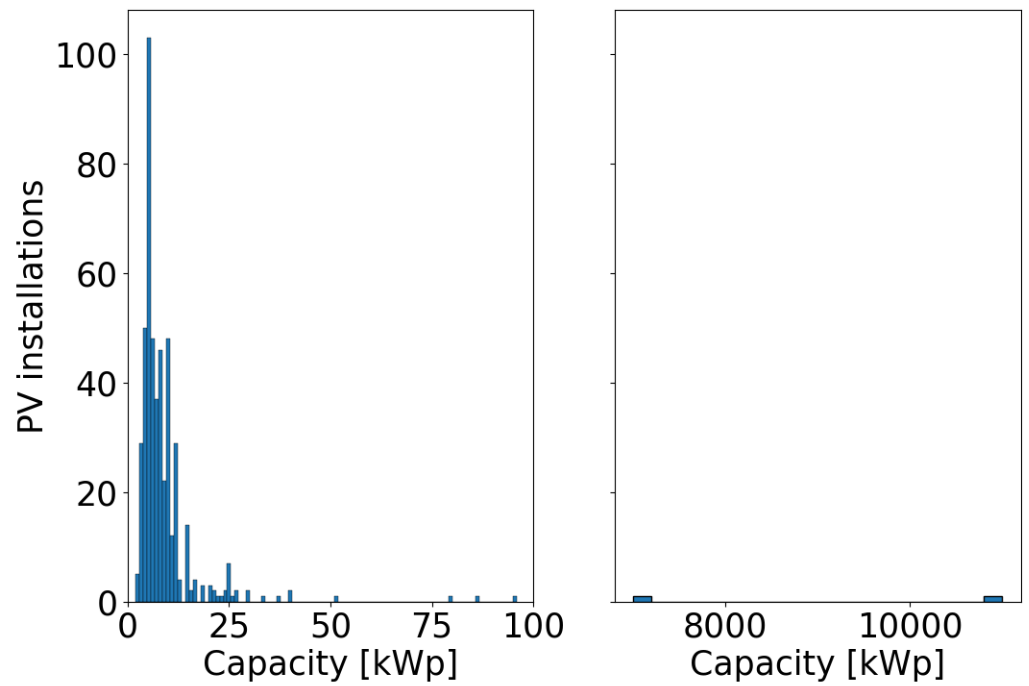


Figure 6.13: Raw data: Distribution of capacity

6.5 Performance Ratio

Figure 6.14 shows the resulting PR for all the installations using raw data (6.14a, 6.14b), RANSAC inliers (6.14c, 6.14d), and polynomial inliers (6.14e, 6.14f). The RANSAC filter redistributes the PV installation's PR values, as seen in the difference between figure 6.14a and 6.14c. The RANSAC regression improves the result as some low and high PR installations are adjusted to align with the most frequent PR values from the unfiltered plot. For example, the number of PV installations that get a PR close to 0.5 slightly decreases as these are shifted to a higher PR. This indicates that these installations had a large spread in $Y_f - Y_r$ scatterplot data and that CAMS irradiance data was too high, leading to a low PR. There is also a slight decrease in the number of installations getting a PR close to 1. This is likely due to the RANSAC regression finding an optimal fit and getting a more plausible PR value; it also indicates the opposite of the low PR values that changed, specifically that the irradiance from CAMS was too low, leading to a higher PR. The fitted polynomial inliers further align the PV installation's PR with that of the most frequent values. However, it also increases the number of installations getting a high PR, close to 1. This is likely due to fewer data at the higher portions of the $Y_f - Y_r$ scatterplot and the polynomial filtering narrowing in too much, only including the uppermost inliers. An example is the installation in figure 6.22. This result may, therefore, be over-filtered. Therefore, a PR of 0.83 from the RANSAC filtering is considered to describe the dataset best.

Moving on to the regional PR, Figure 6.15 shows the regional PR values for datasets 1), 2), and 3). The number above each boxplot highlights the number of available PV installations. Appendix B contains tables with exact values. The fact that counties close to one another have similar PR might indicates that the inferred results are correct. However, very few statistical differences could be found in Table 6.7, which utilized the Dunn‘s and Mann-Whitney U-test. Testing dataset 1), only Rogaland and Akershus could be seen as statistically different. Moreover, Rogaland differs from Akershus and Østfold when using dataset 2), and no difference was found with dataset 3). Therefore, with Rogaland, Akershus, and Østfold being the top three countries with the most data, it can be assumed that the answer is valid and has some differences between counties. However, the difference in results for the statistical test from the three datasets highlights that the answer depends on the filtering process applied. Another factor is that some counties have less data, and statistical differences are harder to spot. Due to the possibility of overfitting, Figure 6.15b and Table B.2 in the Appendix are seen to reflect the given dataset the best.

One theory for the identified statistical differences in PR values between counties is that there may be more snow in Akershus and Østfold. Appendix F, G, and H includes the monthly PR for the different counties for datasets 1), 2), and 3), respectively. They show that the PR in January and December significantly increases when applying the RANSAC filter, possibly due to removing 0 Y_f values. Removal of 0 Y_f values might remove instances where snow is present. Other possibilities for the statistical difference are variations in CAMS irradiation data in different counties, roof azimuths, shading variations, and thus variations in the accuracy of the inferred result of tilt and azimuth between counties.

Figure 6.16 show the PR values distributed across the tilt and azimuth of the corresponding PV installation. Due to the low amount of installations per tilt/azimuth degree, trends are difficult to detect with certainties.

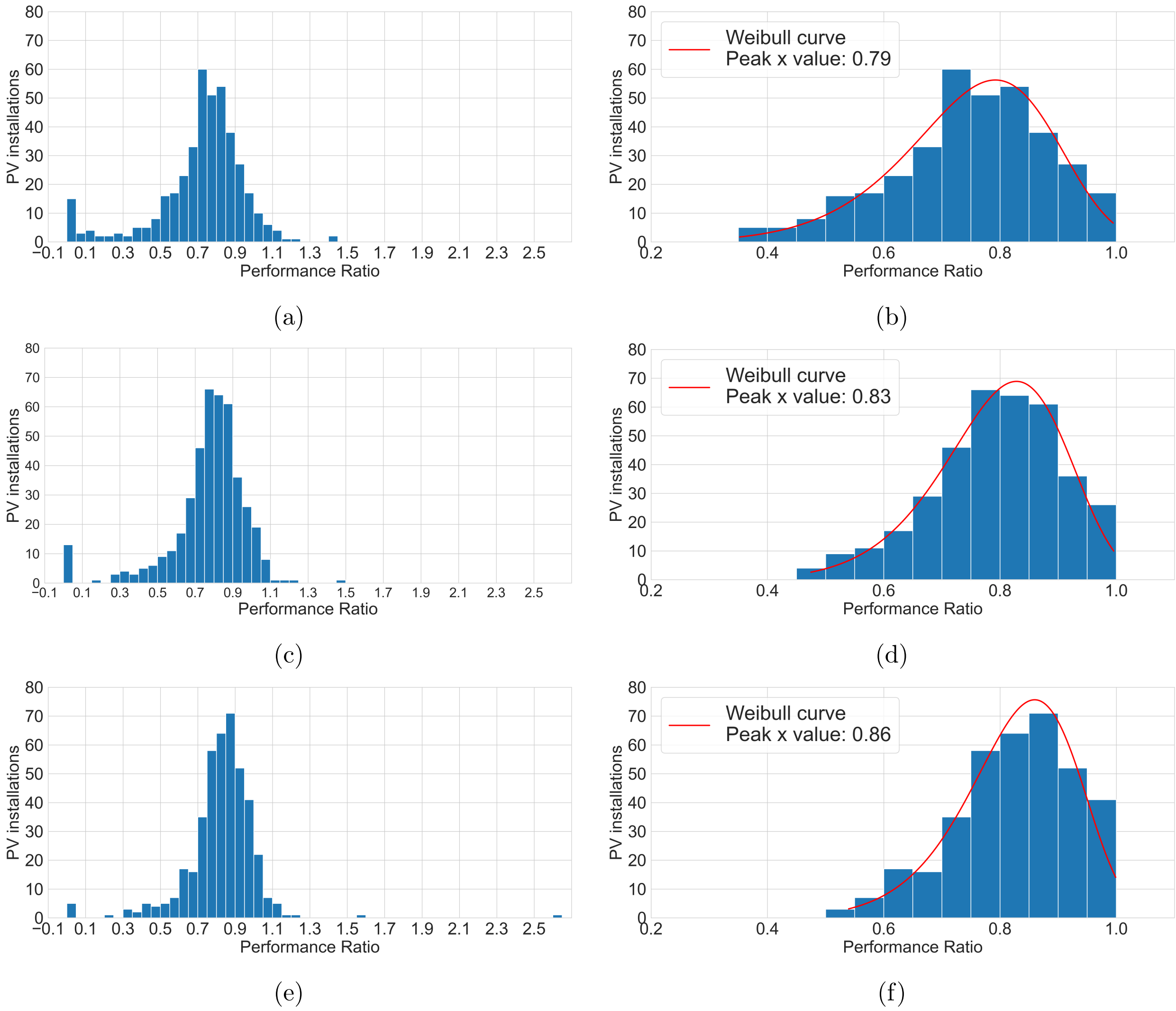
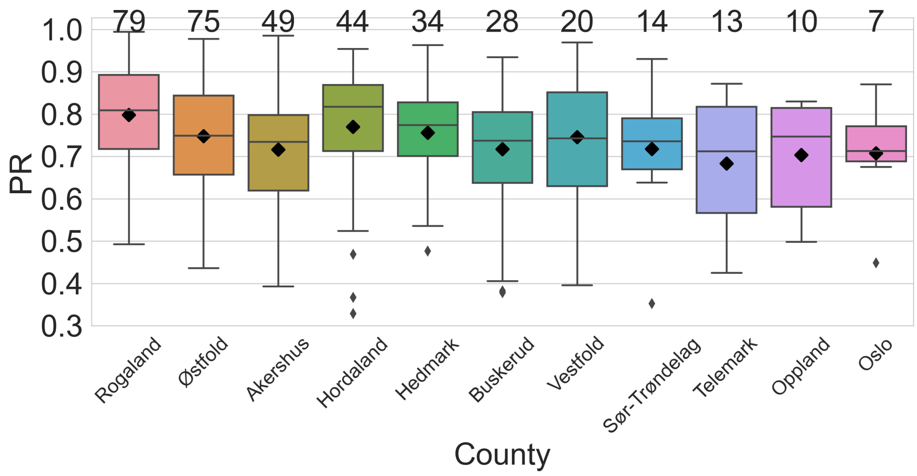
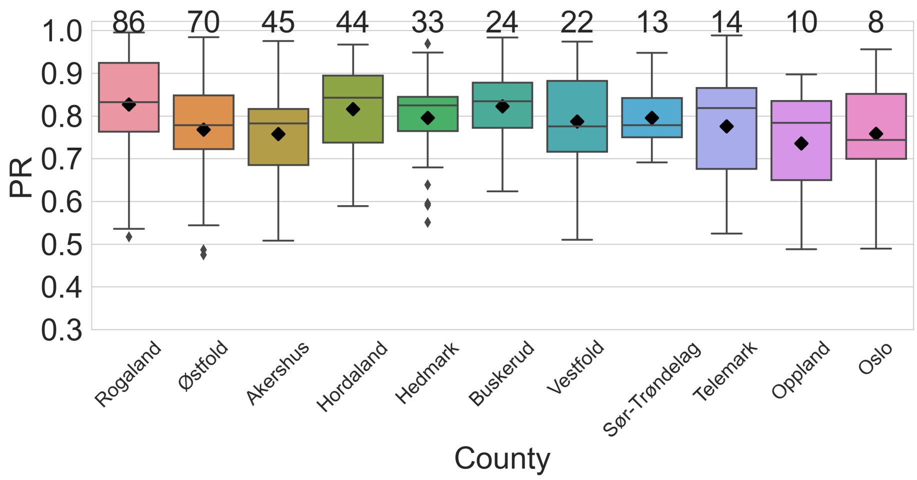


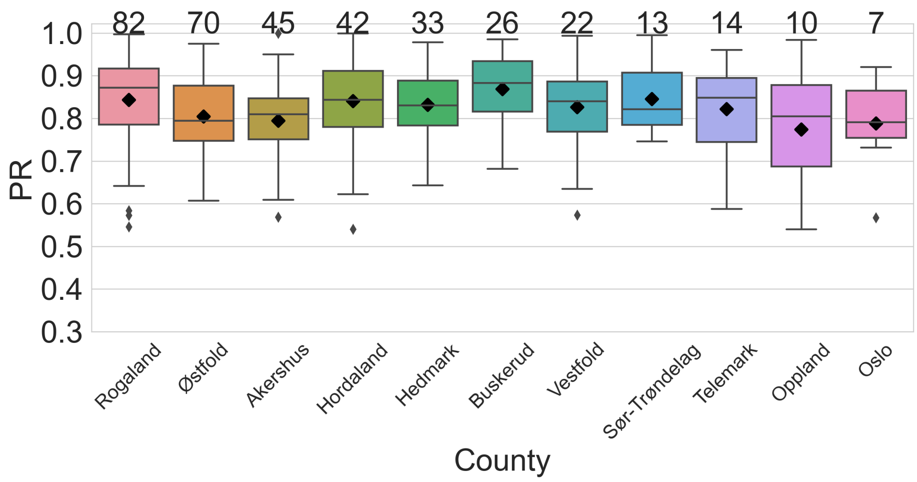
Figure 6.14: Distribution of integrated Performance Ratio: PR values are computed using the inferred orientation from section 6.2.3. Y-axis represents the percentage of data falling into each PR value range, while the x-axis displays the PR value. The bin size is 0.05 along the x-axis. (a): Using all data and (b): including Turkey's filter. (c): Using data from the RANSAC regression and (d): including Turkey's filter. (e): Using data from the Polynomial fit and (f): including Turkey's filter.



(a) All data: Boxplot of PR for each county

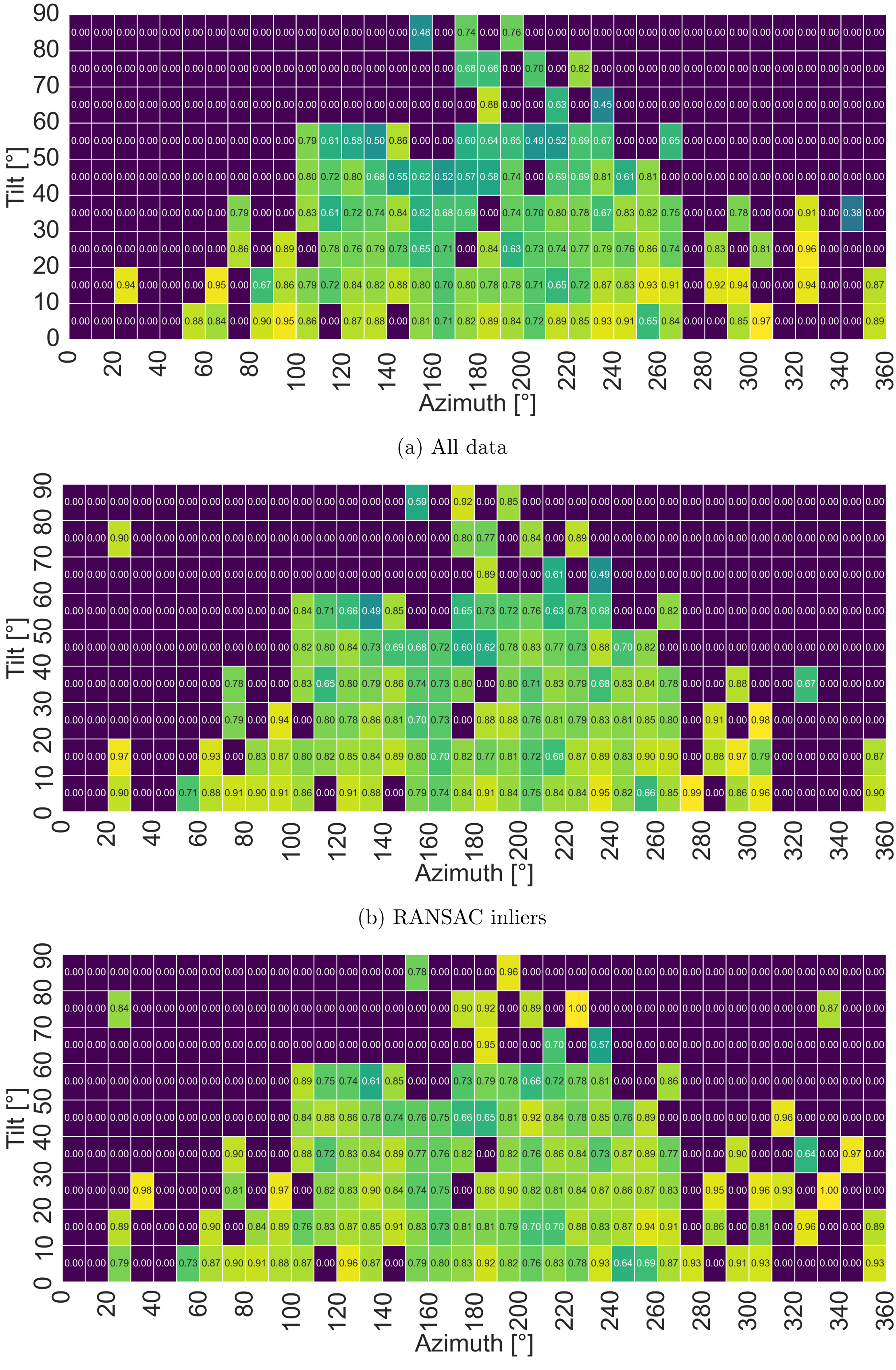


(b) RANSAC inliers: Boxplot of PR for each county



(c) Polynomial inliers: Boxplot of PR for each county

Figure 6.15: Boxplots of PR values across counties for different data processing techniques. Data is processed using three different datasets: All Data (6.15a), RANSAC inliers (6.15b), and Polynomial Inliers (6.15c). In all cases, Tukey's Method is applied, and PR values above 1 are excluded.



(a) All data

(b) RANSAC inliers

(c) Polynomial inliers

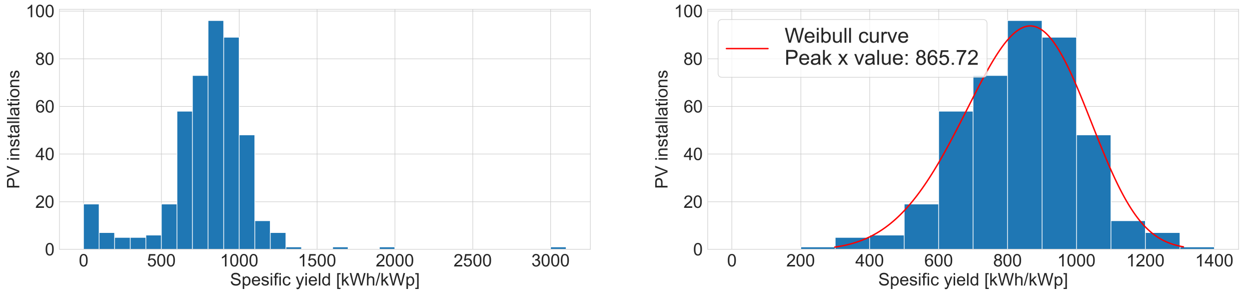
Figure 6.16: Heatmap matrices illustrating the PR for various tilt and azimuth angles in 10-degree intervals. Data is processed using three different datasets: All Data (6.16a), RANSAC inliers (6.16b), and Polynomial inliers (6.16c). Tukey's Method is applied in all cases, and PR values above 1 are excluded.

Table 6.7: Counties with significant differences in PR

County	All Data	RANSAC Inliers	Polynomial Inliers
Rogaland	Akershus (p=0.0442)	Østfold (p=0.0263), Akershus (p=0.0263)	None
Hordaland	None	None	None
Østfold	None	None	None
Akershus	None	None	None
Buskerud	None	None	None
Hedmark	None	None	None
Sør-Trøndelag	None	None	None
Oslo	None	None	None
Vestfold	None	None	None
Telemark	None	None	None
Oppland	None	None	None

6.6 Specific Yield

Figure 6.17a shows the specific yield of all PV installations in dataset 1) All data. Figure 6.17b shows the remaining data after filtering with Turkey's method. The yearly specific yield for the whole dataset is inferred to be 866 kWh/kW$_p$.



(a) Unfiltered data. (b) Filtered data.

Figure 6.17: Infered specific yield for dataset 1

Figure C.1 displays the specific yield result for the different counties; exact numbers can be seen in appendix C. Table 6.8 displays the result of the Turkey HSD test. More statistical differences were detected for the specific yield than for the PR. Furthermore, most statistical differences appear not to be located close to one another, which might indicate a valid result. These differences could be due to geographical differences, such as the amount of irradiance and weather. However, they could also be due to the dataset itself, including differences in tilt, orientation, variation in shading, and corresponding variance in the accuracy of inferred tilt and azimuth. Oslo had the highest Weibull curve peak; however, it does not have the highest mean or median value, indicating that the Weibull fit might not be the best for Oslo, as there are few PV installations. Figure 6.20 shows the specific yield for the tilt and azimuth of the corresponding PV installation. Moreover, as with PR, the lack of data limits further analysis regarding tilt and azimuth.

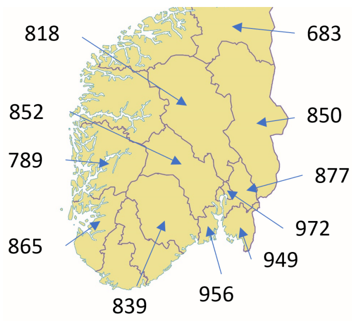


Figure 6.18: Map of yearly specific yield kWh/kW$_p$. Background image from [85]

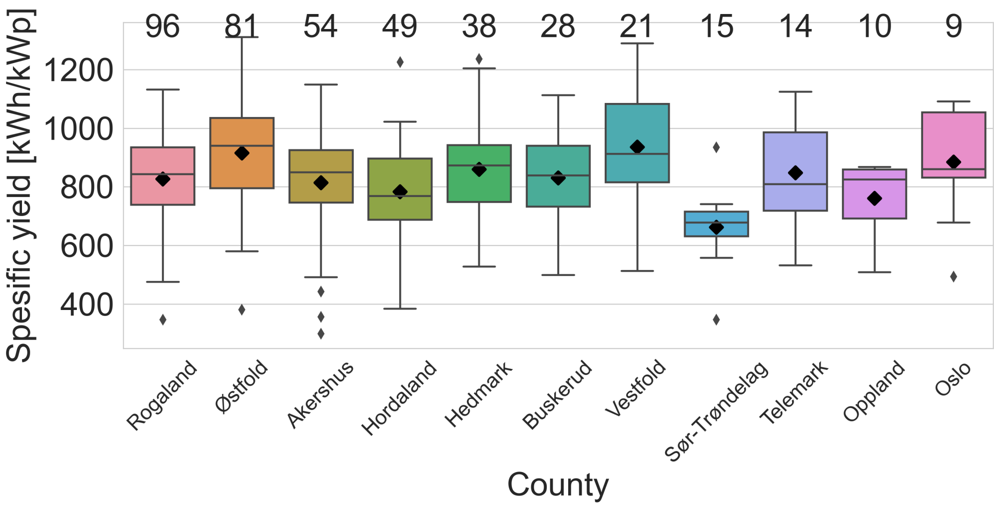


Figure 6.19: Boxplots of specific yield across counties

Table 6.8: Counties with significant differences in specific yield

County	Significantly different from
Rogaland	Sor-Trondelag (p=0.0142)
Hordaland	Ostfold (p=0.0005), Vestfold (p=0.0171)
Ostfold	Rogaland (p=0.0156), Sor-Trondelag (p=0.0000)
Akershus	Ostfold (p=0.0197)
Buskerud	None
Hedmark	Sor-Trondelag (p=0.0042)
Sor-Trondelag	Vestfold (p=0.0001)
Oslo	None
Vestfold	None
Telemark	None
Oppland	None

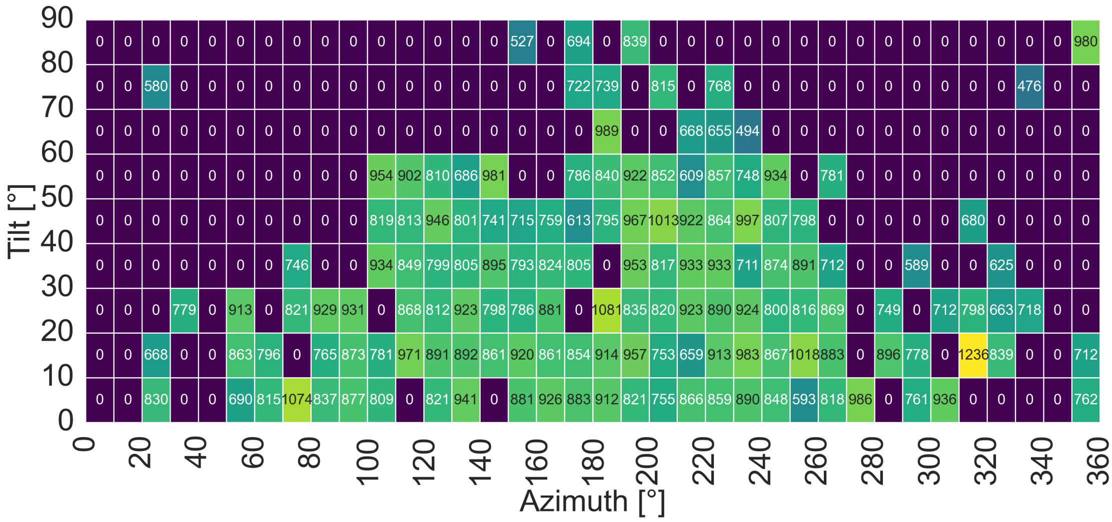


Figure 6.20: Heatmap matrices illustrating the specific yield for various tilt and azimuth angles in 10-degree intervals. Tukey's Method is applied

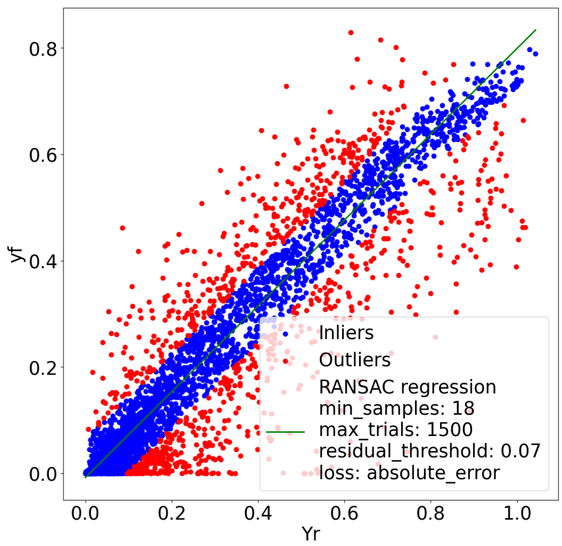
6.7 Clustering

This section illustrates the result of the RANSAC inliers and the polynomial fit. Due to space limitations, not all 448 PV installations are shown. In this section, two examples are highlighted, one considered an acceptable fit and one considered insufficient. Appendix J includes some more results.

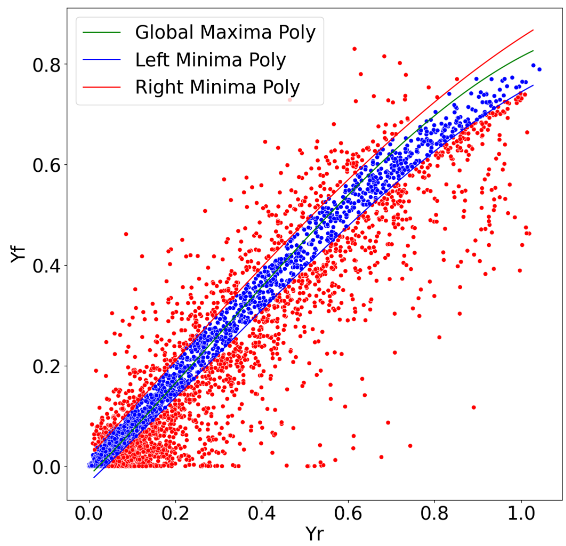
Figure 6.21 is considered an acceptable fit; The graph shows a close-to-linear trend. This is expected, as a high amount of irradiation correlated with high module temperature and, therefore, less efficiency at higher irradiance instances. Figure 6.21c shows the bins. The first three bins (starting at the top left, as these are closest to 0 on the Y_r axis) show a defined distribution (Normal, Weibull, among others). This clearly defined distribution makes the process of defining where the inlier/outlier limit is. However, as the binning progresses, this limit gets less defined. Nevertheless, the binning process works well enough for a tolerable fit.

Figure 6.22 illustrates what is seen as an insufficient result. The RANSAC regression in Figure (a) is sufficient, as it finds a plausible linear regression. The width of the inliers also seems decent, altho it could have been wider to allow for more data in the binning process.

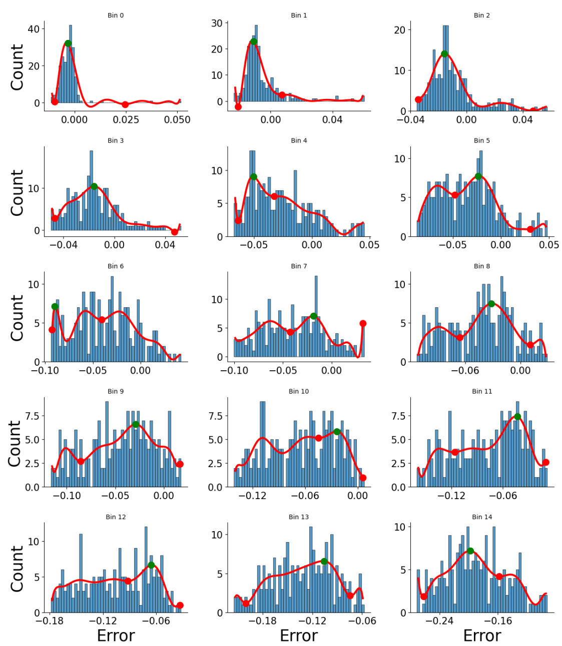
The binning process in fig 6.22c has a less defined distribution, both in the first bins (top left, which is closest to 0 on the Y_r axis) and the later bins. Due to this, maxima points (se 6.22d) shift toward higher values, resulting in a bad fit. Therefore, the main problem with this filtering is that the inlier/outlier limit is hard to detect. Leading to poorly chosen maxima and minima points. A lack of data points in this dataset at ($\approx Y_f$:0.6, Y_r:0.64) is also visible; this could somewhat contribute to the inferior fit.



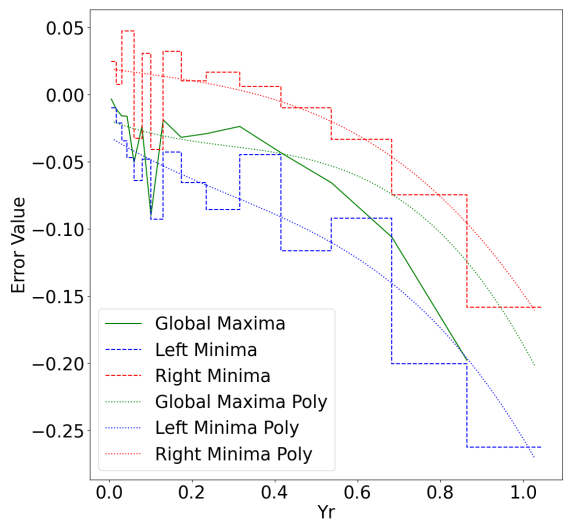
(a) RANSAC fit



(b) Polynomial fit

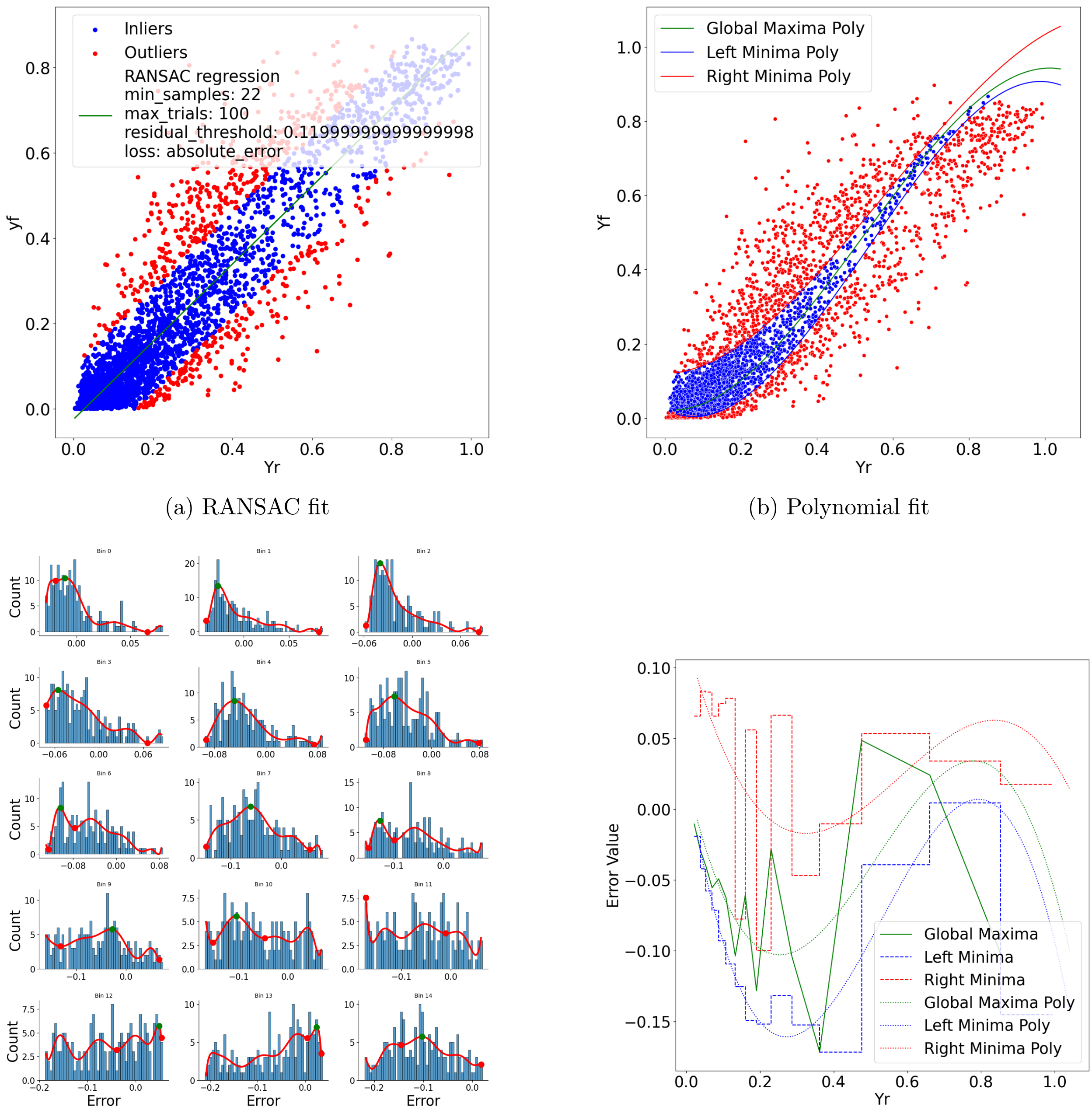


(c) Histograms



(d) Polynomial borders

Figure 6.21: Clustering Figure 1: Acceptable fit. (a) RANSAC fit: is the result of the section 4.6.3. (b) Polynomial fit: is the result of section 4.6.4, and shows the fitted left and right polynomial curves. (c) Histograms: shows the histograms of the bins the data has been grouped into; it also shows the maxima point as a green dot and the left/right minima as a red point. (d) Error graph: shows the maxima and left/right minima error values. The error value is gathered from the x-axis value figure (c), where the error is calculated by equation 4.1.



(a) RANSAC fit (b) Polynomial fit

(c) Histograms (d) Polynomial borders

Figure 6.22: Clustering Figure 1: insufficient fit. (a) RANSAC fit: is the result of the section 4.6.3. (b) Polynomial fit: is the result of section 4.6.4, and shows the fitted left and right polynomial curves. (c) Histograms: shows the histograms of the bins the data has been grouped into; it also shows the maxima point as a green dot and the left/right minima as a red point. (d) Error graph: shows the maxima and left/right minima error values. The error value is gathered from the x-axis value figure (c), where the error is calculated by equation 4.1.

Chapter 7

Discussions

7.1 Data and Metadata

PV installation data was given by Solcellespesialisten; however, the need for more information regarding the units created some uncertainties. The unit of installation capacity was solved by estimating it to be in kW_p, thereafter calculating the specific yield, and adjusting the capacity in factors of 1000 (W_p, kW_p, and MW_p) and utilizing the unit that gave the most likely specific yield. However, some PV installations gave no likely answer when the capacity unit was set to be in W_p, kW_p, or MW_p. These values were removed from the dataset based on not overfitting the result.

The need for more information regarding the timezone of the timestamps is also a source of inaccuracies in the study. However, the timezone was located by manually checking for the correlation between the power data and sunrise/sunset and irradiance data, which has a known timezone. The result can be verified by getting accurate results in inferring the tilt and azimuth on a known PV installation such as UIA. However, more than correlation control is difficult when the tilt and azimuth are also unknown.

The result is also volatile regarding annual fluctuations in irradiance, as the dataset is limited to one year. A dataset over multiple years, such as ten years, would give a more representative result regarding such deviations. This thesis's developed method for analysis is scalable to such a dataset.

Regarding expanding the metadata, recommendations are to standardize the logged PV capacity unit and display the timezone and the measurement location of temperature data. Furthermore, giving the owner of the PV installation the ability to log the tilt and azimuth would be a great inclusion. Finally, in the dataset analyzed, the creation date of the PV installations matched the start of the dataset better than possible creation data; it is uncertain what caused this, but including an accurate creation date would allow for further analysis, such as year-over-year degradation. Furthermore, including module and inverter type/brand would also allow a more detailed analysis.

7.2 Comparative Analysis of Local and CAMS Irradiance Measurements

The GHI, DHI, and DNI measurements from CAMS for the location of UIA, Grimstad, are compared against the local GHI, DHI, and DNI measurements. Nighttime has been removed from both datasets to not account for similarity during the nighttime. The data is aggregated to hourly format and consists of $11,048$ hours from 04-08-2019 to 03-12-2021.

The accuracy of the GHI and DHI measurements was quite similar. The DNI measurement had the largest deviance at a median value of 44.3 W/m^2. As stated in the literature review, the CAMS quarterly report [67] found an absolute average error of 16 W/m^2 for the GHI, 15 W/m^2 for DHI, and 27 W/m^2 for DNI. The results from this study found an absolute average error of 31.1 W/m^2 for GHI, 30.5 W/m^2 for DHI, and 90 W/m^2 for DNI. Therefore, the values found in this study are significantly higher, especially for the DNI value. This might be due to Norway being near the satellite viewing edge and cloud/snow detection being less accurate as a result [66]. The authors of another report [66] found a correlation between the Heliosat-4 and local measurements; these results follow that of which is found in this study. [66] found a Pearson correlation of 0.90-0.96 (GHI), 0.68-0.87 (DNI), and 0.68-0.87 (DHI). The results found in this study are 0.94 (GHI), 0.84 (DHI), and 0.83 (DNI). It is essential to mention that the expected correlation result in [66] was with 15-min data and hourly data was used in this study. Overall, the correlation aligns with what is expected, but mean values deviate from the expected results.

7.3 Inference of Tilt and Azimuth

The utilized method can accurately calculate the tilt and azimuth, with a mean error of 14° for azimuth and 11.4° for tilt, when using the 15th percentile of data: tested on CAMS irradiance data and UIA PV installation. The IQR range of the calculated tilt and angle was different than in previous studies [13] where the best fit was found using a small amount (0,5 to 4th percentile) of the best-fit RMSE values to generate the monthly result. The best fit found in this study was in the 6th to 15th percentile range. This deviance might be the latitude difference causing the sun to be lower in the sky and fewer daylight hours available. The removal of days when the sun reaches a zenith angle above 70° was used in both cases; this will somewhat reduce the impact of the difference in latitude. This, however, will lead to more data being removed during the winter months at higher latitudes and might be a reason why this study had to use a higher percentile of data in the calculation process.

Shading was a source of inaccuracy when calculating the tilt and azimuth. For 40 of the 120 modules installed on UIA, a tilt or azimuth with a greater than 20° error was calculated. These 40 modules are located in places with known shadows due to the railing and pyranometer instrumentation. The shading of some panels was limited to only months with a high solar zenith angle as the shadow stretched further. This was, however, enough to make the result less accurate. Indicating how sensitive the method is to shade. The azimuth is affected most by shade, as eight of the modules with above 50° azimuth error had a tilt error below 10°. One possible explanation for this could be that the impact of power output is more sensitive to alteration in tilt than azimuth when the panel tilt is low($\leq 15°$) [3, p. 38-41]. However, the result was satisfactory even with the shaded panels included; the median error was 12.2° for tilt and 14.1° for azimuth, Q3 values were 15.5° and 24.1° respectively when utilizing the 15th percentile of the data.

Selecting the day with the least probability of clouds might also be a source of error at higher latitudes with more snow. The daily diffuse fraction calculation only considers the DHI and GHI values, not snow-cover. As only one day is selected to represent each month, snow-covered panels will significantly lessen the accuracy of the curve mishmash. Further improvements in this field could significantly improve the model accuracy during winter. Limiting the model to only summer months is also a possibility. This study has yet to be done due to limited PV system configurations to test it on and the chance of the result changing at different configurations of tilt and azimuth. Limiting the data to only utilize summer months could also offer performance when it comes to shading, as the solar zenith angle is substantially less during the summer months.

The choice of utilizing the top 15th percentile of best matches from the curve fitting for each month was selected based on the result in table 6.6, where the irradiance data from CAMS was utilized. The exact value of 15th was selected based on the Q3 value for the azimuth being $\approx 9°$ lower than the 10th percentile, as well as the other metrics having less of a change ($\approx \pm 2 - 3°$ between the 10th and 15th percentile). It is important to note that this was not the most optimal choice regarding tilt but that a trade-off has been made between optimal tilt, angle, mean, median, Q1, Q3, and max values.

7.4 Filtering and Clustering Method for Performance Analysis

The polynomial filtering technique has challenges; however, some are tied to the dataset used. First, there needs to be more data points in the dataset used. This is due to two reasons; 1) the $Y_f - Yr$ has a concentration of data points at lower values for both axes, and the number of data points diminishes as the axis values get larger. This creates a problem with the binning of the Y_r axis. The best solution was to include the same amount of data points in each bin instead of binning based on the Y_r value. This did, however not entirely fix the other problem. 2) The polynomial fit is only appropriately selected if there is a precise distribution (Normal, Weibull, etc.) with a clear point where outliers start. The given dataset did not have that, mainly due to a lack of data points in the upper region. A possible quick improvement for this would have been to execute the polynomial fit on 5-min data instead of the 1-hour data. However, the 1-hour data format was chosen based on the more accurate irradiance data at this timescale, and time constraints made it difficult to revert to 15-min data. Several ways to improve the selection of minimum points have been tried, including locating the knee point and selecting the lowest point on the curve; however, neither gave significant improvements. Finding the minima with the largest drop closest to the maxima was deemed the best. Due to these reasons, this method is only recommended if the data contains multiple years of data, the timestep is lowered, or both.

Other challenges include the degree of polynomial fit for finding the maxima and minima points (Figure 4.2). Lower degrees have a higher chance of getting the minimum point closest to the maxima to be the border between inliers and outliers; however, a higher polynomial degree will fit the data better. Therefore the best fit was in the range of 2-10th degree and was chosen based on the lowest RMSE. The polynomial degree is also challenging when fitting the left/right polyline (Figure 4.3). A compromise was chosen between fitting a high polynomial that follows the minima points found in (Figure 4.2) and a low polynomial that filters out any rapid and possible inaccurate choices.

Given the size of the utilized dataset, the RANSAC regression is a better filtering method, as it could find a plausible regressional line for nearly all datasets. The chosen parameters were semi-automatically based on a range that worked. RANSAC has found the optimal parameters from the list given, this list could be widened, and the only downside would be increased computational times. However, the chosen parameters remained the same when done so.

7.5 Performance Analysis

The PR across all installations was determined to be 0.79 (all-data), 0.83 (RANSAC inliers), and 0.86 (Polynomial inliers), all of which is inline with the findings of the literature review. The resulting values might, however, be skewed as the number of PV installations is unequal for different parts of the country. The numbers will therefore represent the average on the East coast and Østlandet the most, as the majority of PV innovations are located here. The PR result across all installations when all data and RANSAC inliers were used also indicates that few shading, non-optimized operation, and other problems are present in the dataset, as a relatively small increase in PR was seen after the RANSAC filtering.

It should be noted that further analysis has yet to be conducted to determine whether the differences observed between the use of all data and RANSAC inliers were due to shading removal, irradiance measurement inaccuracies, or a slight inaccuracy in the RANSAC regression line.

Each county's PR and specific yield represents the respective location's PR and specific yield more accurately than the result from all PV installations. However, the limited amount of PV installation in each county can lead to some uncertainties, especially those with fewer installations, as one or multiple faulty values have a more significant impact when a low amount of PV installations is present. In addition, as PR is adjusted for the irradiation regarding tilt and azimuth, inaccurate results from the inference of tilt and azimuth may also supplement inaccuracies.

The use of a fitted Weibull curve to estimate a value for the PR and specific yield has been used in previous studies [83]. However, the limitation regarding the need for more available PV installations to get a good fit is a concern when using a low amount of installations; other metrics, such as mean or median, might be better in such cases. Furthermore, including mean and median values makes comparisons across studies more feasible.

7.6 Statistical Analysis

Two methods have been utilized for the statistical probability test, depending on whether the data is normally distributed. PR has been estimated not to be normally distributed. This decision is based upon previous findings [23], [83], using nonparametric tests have therefore been utilized. Using parametric tests for the specific yield was based upon findings that it should be normally distributed. Parametric tests are seen as more potent than nonparametric tests and are therefore preferred if the conditions are met. However, one could argue that nonparametric tests are better when the sample size is limited and should therefore be used. What states a limited number was, however, hard to find. Parametric tests have also been previously used on PR [83], which is not expected to be normally distributed, so arguments can be made for both choices of parametric and nonparametric tests.

Chapter 8

Conclusions

In conclusion, this study has highlighted the challenges of analyzing a large number of PV systems with limited metadata, primarily attributed to needing more information such as orientation, temperature, and unknown measurement units. However, despite these challenges, it is possible to infer these and conduct an informative analysis that generates a knowledge foundation about the PV installations in Norway

Metadata is supplemented with irradiance measurements, as well as tilt and azimuth inferred only utilizing power and satellite irradiance data. The method is tested on a known installation with 120 PV modules with module-level monitoring (Tigo optimizers). The result shows a median accuracy of 12.2° and 14.1° for tilt and azimuth, respectively. Shading is found to be the most impactful metric for accuracy when inferring tilt and azimuth. Unknown units of measurement are inferred by utilizing the method of locating highly plausible units. W_p, kW_p, and MW_p are likely installed capacity units; thus, every PV installation that does not give highly probable specific yields using these units is disregarded.

Variations in the specific yield are found across regions in Norway, with Østfold, Vestfold, and Oslo recording an estimated specific yield of over 900 kWh/kW_p, and Rogaland, Akershus, Hedemark, Buskerud, Telemark, and Oppland generating over 800 kWh/kW_p. PR is found using three datasets utilizing different filtering procedures. The first is no filtration, giving a PR of 0.79 for all installations. The second is a linear filtration process (RANSAC), finding a PR of 0.83. Finally, a non-linear filtration process accounted for the near-linear relationship between power and irradiance, giving a PR of 0.86 for all installations. Albit limitations regarding the number and distribution of PV installations, together with the limited period available, created some uncertainties regarding the validity of these results, highlighting the need for data collection.

Chapter 9

Further work

Further work includes more suitable methods where the current ones do not work optimally and optimization of the suited ones. Regarding the inference of tilt and azimuth, more data to find the optimal percentile parameter for the northern climate would be a great inclusion. The data could either be from existing PV facilities or developing a method to use simulated facilities. As shading is a major problem that greatly reduces the accuracy of the utilized method, incorporating a procedure to detect shading in the dataset would increase the accuracy of the inferred tilt and azimuth.

A new method could be to utilize the method developed in [34]. The authors have developed a procedure to get the typical power output of an optimal day for each month. This optimal day power curve might also be an interesting topic to look further into, as this could be utilized instead of the daily diffuse fraction method utilized in this report. The benefit of implementing such a procedure could be to remove short-lasting power dips that might not be reflected in the satellite irradiance data. It would also ensure that the power curve does not include downtime. However, accurate orientation information could be lost due to the irradiance also being altered to typical daily profiles. The presence of local shading possibly not being removed is also a concern.

Filtering the PR and Specific yield based on geographical regions is also a valuable inclusion; methods that could be considered are, for example [86], where they have first filtered on a national level, then on a more local level.

The process of selecting the day with the least probability of clouds might also be a source of error at higher latitudes with more snow. The daily diffuse fraction only considers the DHI and GHI values, not snow-cover. As only one day is selected to represent each month, a simple improvement could be to analyze the power curve of the clearest-sky day and detect if it is beneath a threshold. The threshold could, for example, be around the 90th percentile power production during that month, as this is near clear-sky performance [34]. By implementing such a technique, days with snow cover and low cloud formation can be detected, and a day with less snow cover can be selected for that month. This would also enable days with downtime to not be used.

Appendix A

Modules where Shading Affected the Inference of Tilt and Azimuth

Table A.1: Modules where the predicted azimuth or tilt error was above 20 degrees. Irradiance measurements from CAMS and 15th percentile

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	83.200	10.000	A1.csv	123.219	67.078	40.019	57.078
15.000	83.200	10.000	A2.csv	85.352	37.524	2.152	27.524
15.000	83.200	10.000	A3.csv	77.966	33.359	5.234	23.359
15.000	83.200	10.000	A4.csv	80.428	34.772	2.772	24.772
15.000	83.200	10.000	A5.csv	126.278	43.630	43.078	33.630
15.000	263.200	10.000	B1.csv	206.911	17.454	56.289	7.454
15.000	263.200	10.000	B2.csv	209.863	17.695	53.337	7.695
15.000	263.200	10.000	B3.csv	169.310	19.650	93.890	9.650
15.000	263.200	10.000	B4.csv	184.662	28.676	78.538	18.676
15.000	263.200	10.000	B5.csv	214.700	20.851	48.500	10.851
15.000	83.200	10.000	C5.csv	19.833	27.667	63.367	17.667
15.000	83.200	10.000	C6.csv	165.321	22.226	82.121	12.226
15.000	263.200	10.000	D5.csv	242.866	9.677	20.334	0.323
15.000	263.200	10.000	D6.csv	330.136	24.818	66.936	14.818
15.000	83.200	10.000	E1.csv	104.000	27.017	20.800	17.017
15.000	83.200	10.000	E2.csv	39.610	23.184	43.590	13.184
15.000	263.200	10.000	F1.csv	326.778	12.395	63.578	2.395
15.000	263.200	10.000	F5.csv	226.075	33.423	37.125	23.423
15.000	263.200	10.000	F6.csv	226.242	33.512	36.958	23.512
15.000	263.200	10.000	F7.csv	226.757	33.243	36.443	23.243
15.000	83.200	10.000	K1.csv	79.139	34.784	4.061	24.784
15.000	83.200	10.000	K2.csv	77.930	34.102	5.270	24.102
15.000	83.200	10.000	K3.csv	79.653	34.530	3.547	24.530
15.000	83.200	10.000	K4.csv	79.471	34.783	3.729	24.783
15.000	83.200	10.000	K5.csv	75.038	30.551	8.162	20.551
15.000	263.200	10.000	S1.csv	196.271	14.745	66.929	4.745
15.000	263.200	10.000	S2.csv	199.284	14.191	63.916	4.191
15.000	263.200	10.000	S3.csv	199.139	14.066	64.061	4.066
15.000	263.200	10.000	S4.csv	198.533	15.099	64.667	5.099
15.000	263.200	10.000	S5.csv	222.922	14.626	40.278	4.626
15.000	263.200	10.000	V1.csv	229.321	30.074	33.879	20.074
15.000	263.200	10.000	V2.csv	223.311	33.834	39.889	23.834

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	263.200	10.000	V3.csv	225.411	32.982	37.789	22.982
15.000	263.200	10.000	V4.csv	225.350	33.404	37.850	23.404
15.000	263.200	10.000	V5.csv	225.258	33.404	37.942	23.404
15.000	263.200	10.000	W1.csv	191.209	28.528	71.991	18.528
15.000	263.200	10.000	W2.csv	190.936	29.679	72.264	19.679
15.000	263.200	10.000	W3.csv	192.409	25.888	70.791	15.888
15.000	263.200	10.000	W4.csv	191.501	27.543	71.699	17.543
15.000	263.200	10.000	W5.csv	193.135	24.122	70.065	14.122

Appendix B

PR Values of the PV Installations by County

Table B.1: PR values of PV installations by county: all data filtered using Tukey's method and values Above 1 Excluded. The PR value is chosen by the peak of a Weibull curve fitted to a histogram of all PR Values

Fylke	Num PV Installations	PR Max Peak	Q1	Mean	Median	Q3	Std
Rogaland	79	0.83	0.72	0.80	0.81	0.89	0.12
Østfold	75	0.77	0.66	0.75	0.75	0.84	0.13
Akershus	49	0.74	0.62	0.72	0.73	0.80	0.14
Hordaland	44	0.83	0.71	0.77	0.82	0.87	0.15
Hedmark	34	0.78	0.70	0.76	0.77	0.83	0.11
Buskerud	28	0.78	0.64	0.72	0.74	0.80	0.15
Vestfold	20	0.79	0.63	0.75	0.74	0.85	0.14
Sør-Trøndelag	14	0.75	0.67	0.72	0.74	0.79	0.13
Telemark	13	0.72	0.57	0.68	0.71	0.82	0.14
Oppland	10	0.76	0.58	0.70	0.75	0.81	0.13
Oslo	7	0.76	0.69	0.71	0.71	0.77	0.12

Table B.2: PR values of PV installations by county: RANSAC inliers derived from Tukey's method filtered data and values above 1 excluded. The PR value is chosen by the peak of a Weibull curve fitted to a histogram of all PR Values

County	Num PV Installations	PR Max Peak	Q1	Mean	Median	Q3	Std
Rogaland	86	0.87	0.76	0.83	0.83	0.92	0.12
Østfold	70	0.80	0.72	0.77	0.78	0.85	0.11
Akershus	45	0.78	0.69	0.76	0.78	0.82	0.10
Hordaland	44	0.86	0.74	0.82	0.84	0.89	0.10
Hedmark	33	0.83	0.76	0.80	0.82	0.84	0.10
Buskerud	24	0.84	0.77	0.82	0.83	0.88	0.09
Vestfold	22	0.82	0.72	0.79	0.78	0.88	0.12
Sør-Trøndelag	13	0.75	0.75	0.80	0.78	0.84	0.07
Telemark	14	0.82	0.68	0.78	0.82	0.87	0.13
Oppland	10	0.80	0.65	0.74	0.78	0.84	0.14
Oslo	8	0.79	0.70	0.76	0.74	0.85	0.14

Table B.3: PR values of PV installations by county: polynomial inliers derived from Tukey's method filtered data and values above 1 excluded. The PR value is chosen by the peak of a Weibull curve fitted to a histogram of all PR Values

County	Num PV Installations	PR Max Peak	Q1	Mean	Median	Q3	Std
Rogaland	82	0.89	0.79	0.84	0.87	0.92	0.11
Østfold	70	0.81	0.75	0.80	0.79	0.88	0.10
Akershus	45	0.81	0.75	0.79	0.81	0.85	0.10
Hordaland	42	0.88	0.78	0.84	0.84	0.91	0.11
Hedmark	33	0.85	0.78	0.83	0.83	0.89	0.08
Buskerud	26	0.90	0.82	0.87	0.88	0.93	0.08
Vestfold	22	0.86	0.77	0.83	0.84	0.89	0.11
Sør-Trøndelag	13	0.75	0.78	0.85	0.82	0.91	0.08
Telemark	14	0.87	0.74	0.82	0.85	0.89	0.11
Oppland	10	0.83	0.69	0.77	0.80	0.88	0.14
Oslo	7	0.84	0.75	0.79	0.79	0.87	0.11

Appendix C

Specific Yield of the PV Installations by County

Table C.1: Specific yield of PV installations by county: All data filtered using Tukey's method. The specific yield value is chosen by the peak of a Weibull curve fitted to a histogram of all specific yield Values

County	Num PV Installations	Max Peak	Q1	Mean	Median	Q3	STD
Rogaland	96	864.84	738.46	826.29	843.10	934.40	146.55
Østfold	81	949.12	794.89	915.09	939.52	1034.88	174.05
Akershus	54	876.98	745.67	813.63	848.97	925.24	183.91
Hordaland	49	788.57	687.29	782.83	767.96	896.60	153.23
Hedmark	38	849.70	748.08	859.21	872.31	942.07	147.89
Buskerud	28	851.52	731.97	829.98	838.88	940.03	160.40
Vestfold	21	955.61	815.02	935.22	912.60	1082.60	171.22
Sør-Trøndelag	15	682.61	630.37	661.59	677.37	714.96	119.72
Telemark	14	839.23	718.39	847.44	808.88	985.27	174.60
Oppland	10	818.08	691.34	760.30	824.46	858.82	132.63
Oslo	9	972.46	831.57	884.07	859.63	1054.04	190.82

Appendix D

Estimated Tilt and Azimuth: UIA: Local

Table D.1: Estimated orientation: UIA: Local

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	83.200	10.000	A1.csv	26.000	36.000	57.200	26.000
15.000	83.200	10.000	A2.csv	76.019	34.269	7.181	24.269
15.000	83.200	10.000	A3.csv	78.868	33.078	4.332	23.078
15.000	83.200	10.000	A4.csv	78.653	32.622	4.547	22.622
15.000	83.200	10.000	A5.csv	131.669	42.607	48.469	32.607
15.000	83.200	10.000	A6.csv	58.147	23.104	25.053	13.104
15.000	263.200	10.000	B1.csv	182.414	49.103	80.786	39.103
15.000	263.200	10.000	B2.csv	182.319	52.986	80.881	42.986
15.000	263.200	10.000	B3.csv	166.990	18.729	96.210	8.729
15.000	263.200	10.000	B4.csv	162.415	24.623	100.785	14.623
15.000	263.200	10.000	B5.csv	229.997	17.280	33.203	7.280
15.000	263.200	10.000	B6.csv	182.604	46.047	80.596	36.047
15.000	83.200	10.000	C1.csv	95.098	22.258	11.898	12.258
15.000	83.200	10.000	C2.csv	95.118	22.189	11.918	12.189
15.000	83.200	10.000	C3.csv	95.378	22.953	12.178	12.953
15.000	83.200	10.000	C4.csv	94.624	22.647	11.424	12.647
15.000	83.200	10.000	C5.csv	139.590	29.466	56.390	19.466
15.000	83.200	10.000	C6.csv	9.531	38.094	73.669	28.094
15.000	263.200	10.000	D1.csv	250.805	12.886	12.395	2.886
15.000	263.200	10.000	D2.csv	249.980	12.884	13.220	2.884
15.000	263.200	10.000	D3.csv	242.292	9.930	20.908	0.070
15.000	263.200	10.000	D4.csv	252.506	11.104	10.694	1.104
15.000	263.200	10.000	D5.csv	253.082	11.560	10.118	1.560
15.000	263.200	10.000	D6.csv	333.500	30.500	70.300	20.500
15.000	83.200	10.000	E1.csv	106.982	28.747	23.782	18.747
15.000	83.200	10.000	E2.csv	37.165	22.477	46.035	12.477
15.000	83.200	10.000	E3.csv	92.219	24.344	9.019	14.344
15.000	83.200	10.000	E4.csv	91.999	24.205	8.799	14.205
15.000	83.200	10.000	E5.csv	108.835	25.196	25.635	15.196
15.000	83.200	10.000	E6.csv	88.348	20.273	5.148	10.273
15.000	83.200	10.000	E7.csv	88.996	20.253	5.796	10.253
15.000	83.200	10.000	E8.csv	89.859	20.674	6.659	10.674
15.000	263.200	10.000	F1.csv	254.510	12.197	8.690	2.197
15.000	263.200	10.000	F2.csv	277.003	11.988	13.803	1.988
15.000	263.200	10.000	F3.csv	260.368	12.138	2.832	2.138
15.000	263.200	10.000	F4.csv	259.184	12.330	4.016	2.330

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	263.200	10.000	F5.csv	220.393	37.660	42.807	27.660
15.000	263.200	10.000	F6.csv	220.124	37.925	43.076	27.925
15.000	263.200	10.000	F7.csv	220.519	37.663	42.681	27.663
15.000	263.200	10.000	F8.csv	219.405	38.182	43.795	28.182
15.000	83.200	10.000	G1.csv	98.141	22.176	14.941	12.176
15.000	83.200	10.000	G2.csv	96.382	22.204	13.182	12.204
15.000	83.200	10.000	G3.csv	97.950	21.955	14.750	11.955
15.000	83.200	10.000	G4.csv	96.682	22.042	13.482	12.042
15.000	83.200	10.000	G5.csv	96.874	22.166	13.674	12.166
15.000	83.200	10.000	H1.csv	96.932	22.585	13.732	12.585
15.000	83.200	10.000	H2.csv	96.178	21.995	12.978	11.995
15.000	83.200	10.000	H3.csv	96.379	21.678	13.179	11.678
15.000	83.200	10.000	H4.csv	97.096	21.758	13.896	11.758
15.000	83.200	10.000	H5.csv	96.733	22.443	13.533	12.443
15.000	83.200	10.000	I1.csv	95.636	24.713	12.436	14.713
15.000	83.200	10.000	I2.csv	96.688	26.289	13.488	16.289
15.000	83.200	10.000	I3.csv	95.491	25.049	12.291	15.049
15.000	83.200	10.000	I4.csv	95.850	24.663	12.650	14.663
15.000	83.200	10.000	I5.csv	96.212	24.445	13.012	14.445
15.000	83.200	10.000	J1.csv	92.034	19.609	8.834	9.609
15.000	83.200	10.000	J2.csv	99.254	21.023	16.054	11.023
15.000	83.200	10.000	J3.csv	96.592	21.149	13.392	11.149
15.000	83.200	10.000	J4.csv	94.029	20.712	10.829	10.712
15.000	83.200	10.000	J5.csv	95.792	20.785	12.592	10.785
15.000	83.200	10.000	K1.csv	79.487	34.459	3.713	24.459
15.000	83.200	10.000	K2.csv	79.156	34.448	4.044	24.448
15.000	83.200	10.000	K3.csv	81.444	35.184	1.756	25.184
15.000	83.200	10.000	K4.csv	80.378	34.731	2.822	24.731
15.000	83.200	10.000	K5.csv	78.560	32.320	4.640	22.320
15.000	83.200	10.000	L1.csv	95.743	23.074	12.543	13.074
15.000	83.200	10.000	L2.csv	95.130	22.569	11.930	12.569
15.000	83.200	10.000	L3.csv	95.511	22.765	12.311	12.765
15.000	83.200	10.000	L4.csv	93.271	23.462	10.071	13.462
15.000	83.200	10.000	L5.csv	96.055	23.867	12.855	13.867
15.000	83.200	10.000	M1.csv	95.348	23.697	12.148	13.697
15.000	83.200	10.000	M2.csv	95.591	23.568	12.391	13.568
15.000	83.200	10.000	M4.csv	93.541	25.304	10.341	15.304
15.000	83.200	10.000	M5.csv	92.588	25.087	9.388	15.087
15.000	83.200	10.000	N1.csv	89.764	21.700	6.564	11.700
15.000	83.200	10.000	N2.csv	88.426	21.632	5.226	11.632
15.000	83.200	10.000	N3.csv	89.495	21.384	6.295	11.384
15.000	83.200	10.000	N4.csv	90.831	21.266	7.631	11.266
15.000	83.200	10.000	N5.csv	91.171	21.849	7.971	11.849
15.000	263.200	10.000	S1.csv	184.486	5.156	78.714	4.844
15.000	263.200	10.000	S2.csv	182.271	6.712	80.929	3.288
15.000	263.200	10.000	S3.csv	185.045	6.197	78.155	3.803
15.000	263.200	10.000	S4.csv	180.470	2.625	82.730	7.375
15.000	263.200	10.000	S5.csv	205.853	10.882	57.347	0.882
15.000	263.200	10.000	T1.csv	247.847	12.698	15.353	2.698
15.000	263.200	10.000	T2.csv	247.890	12.803	15.310	2.803
15.000	263.200	10.000	T3.csv	247.441	12.938	15.759	2.938

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	263.200	10.000	T4.csv	239.873	10.129	23.327	0.129
15.000	263.200	10.000	T5.csv	240.217	9.969	22.983	0.031
15.000	263.200	10.000	U1.csv	241.757	9.670	21.443	0.330
15.000	263.200	10.000	U2.csv	241.556	9.708	21.644	0.292
15.000	263.200	10.000	U3.csv	240.954	9.500	22.246	0.500
15.000	263.200	10.000	U4.csv	241.583	9.361	21.617	0.639
15.000	263.200	10.000	U5.csv	241.358	9.453	21.842	0.547
15.000	263.200	10.000	V1.csv	223.358	35.123	39.842	25.123
15.000	263.200	10.000	V2.csv	219.945	37.522	43.255	27.522
15.000	263.200	10.000	V3.csv	221.075	37.046	42.125	27.046
15.000	263.200	10.000	V4.csv	219.820	37.578	43.380	27.578
15.000	263.200	10.000	V5.csv	219.864	37.736	43.336	27.736
15.000	263.200	10.000	W1.csv	182.061	18.391	81.139	8.391
15.000	263.200	10.000	W2.csv	182.788	19.130	80.412	9.130
15.000	263.200	10.000	W3.csv	183.041	23.398	80.159	13.398
15.000	263.200	10.000	W4.csv	183.827	17.645	79.373	7.645
15.000	263.200	10.000	W5.csv	181.042	12.966	82.158	2.966
15.000	263.200	10.000	X1.csv	246.058	13.140	17.142	3.140
15.000	263.200	10.000	X2.csv	247.797	12.822	15.403	2.822
15.000	263.200	10.000	X3.csv	246.647	12.768	16.553	2.768
15.000	263.200	10.000	X4.csv	242.866	14.403	20.334	4.403
15.000	263.200	10.000	X5.csv	247.207	12.737	15.993	2.737
15.000	263.200	10.000	Y1.csv	239.208	10.088	23.992	0.088
15.000	263.200	10.000	Y2.csv	240.670	9.878	22.530	0.122
15.000	263.200	10.000	Y3.csv	240.157	9.772	23.043	0.228
15.000	263.200	10.000	Y4.csv	239.660	10.056	23.540	0.056
15.000	263.200	10.000	Y5.csv	240.157	10.137	23.043	0.137
15.000	263.200	10.000	Z1.csv	239.376	9.655	23.824	0.345
15.000	263.200	10.000	Z2.csv	240.090	9.554	23.110	0.446
15.000	263.200	10.000	Z3.csv	238.972	9.749	24.228	0.251
15.000	263.200	10.000	Z4.csv	239.595	9.489	23.605	0.511
15.000	263.200	10.000	Z5.csv	240.800	9.572	22.400	0.428

Appendix E

Estimated Tilt and Azimuth: UIA: CAMS

Table E.1: Estimated orientation: UIA: CAMS

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	83.200	10.000	A1.csv	123.219	67.078	40.019	57.078
15.000	83.200	10.000	A2.csv	85.352	37.524	2.152	27.524
15.000	83.200	10.000	A3.csv	77.966	33.359	5.234	23.359
15.000	83.200	10.000	A4.csv	80.428	34.772	2.772	24.772
15.000	83.200	10.000	A5.csv	126.278	43.630	43.078	33.630
15.000	83.200	10.000	A6.csv	64.740	25.407	18.460	15.407
15.000	263.200	10.000	B1.csv	206.911	17.454	56.289	7.454
15.000	263.200	10.000	B2.csv	209.863	17.695	53.337	7.695
15.000	263.200	10.000	B3.csv	169.310	19.650	93.890	9.650
15.000	263.200	10.000	B4.csv	184.662	28.676	78.538	18.676
15.000	263.200	10.000	B5.csv	214.700	20.851	48.500	10.851
15.000	263.200	10.000	B6.csv	264.535	3.219	1.335	6.781
15.000	83.200	10.000	C1.csv	97.051	22.801	13.851	12.801
15.000	83.200	10.000	C2.csv	97.233	22.850	14.033	12.850
15.000	83.200	10.000	C3.csv	97.038	23.424	13.838	13.424
15.000	83.200	10.000	C4.csv	96.323	23.053	13.123	13.053
15.000	83.200	10.000	C5.csv	19.833	27.667	63.367	17.667
15.000	83.200	10.000	C6.csv	165.321	22.226	82.121	12.226
15.000	263.200	10.000	D1.csv	270.405	11.837	7.205	1.837
15.000	263.200	10.000	D2.csv	271.342	11.690	8.142	1.690
15.000	263.200	10.000	D3.csv	276.849	10.538	13.649	0.538
15.000	263.200	10.000	D4.csv	277.086	10.763	13.886	0.763
15.000	263.200	10.000	D5.csv	242.866	9.677	20.334	0.323
15.000	263.200	10.000	D6.csv	330.136	24.818	66.936	14.818
15.000	83.200	10.000	E1.csv	104.000	27.017	20.800	17.017
15.000	83.200	10.000	E2.csv	39.610	23.184	43.590	13.184
15.000	83.200	10.000	E3.csv	94.716	25.294	11.516	15.294
15.000	83.200	10.000	E4.csv	95.095	25.374	11.895	15.374
15.000	83.200	10.000	E5.csv	102.490	23.597	19.290	13.597
15.000	83.200	10.000	E6.csv	89.670	20.607	6.470	10.607
15.000	83.200	10.000	E7.csv	90.549	20.640	7.349	10.640
15.000	83.200	10.000	E8.csv	91.625	21.177	8.425	11.177
15.000	263.200	10.000	F1.csv	326.778	12.395	63.578	2.395
15.000	263.200	10.000	F2.csv	252.574	11.421	10.626	1.421

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	263.200	10.000	F3.csv	273.031	10.821	9.831	0.821
15.000	263.200	10.000	F4.csv	272.365	10.851	9.165	0.851
15.000	263.200	10.000	F5.csv	226.075	33.423	37.125	23.423
15.000	263.200	10.000	F6.csv	226.242	33.512	36.958	23.512
15.000	263.200	10.000	F7.csv	226.757	33.243	36.443	23.243
15.000	263.200	10.000	F8.csv	245.858	21.771	17.342	11.771
15.000	83.200	10.000	G1.csv	99.536	22.690	16.336	12.690
15.000	83.200	10.000	G2.csv	98.153	22.607	14.953	12.607
15.000	83.200	10.000	G3.csv	99.355	22.385	16.155	12.385
15.000	83.200	10.000	G4.csv	97.233	22.002	14.033	12.002
15.000	83.200	10.000	G5.csv	98.196	22.537	14.996	12.537
15.000	83.200	10.000	H1.csv	99.364	23.359	16.164	13.359
15.000	83.200	10.000	H2.csv	97.224	22.047	14.024	12.047
15.000	83.200	10.000	H3.csv	97.391	21.989	14.191	11.989
15.000	83.200	10.000	H4.csv	98.051	22.031	14.851	12.031
15.000	83.200	10.000	H5.csv	98.471	22.877	15.271	12.877
15.000	83.200	10.000	I1.csv	97.506	25.221	14.306	15.221
15.000	83.200	10.000	I2.csv	98.315	26.935	15.115	16.935
15.000	83.200	10.000	I3.csv	97.377	25.682	14.177	15.682
15.000	83.200	10.000	I4.csv	97.288	24.844	14.088	14.844
15.000	83.200	10.000	I5.csv	98.011	24.922	14.811	14.922
15.000	83.200	10.000	J1.csv	93.609	20.096	10.409	10.096
15.000	83.200	10.000	J2.csv	100.857	22.193	17.657	12.193
15.000	83.200	10.000	J3.csv	96.895	21.204	13.695	11.204
15.000	83.200	10.000	J4.csv	95.476	21.269	12.276	11.269
15.000	83.200	10.000	J5.csv	96.985	21.289	13.785	11.289
15.000	83.200	10.000	K1.csv	79.139	34.784	4.061	24.784
15.000	83.200	10.000	K2.csv	77.930	34.102	5.270	24.102
15.000	83.200	10.000	K3.csv	79.653	34.530	3.547	24.530
15.000	83.200	10.000	K4.csv	79.471	34.783	3.729	24.783
15.000	83.200	10.000	K5.csv	75.038	30.551	8.162	20.551
15.000	83.200	10.000	L1.csv	97.476	23.473	14.276	13.473
15.000	83.200	10.000	L2.csv	97.249	23.175	14.049	13.175
15.000	83.200	10.000	L3.csv	97.870	23.489	14.670	13.489
15.000	83.200	10.000	L4.csv	95.578	24.270	12.378	14.270
15.000	83.200	10.000	L5.csv	98.259	24.642	15.059	14.642
15.000	83.200	10.000	M1.csv	97.442	24.308	14.242	14.308
15.000	83.200	10.000	M2.csv	97.405	23.980	14.205	13.980
15.000	83.200	10.000	M3.csv	95.266	25.569	12.066	15.569
15.000	83.200	10.000	M4.csv	95.583	25.824	12.383	15.824
15.000	83.200	10.000	M5.csv	95.190	25.886	11.990	15.886
15.000	83.200	10.000	N1.csv	93.074	22.887	9.874	12.887
15.000	83.200	10.000	N2.csv	91.580	22.636	8.380	12.636
15.000	83.200	10.000	N3.csv	91.373	21.939	8.173	11.939
15.000	83.200	10.000	N4.csv	92.544	21.766	9.344	11.766
15.000	83.200	10.000	N5.csv	91.793	21.693	8.593	11.693
15.000	263.200	10.000	S1.csv	196.271	14.745	66.929	4.745
15.000	263.200	10.000	S2.csv	199.284	14.191	63.916	4.191
15.000	263.200	10.000	S3.csv	199.139	14.066	64.061	4.066
15.000	263.200	10.000	S4.csv	198.533	15.099	64.667	5.099
15.000	263.200	10.000	S5.csv	222.922	14.626	40.278	4.626

Percentile	True azimuth	True tilt	Inst. No	Azimuth	Tilt	Azimuth error	Tilt error
15.000	263.200	10.000	T1.csv	272.201	11.626	9.001	1.626
15.000	263.200	10.000	T2.csv	271.652	11.718	8.452	1.718
15.000	263.200	10.000	T3.csv	270.521	11.692	7.321	1.692
15.000	263.200	10.000	T4.csv	275.030	10.310	11.830	0.310
15.000	263.200	10.000	T5.csv	278.172	10.383	14.972	0.383
15.000	263.200	10.000	U1.csv	279.303	10.504	16.103	0.504
15.000	263.200	10.000	U2.csv	277.763	10.700	14.563	0.700
15.000	263.200	10.000	U3.csv	278.576	10.178	15.376	0.178
15.000	263.200	10.000	U4.csv	280.737	10.233	17.537	0.233
15.000	263.200	10.000	U5.csv	279.105	10.170	15.905	0.170
15.000	263.200	10.000	V1.csv	229.321	30.074	33.879	20.074
15.000	263.200	10.000	V2.csv	223.311	33.834	39.889	23.834
15.000	263.200	10.000	V3.csv	225.411	32.982	37.789	22.982
15.000	263.200	10.000	V4.csv	225.350	33.404	37.850	23.404
15.000	263.200	10.000	V5.csv	225.258	33.404	37.942	23.404
15.000	263.200	10.000	W1.csv	191.209	28.528	71.991	18.528
15.000	263.200	10.000	W2.csv	190.936	29.679	72.264	19.679
15.000	263.200	10.000	W3.csv	192.409	25.888	70.791	15.888
15.000	263.200	10.000	W4.csv	191.501	27.543	71.699	17.543
15.000	263.200	10.000	W5.csv	193.135	24.122	70.065	14.122
15.000	263.200	10.000	X1.csv	265.971	12.231	2.771	2.231
15.000	263.200	10.000	X2.csv	266.597	12.222	3.397	2.222
15.000	263.200	10.000	X3.csv	267.195	11.856	3.995	1.856
15.000	263.200	10.000	X4.csv	259.633	12.719	3.567	2.719
15.000	263.200	10.000	X5.csv	267.890	11.992	4.690	1.992
15.000	263.200	10.000	Y1.csv	269.019	10.621	5.819	0.621
15.000	263.200	10.000	Y2.csv	270.543	10.583	7.343	0.583
15.000	263.200	10.000	Y3.csv	270.973	10.505	7.773	0.505
15.000	263.200	10.000	Y4.csv	269.308	10.782	6.108	0.782
15.000	263.200	10.000	Y5.csv	269.009	10.694	5.809	0.694
15.000	263.200	10.000	Z1.csv	271.204	10.312	8.004	0.312
15.000	263.200	10.000	Z2.csv	272.484	10.316	9.284	0.316
15.000	263.200	10.000	Z3.csv	265.204	10.253	2.004	0.253
15.000	263.200	10.000	Z4.csv	270.992	10.003	7.792	0.003
15.000	263.200	10.000	Z5.csv	272.737	10.320	9.537	0.320

Appendix F

PR for Each County and Month: Dataset 1) All data

Table F.1: PR for each county and month. Dataset 1) All data

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Rogaland	Jan	0.19	0.17	0.28	0.44	0.33
Rogaland	Feb	0.69	0.47	0.59	0.76	0.58
Rogaland	Mar	0.81	0.66	0.75	0.87	0.72
Rogaland	Apr	0.86	0.74	0.83	0.90	0.79
Rogaland	May	0.88	0.78	0.84	0.91	0.81
Rogaland	Jun	0.88	0.77	0.86	0.93	0.81
Rogaland	Jul	0.89	0.79	0.88	0.93	0.82
Rogaland	Aug	0.86	0.75	0.83	0.90	0.77
Rogaland	Sep	0.82	0.68	0.78	0.88	0.72
Rogaland	Oct	0.74	0.54	0.68	0.79	0.64
Rogaland	Nov	0.35	0.24	0.38	0.54	0.40
Rogaland	Dec	0.00	0.08	0.19	0.38	0.27
Hordaland	Jan	0.19	0.19	0.31	0.41	0.33
Hordaland	Feb	0.72	0.48	0.70	0.81	0.63
Hordaland	Mar	0.79	0.58	0.67	0.87	0.67
Hordaland	Apr	0.86	0.73	0.85	0.91	0.77
Hordaland	May	0.88	0.77	0.86	0.93	0.80
Hordaland	Jun	0.86	0.74	0.85	0.91	0.78
Hordaland	Jul	0.88	0.78	0.86	0.92	0.82
Hordaland	Aug	0.85	0.71	0.82	0.91	0.78
Hordaland	Sep	0.82	0.64	0.78	0.86	0.73
Hordaland	Oct	0.70	0.52	0.70	0.78	0.66
Hordaland	Nov	0.35	0.26	0.41	0.57	0.43
Hordaland	Dec	0.03	0.11	0.22	0.39	0.28
Akershus	Jan	0.00	0.08	0.18	0.29	0.21
Akershus	Feb	0.29	0.24	0.34	0.52	0.38
Akershus	Mar	0.59	0.22	0.57	0.70	0.50
Akershus	Apr	0.77	0.59	0.76	0.83	0.66
Akershus	May	0.81	0.66	0.80	0.88	0.70
Akershus	Jun	0.78	0.61	0.76	0.84	0.67
Akershus	Jul	0.79	0.62	0.77	0.85	0.68
Akershus	Aug	0.79	0.60	0.79	0.87	0.67

Continued on next page

Table F.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Akershus	Sep	0.76	0.56	0.74	0.83	0.64
Akershus	Oct	0.65	0.42	0.56	0.73	0.52
Akershus	Nov	0.23	0.20	0.25	0.38	0.26
Akershus	Dec	0.00	0.02	0.06	0.12	0.10
Buskerud	Jan	0.00	0.03	0.07	0.14	0.12
Buskerud	Feb	0.43	0.25	0.46	0.58	0.41
Buskerud	Mar	0.73	0.47	0.65	0.86	0.59
Buskerud	Apr	0.82	0.65	0.81	0.88	0.70
Buskerud	May	0.83	0.70	0.83	0.88	0.70
Buskerud	Jun	0.84	0.71	0.80	0.89	0.72
Buskerud	Jul	0.83	0.69	0.83	0.87	0.72
Buskerud	Aug	0.85	0.73	0.81	0.90	0.75
Buskerud	Sep	0.79	0.62	0.75	0.86	0.67
Buskerud	Oct	0.69	0.49	0.63	0.79	0.58
Buskerud	Nov	0.13	0.17	0.30	0.41	0.30
Buskerud	Dec	0.00	0.01	0.04	0.09	0.10
Østfold	Jan	0.19	0.14	0.27	0.37	0.27
Østfold	Feb	0.54	0.31	0.49	0.66	0.47
Østfold	Mar	0.75	0.52	0.69	0.83	0.62
Østfold	Apr	0.81	0.65	0.80	0.86	0.71
Østfold	May	0.81	0.66	0.79	0.87	0.71
Østfold	Jun	0.82	0.69	0.82	0.88	0.72
Østfold	Jul	0.81	0.69	0.81	0.86	0.71
Østfold	Aug	0.79	0.63	0.78	0.85	0.69
Østfold	Sep	0.78	0.61	0.74	0.84	0.67
Østfold	Oct	0.69	0.46	0.62	0.75	0.57
Østfold	Nov	0.30	0.21	0.34	0.45	0.33
Østfold	Dec	0.00	0.06	0.12	0.21	0.15
Hedmark	Jan	0.00	0.07	0.15	0.33	0.21
Hedmark	Feb	0.30	0.23	0.42	0.62	0.43
Hedmark	Mar	0.75	0.26	0.74	0.83	0.61
Hedmark	Apr	0.83	0.73	0.82	0.87	0.75
Hedmark	May	0.85	0.78	0.82	0.86	0.78
Hedmark	Jun	0.87	0.78	0.85	0.90	0.80
Hedmark	Jul	0.85	0.77	0.82	0.87	0.77
Hedmark	Aug	0.84	0.78	0.83	0.87	0.75
Hedmark	Sep	0.82	0.72	0.80	0.84	0.74
Hedmark	Oct	0.70	0.56	0.64	0.70	0.61
Hedmark	Nov	0.24	0.23	0.31	0.40	0.36
Hedmark	Dec	0.00	0.01	0.04	0.08	0.08
Sør-Trøndelag	Jan	0.00	0.02	0.05	0.14	0.08
Sør-Trøndelag	Feb	0.19	0.13	0.22	0.38	0.25
Sør-Trøndelag	Mar	0.78	0.71	0.76	0.81	0.70
Sør-Trøndelag	Apr	0.87	0.74	0.81	0.94	0.79
Sør-Trøndelag	May	0.88	0.79	0.87	0.91	0.80
Sør-Trøndelag	Jun	0.82	0.76	0.82	0.85	0.76
Sør-Trøndelag	Jul	0.80	0.72	0.80	0.83	0.73
Sør-Trøndelag	Aug	0.79	0.67	0.78	0.82	0.71
Sør-Trøndelag	Sep	0.73	0.62	0.69	0.77	0.62

Continued on next page

Table F.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Sør-Trøndelag	Oct	0.51	0.42	0.53	0.59	0.49
Sør-Trøndelag	Nov	0.00	0.08	0.16	0.40	0.25
Sør-Trøndelag	Dec	0.00	0.00	0.00	0.07	0.05
Oslo	Jan	0.00	0.04	0.18	0.31	0.21
Oslo	Feb	0.00	0.11	0.40	0.60	0.38
Oslo	Mar	0.00	0.20	0.22	0.23	0.25
Oslo	Apr	0.66	0.55	0.67	0.71	0.55
Oslo	May	0.76	0.68	0.77	0.81	0.64
Oslo	Jun	0.75	0.65	0.77	0.80	0.63
Oslo	Jul	0.76	0.64	0.75	0.83	0.64
Oslo	Aug	0.79	0.57	0.78	0.90	0.66
Oslo	Sep	0.77	0.47	0.76	0.86	0.62
Oslo	Oct	0.65	0.30	0.62	0.76	0.51
Oslo	Nov	0.19	0.13	0.29	0.44	0.28
Oslo	Dec	0.00	0.02	0.05	0.16	0.09
Vestfold	Jan	0.00	0.14	0.25	0.49	0.30
Vestfold	Feb	0.69	0.44	0.63	0.78	0.56
Vestfold	Mar	0.79	0.60	0.71	0.86	0.68
Vestfold	Apr	0.82	0.67	0.79	0.85	0.72
Vestfold	May	0.81	0.69	0.78	0.87	0.69
Vestfold	Jun	0.81	0.68	0.82	0.88	0.69
Vestfold	Jul	0.81	0.69	0.80	0.86	0.70
Vestfold	Aug	0.81	0.69	0.79	0.85	0.69
Vestfold	Sep	0.79	0.65	0.73	0.84	0.67
Vestfold	Oct	0.69	0.47	0.57	0.77	0.56
Vestfold	Nov	0.33	0.23	0.33	0.48	0.33
Vestfold	Dec	0.00	0.06	0.14	0.31	0.19
Telemark	Jan	0.00	0.00	0.12	0.23	0.12
Telemark	Feb	0.51	0.36	0.47	0.57	0.42
Telemark	Mar	0.75	0.54	0.72	0.80	0.65
Telemark	Apr	0.81	0.69	0.80	0.88	0.77
Telemark	May	0.83	0.72	0.85	0.92	0.82
Telemark	Jun	0.65	0.70	0.77	0.88	0.79
Telemark	Jul	0.83	0.67	0.78	0.89	0.73
Telemark	Aug	0.76	0.57	0.69	0.84	0.66
Telemark	Sep	0.74	0.54	0.70	0.84	0.62
Telemark	Oct	0.61	0.44	0.59	0.63	0.50
Telemark	Nov	0.22	0.22	0.27	0.36	0.27
Telemark	Dec	0.00	0.00	0.05	0.08	0.07
Oppland	Jan	0.00	0.00	0.05	0.10	0.09
Oppland	Feb	0.00	0.06	0.25	0.49	0.32
Oppland	Mar	0.75	0.46	0.73	0.83	0.61
Oppland	Apr	0.83	0.62	0.84	0.91	0.71
Oppland	May	0.77	0.48	0.74	0.87	0.65
Oppland	Jun	0.81	0.62	0.80	0.88	0.70
Oppland	Jul	0.81	0.69	0.81	0.88	0.71
Oppland	Aug	0.83	0.72	0.82	0.89	0.73
Oppland	Sep	0.79	0.62	0.81	0.84	0.69
Oppland	Oct	0.66	0.52	0.59	0.68	0.56

Continued on next page

Table F.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Oppland	Nov	0.26	0.20	0.25	0.31	0.24
Oppland	Dec	0.00	0.00	0.03	0.07	0.04

Appendix G

PR for Each County and Month: Dataset 2) RANSAC data

Table G.1: PR for each county and month. Dataset 2) RANSAC data

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Rogaland	Jan	0.52	0.38	0.48	0.64	0.51
Rogaland	Feb	0.77	0.58	0.70	0.83	0.69
Rogaland	Mar	0.86	0.73	0.80	0.92	0.79
Rogaland	Apr	0.87	0.76	0.83	0.91	0.80
Rogaland	May	0.87	0.77	0.84	0.90	0.80
Rogaland	Jun	0.88	0.78	0.85	0.92	0.81
Rogaland	Jul	0.89	0.79	0.86	0.92	0.82
Rogaland	Aug	0.87	0.76	0.84	0.91	0.80
Rogaland	Sep	0.86	0.74	0.82	0.90	0.79
Rogaland	Oct	0.79	0.64	0.74	0.83	0.71
Rogaland	Nov	0.64	0.46	0.61	0.71	0.60
Rogaland	Dec	0.39	0.27	0.44	0.66	0.47
Hordaland	Jan	0.58	0.47	0.56	0.68	0.57
Hordaland	Feb	0.78	0.62	0.74	0.83	0.72
Hordaland	Mar	0.86	0.69	0.81	0.90	0.79
Hordaland	Apr	0.87	0.74	0.86	0.92	0.81
Hordaland	May	0.88	0.74	0.86	0.92	0.82
Hordaland	Jun	0.88	0.75	0.86	0.92	0.83
Hordaland	Jul	0.89	0.77	0.88	0.92	0.84
Hordaland	Aug	0.87	0.76	0.86	0.89	0.82
Hordaland	Sep	0.85	0.72	0.81	0.89	0.79
Hordaland	Oct	0.77	0.63	0.75	0.81	0.71
Hordaland	Nov	0.66	0.55	0.63	0.75	0.62
Hordaland	Dec	0.54	0.35	0.54	0.70	0.52
Østfold	Jan	0.51	0.36	0.48	0.62	0.48
Østfold	Feb	0.73	0.53	0.68	0.76	0.64
Østfold	Mar	0.79	0.63	0.75	0.84	0.70
Østfold	Apr	0.83	0.70	0.80	0.87	0.75
Østfold	May	0.82	0.72	0.79	0.85	0.74
Østfold	Jun	0.82	0.72	0.80	0.86	0.75
Østfold	Jul	0.82	0.72	0.79	0.86	0.74
Østfold	Aug	0.82	0.73	0.79	0.86	0.75

Continued on next page

Table G.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Østfold	Sep	0.80	0.69	0.76	0.84	0.72
Østfold	Oct	0.76	0.58	0.70	0.78	0.68
Østfold	Nov	0.56	0.42	0.53	0.63	0.52
Østfold	Dec	0.35	0.27	0.35	0.51	0.38
Akershus	Jan	0.49	0.33	0.49	0.65	0.49
Akershus	Feb	0.70	0.48	0.64	0.75	0.61
Akershus	Mar	0.76	0.56	0.71	0.83	0.65
Akershus	Apr	0.80	0.62	0.76	0.85	0.71
Akershus	May	0.79	0.66	0.78	0.83	0.70
Akershus	Jun	0.79	0.64	0.78	0.83	0.70
Akershus	Jul	0.79	0.65	0.78	0.83	0.70
Akershus	Aug	0.81	0.71	0.81	0.83	0.73
Akershus	Sep	0.78	0.68	0.75	0.82	0.71
Akershus	Oct	0.73	0.60	0.67	0.76	0.65
Akershus	Nov	0.47	0.35	0.44	0.58	0.46
Akershus	Dec	0.30	0.18	0.36	0.45	0.35
Buskerud	Jan	0.47	0.28	0.53	0.66	0.47
Buskerud	Feb	0.76	0.58	0.71	0.83	0.64
Buskerud	Mar	0.83	0.69	0.80	0.88	0.72
Buskerud	Apr	0.84	0.74	0.82	0.89	0.74
Buskerud	May	0.84	0.75	0.80	0.88	0.75
Buskerud	Jun	0.85	0.76	0.82	0.88	0.75
Buskerud	Jul	0.84	0.78	0.80	0.89	0.76
Buskerud	Aug	0.84	0.74	0.81	0.89	0.75
Buskerud	Sep	0.83	0.72	0.81	0.86	0.74
Buskerud	Oct	0.76	0.59	0.71	0.83	0.66
Buskerud	Nov	0.56	0.40	0.53	0.64	0.50
Buskerud	Dec	0.34	0.25	0.36	0.54	0.41
Hedmark	Jan	0.58	0.37	0.57	0.64	0.51
Hedmark	Feb	0.75	0.57	0.72	0.81	0.66
Hedmark	Mar	0.82	0.70	0.78	0.88	0.73
Hedmark	Apr	0.85	0.76	0.83	0.88	0.77
Hedmark	May	0.84	0.75	0.82	0.86	0.77
Hedmark	Jun	0.84	0.76	0.82	0.87	0.78
Hedmark	Jul	0.85	0.75	0.84	0.87	0.79
Hedmark	Aug	0.85	0.76	0.82	0.87	0.79
Hedmark	Sep	0.81	0.75	0.78	0.82	0.76
Hedmark	Oct	0.76	0.64	0.70	0.76	0.70
Hedmark	Nov	0.54	0.43	0.48	0.58	0.50
Hedmark	Dec	0.22	0.15	0.25	0.43	0.29
Sør-Trøndelag	Jan	0.04	0.12	0.29	0.52	0.36
Sør-Trøndelag	Feb	0.59	0.46	0.53	0.70	0.53
Sør-Trøndelag	Mar	0.80	0.72	0.78	0.81	0.73
Sør-Trøndelag	Apr	0.83	0.74	0.77	0.85	0.75
Sør-Trøndelag	May	0.83	0.76	0.80	0.86	0.76
Sør-Trøndelag	Jun	0.81	0.74	0.79	0.84	0.74
Sør-Trøndelag	Jul	0.81	0.75	0.80	0.83	0.74
Sør-Trøndelag	Aug	0.82	0.73	0.79	0.85	0.75
Sør-Trøndelag	Sep	0.80	0.70	0.76	0.83	0.72

Continued on next page

Table G.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Sør-Trøndelag	Oct	0.73	0.61	0.68	0.74	0.65
Sør-Trøndelag	Nov	0.54	0.36	0.51	0.62	0.48
Sør-Trøndelag	Dec	0.00	0.02	0.13	0.45	0.23
Oslo	Jan	0.55	0.39	0.54	0.58	0.44
Oslo	Feb	0.63	0.52	0.66	0.68	0.53
Oslo	Mar	0.72	0.59	0.63	0.75	0.60
Oslo	Apr	0.75	0.62	0.72	0.77	0.64
Oslo	May	0.75	0.63	0.73	0.76	0.64
Oslo	Jun	0.77	0.62	0.74	0.79	0.65
Oslo	Jul	0.83	0.69	0.77	0.89	0.71
Oslo	Aug	0.79	0.70	0.76	0.81	0.68
Oslo	Sep	0.77	0.64	0.71	0.82	0.65
Oslo	Oct	0.67	0.62	0.67	0.69	0.58
Oslo	Nov	0.55	0.40	0.54	0.59	0.46
Oslo	Dec	0.37	0.27	0.42	0.47	0.38
Vestfold	Jan	0.56	0.41	0.51	0.67	0.56
Vestfold	Feb	0.77	0.63	0.71	0.86	0.73
Vestfold	Mar	0.84	0.70	0.76	0.90	0.79
Vestfold	Apr	0.88	0.77	0.82	0.91	0.82
Vestfold	May	0.87	0.77	0.82	0.90	0.82
Vestfold	Jun	0.88	0.76	0.82	0.91	0.83
Vestfold	Jul	0.87	0.76	0.82	0.92	0.82
Vestfold	Aug	0.83	0.76	0.82	0.92	0.83
Vestfold	Sep	0.73	0.71	0.78	0.87	0.79
Vestfold	Oct	0.59	0.62	0.72	0.81	0.73
Vestfold	Nov	0.49	0.46	0.49	0.69	0.57
Vestfold	Dec	0.36	0.27	0.47	0.58	0.48
Telemark	Jan	0.40	0.27	0.38	0.49	0.38
Telemark	Feb	0.67	0.59	0.71	0.76	0.70
Telemark	Mar	0.80	0.67	0.76	0.83	0.74
Telemark	Apr	0.83	0.69	0.82	0.87	0.78
Telemark	May	0.83	0.67	0.85	0.86	0.78
Telemark	Jun	0.86	0.69	0.84	0.88	0.80
Telemark	Jul	0.84	0.66	0.81	0.87	0.78
Telemark	Aug	0.81	0.65	0.80	0.85	0.75
Telemark	Sep	0.77	0.64	0.80	0.84	0.76
Telemark	Oct	0.65	0.60	0.67	0.80	0.69
Telemark	Nov	0.50	0.47	0.51	0.58	0.53
Telemark	Dec	0.13	0.25	0.33	0.55	0.38
Oppland	Jan	0.52	0.13	0.53	0.56	0.40
Oppland	Feb	0.67	0.58	0.68	0.70	0.58
Oppland	Mar	0.78	0.58	0.77	0.83	0.67
Oppland	Apr	0.79	0.59	0.78	0.86	0.68
Oppland	May	0.78	0.57	0.77	0.85	0.68
Oppland	Jun	0.79	0.58	0.80	0.85	0.68
Oppland	Jul	0.79	0.60	0.78	0.84	0.69
Oppland	Aug	0.79	0.64	0.78	0.85	0.69
Oppland	Sep	0.78	0.62	0.74	0.83	0.68
Oppland	Oct	0.69	0.53	0.65	0.75	0.60

Continued on next page

Table G.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Oppland	Nov	0.48	0.34	0.41	0.53	0.41
Oppland	Dec	0.46	0.26	0.44	0.56	0.39

Appendix H

PR for Each County and Month: Dataset 3) Poly data

Table H.1: PR for each county and month. Dataset 3) Poly data

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Rogaland	Jan	0.67	0.52	0.65	0.76	0.64
Rogaland	Feb	0.80	0.64	0.76	0.85	0.74
Rogaland	Mar	0.88	0.77	0.83	0.91	0.82
Rogaland	Apr	0.89	0.80	0.87	0.92	0.84
Rogaland	May	0.89	0.79	0.86	0.93	0.84
Rogaland	Jun	0.90	0.79	0.86	0.93	0.85
Rogaland	Jul	0.90	0.79	0.88	0.94	0.85
Rogaland	Aug	0.89	0.78	0.87	0.91	0.84
Rogaland	Sep	0.88	0.78	0.85	0.91	0.83
Rogaland	Oct	0.83	0.68	0.81	0.87	0.78
Rogaland	Nov	0.75	0.61	0.72	0.80	0.71
Rogaland	Dec	0.59	0.42	0.57	0.76	0.59
Hordaland	Jan	0.72	0.61	0.70	0.79	0.69
Hordaland	Feb	0.82	0.67	0.78	0.87	0.77
Hordaland	Mar	0.85	0.73	0.82	0.88	0.80
Hordaland	Apr	0.90	0.78	0.88	0.94	0.84
Hordaland	May	0.89	0.77	0.85	0.93	0.84
Hordaland	Jun	0.87	0.78	0.85	0.92	0.84
Hordaland	Jul	0.89	0.79	0.86	0.92	0.84
Hordaland	Aug	0.86	0.79	0.86	0.91	0.83
Hordaland	Sep	0.88	0.78	0.85	0.91	0.83
Hordaland	Oct	0.82	0.69	0.79	0.86	0.77
Hordaland	Nov	0.76	0.61	0.72	0.83	0.72
Hordaland	Dec	0.71	0.49	0.67	0.78	0.63
Østfold	Jan	0.62	0.50	0.68	0.80	0.66
Østfold	Feb	0.78	0.64	0.73	0.84	0.73
Østfold	Mar	0.82	0.69	0.78	0.87	0.77
Østfold	Apr	0.86	0.75	0.81	0.89	0.81
Østfold	May	0.85	0.74	0.81	0.89	0.80
Østfold	Jun	0.85	0.75	0.81	0.88	0.80
Østfold	Jul	0.85	0.76	0.81	0.88	0.80
Østfold	Aug	0.86	0.76	0.82	0.89	0.81

Continued on next page

Table H.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Østfold	Sep	0.84	0.72	0.80	0.87	0.79
Østfold	Oct	0.79	0.65	0.74	0.85	0.74
Østfold	Nov	0.68	0.55	0.68	0.80	0.68
Østfold	Dec	0.51	0.40	0.53	0.73	0.56
Akershus	Jan	0.70	0.47	0.64	0.75	0.61
Akershus	Feb	0.76	0.61	0.71	0.78	0.67
Akershus	Mar	0.79	0.62	0.76	0.85	0.70
Akershus	Apr	0.82	0.68	0.79	0.87	0.73
Akershus	May	0.81	0.68	0.80	0.86	0.73
Akershus	Jun	0.82	0.70	0.79	0.86	0.73
Akershus	Jul	0.81	0.67	0.79	0.85	0.72
Akershus	Aug	0.82	0.74	0.80	0.85	0.75
Akershus	Sep	0.81	0.70	0.78	0.84	0.73
Akershus	Oct	0.77	0.64	0.72	0.81	0.69
Akershus	Nov	0.66	0.46	0.60	0.72	0.59
Akershus	Dec	0.55	0.33	0.54	0.67	0.51
Buskerud	Jan	0.73	0.51	0.66	0.79	0.62
Buskerud	Feb	0.84	0.73	0.81	0.88	0.77
Buskerud	Mar	0.87	0.75	0.85	0.93	0.81
Buskerud	Apr	0.90	0.79	0.89	0.92	0.84
Buskerud	May	0.88	0.79	0.87	0.90	0.83
Buskerud	Jun	0.90	0.81	0.86	0.93	0.85
Buskerud	Jul	0.89	0.81	0.86	0.91	0.84
Buskerud	Aug	0.90	0.79	0.87	0.92	0.84
Buskerud	Sep	0.89	0.77	0.87	0.91	0.83
Buskerud	Oct	0.83	0.73	0.81	0.86	0.78
Buskerud	Nov	0.71	0.59	0.69	0.78	0.67
Buskerud	Dec	0.62	0.41	0.55	0.73	0.53
Hedmark	Jan	0.70	0.59	0.71	0.83	0.69
Hedmark	Feb	0.82	0.73	0.80	0.86	0.77
Hedmark	Mar	0.84	0.75	0.80	0.88	0.78
Hedmark	Apr	0.87	0.81	0.84	0.88	0.82
Hedmark	May	0.87	0.79	0.83	0.88	0.82
Hedmark	Jun	0.87	0.80	0.84	0.89	0.83
Hedmark	Jul	0.86	0.81	0.85	0.90	0.83
Hedmark	Aug	0.83	0.80	0.85	0.90	0.84
Hedmark	Sep	0.80	0.75	0.82	0.86	0.82
Hedmark	Oct	0.79	0.70	0.77	0.86	0.79
Hedmark	Nov	0.59	0.57	0.63	0.76	0.66
Hedmark	Dec	0.43	0.32	0.51	0.63	0.50
Sør-Trøndelag	Jan	0.64	0.29	0.59	0.78	0.55
Sør-Trøndelag	Feb	0.82	0.68	0.75	0.86	0.72
Sør-Trøndelag	Mar	0.84	0.74	0.80	0.89	0.76
Sør-Trøndelag	Apr	0.85	0.76	0.82	0.90	0.77
Sør-Trøndelag	May	0.86	0.77	0.83	0.90	0.78
Sør-Trøndelag	Jun	0.83	0.76	0.81	0.87	0.76
Sør-Trøndelag	Jul	0.85	0.76	0.81	0.87	0.77
Sør-Trøndelag	Aug	0.85	0.75	0.82	0.88	0.77
Sør-Trøndelag	Sep	0.86	0.74	0.82	0.92	0.77

Continued on next page

Table H.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Sør-Trøndelag	Oct	0.81	0.68	0.77	0.83	0.72
Sør-Trøndelag	Nov	0.76	0.54	0.73	0.78	0.66
Sør-Trøndelag	Dec	0.00	0.03	0.46	0.70	0.44
Oslo	Jan	0.73	0.55	0.71	0.77	0.63
Oslo	Feb	0.73	0.61	0.74	0.76	0.62
Oslo	Mar	0.77	0.66	0.68	0.85	0.66
Oslo	Apr	0.81	0.70	0.78	0.83	0.70
Oslo	May	0.83	0.70	0.79	0.86	0.72
Oslo	Jun	0.85	0.70	0.81	0.90	0.73
Oslo	Jul	0.83	0.71	0.80	0.87	0.71
Oslo	Aug	0.84	0.75	0.80	0.91	0.73
Oslo	Sep	0.82	0.77	0.77	0.89	0.71
Oslo	Oct	0.74	0.68	0.74	0.76	0.65
Oslo	Nov	0.67	0.57	0.62	0.67	0.57
Oslo	Dec	0.63	0.40	0.61	0.71	0.52
Vestfold	Jan	0.68	0.65	0.70	0.76	0.71
Vestfold	Feb	0.81	0.74	0.81	0.89	0.81
Vestfold	Mar	0.87	0.76	0.84	0.90	0.83
Vestfold	Apr	0.89	0.80	0.86	0.90	0.84
Vestfold	May	0.90	0.78	0.86	0.92	0.85
Vestfold	Jun	0.90	0.80	0.86	0.93	0.85
Vestfold	Jul	0.89	0.78	0.86	0.93	0.85
Vestfold	Aug	0.87	0.78	0.85	0.91	0.85
Vestfold	Sep	0.84	0.76	0.85	0.87	0.83
Vestfold	Oct	0.80	0.73	0.79	0.90	0.80
Vestfold	Nov	0.55	0.59	0.65	0.78	0.70
Vestfold	Dec	0.63	0.52	0.60	0.75	0.60
Telemark	Jan	0.68	0.62	0.64	0.71	0.63
Telemark	Feb	0.76	0.68	0.75	0.83	0.74
Telemark	Mar	0.84	0.73	0.81	0.88	0.80
Telemark	Apr	0.88	0.76	0.86	0.91	0.83
Telemark	May	0.88	0.75	0.87	0.90	0.83
Telemark	Jun	0.86	0.73	0.86	0.90	0.82
Telemark	Jul	0.85	0.72	0.83	0.87	0.80
Telemark	Aug	0.86	0.72	0.86	0.87	0.81
Telemark	Sep	0.85	0.71	0.83	0.88	0.80
Telemark	Oct	0.72	0.69	0.76	0.85	0.76
Telemark	Nov	0.68	0.61	0.68	0.77	0.69
Telemark	Dec	0.47	0.37	0.55	0.70	0.56
Oppland	Jan	0.73	0.52	0.71	0.77	0.59
Oppland	Feb	0.74	0.69	0.73	0.78	0.64
Oppland	Mar	0.76	0.63	0.77	0.83	0.66
Oppland	Apr	0.82	0.61	0.81	0.87	0.71
Oppland	May	0.81	0.60	0.80	0.88	0.70
Oppland	Jun	0.82	0.61	0.83	0.87	0.71
Oppland	Jul	0.82	0.64	0.82	0.89	0.72
Oppland	Aug	0.82	0.68	0.80	0.87	0.72
Oppland	Sep	0.82	0.66	0.77	0.90	0.71
Oppland	Oct	0.78	0.58	0.60	0.89	0.67

Continued on next page

Table H.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Oppland	Nov	0.66	0.46	0.52	0.73	0.56
Oppland	Dec	0.73	0.45	0.68	0.77	0.60

Appendix I

Specific Yield for Each County and Month

Table I.1: Specific yield for each county and month

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Rogaland	Jan	3.60	3.19	5.19	8.17	5.89
Rogaland	Feb	27.47	17.93	24.56	32.90	24.71
Rogaland	Mar	87.48	63.98	85.27	99.13	81.40
Rogaland	Apr	0.00	109.33	124.96	139.81	120.18
Rogaland	May	0.00	104.41	118.34	129.83	113.95
Rogaland	Jun	0.00	108.25	122.01	133.03	115.48
Rogaland	Jul	0.00	104.19	116.62	128.46	111.29
Rogaland	Aug	0.00	95.49	107.54	118.63	100.82
Rogaland	Sep	0.00	67.32	80.67	91.90	75.52
Rogaland	Oct	33.07	23.45	30.03	36.46	29.26
Rogaland	Nov	9.03	6.64	11.11	15.96	11.82
Rogaland	Dec	0.00	1.65	3.08	7.75	4.98
Hordaland	Jan	4.10	3.08	5.47	8.91	6.16
Hordaland	Feb	24.17	16.77	24.89	30.84	24.01
Hordaland	Mar	74.75	56.30	70.06	96.13	73.80
Hordaland	Apr	123.19	107.50	118.32	139.36	120.07
Hordaland	May	114.93	102.51	111.75	125.82	112.18
Hordaland	Jun	112.35	107.19	118.39	131.92	118.48
Hordaland	Jul	99.05	90.05	100.57	119.97	104.55
Hordaland	Aug	91.92	81.51	92.96	110.51	96.22
Hordaland	Sep	84.97	68.66	81.60	96.17	78.86
Hordaland	Oct	25.34	20.29	29.04	34.07	28.18
Hordaland	Nov	7.25	6.17	10.77	16.87	11.48
Hordaland	Dec	0.10	1.22	3.94	7.59	4.90
Østfold	Jan	4.36	3.27	5.19	7.98	5.51
Østfold	Feb	27.16	17.27	27.03	38.18	25.77
Østfold	Mar	92.49	55.83	84.42	104.09	78.09
Østfold	Apr	100.19	104.94	120.16	136.40	117.50
Østfold	May	101.67	116.60	135.99	155.45	137.92
Østfold	Jun	103.81	130.42	152.35	172.78	156.52
Østfold	Jul	110.37	121.76	145.64	161.67	140.96
Østfold	Aug	92.48	98.49	125.54	136.80	112.67
Østfold	Sep	63.71	62.52	76.43	88.65	70.09

Continued on next page

Table I.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Østfold	Oct	29.02	25.95	32.60	41.25	31.70
Østfold	Nov	5.33	4.93	6.66	8.07	6.36
Østfold	Dec	0.91	1.15	2.14	3.77	2.84
Akershus	Jan	0.00	1.95	4.08	6.78	4.59
Akershus	Feb	15.13	12.21	20.12	27.84	20.88
Akershus	Mar	67.48	34.32	73.11	93.86	67.41
Akershus	Apr	0.00	94.86	115.02	128.91	103.89
Akershus	May	131.46	103.73	124.89	137.68	116.06
Akershus	Jun	138.34	108.82	137.02	148.00	121.72
Akershus	Jul	0.00	117.39	129.35	146.75	120.32
Akershus	Aug	0.00	95.51	123.94	131.91	104.29
Akershus	Sep	0.00	55.74	64.10	75.50	58.66
Akershus	Oct	32.17	21.95	30.19	36.25	26.76
Akershus	Nov	5.72	3.36	4.95	6.71	4.67
Akershus	Dec	0.00	0.43	0.93	1.85	1.63
Buskerud	Jan	0.00	0.59	1.53	5.22	3.53
Buskerud	Feb	0.00	13.56	22.32	34.99	24.89
Buskerud	Mar	84.41	65.60	84.04	116.15	81.86
Buskerud	Apr	113.73	97.60	115.77	132.64	108.12
Buskerud	May	122.28	96.61	123.87	137.15	110.44
Buskerud	Jun	0.00	99.20	135.20	149.05	116.83
Buskerud	Jul	0.00	104.65	128.57	140.59	112.52
Buskerud	Aug	0.00	97.54	119.71	130.51	105.48
Buskerud	Sep	73.20	52.84	66.84	83.22	60.86
Buskerud	Oct	32.73	22.64	29.76	39.95	28.92
Buskerud	Nov	4.06	3.75	4.91	7.71	5.55
Buskerud	Dec	0.00	0.14	0.84	1.55	1.25
Hedmark	Jan	0.00	0.88	3.25	5.08	3.87
Hedmark	Feb	0.00	9.13	19.92	31.54	20.50
Hedmark	Mar	99.20	60.20	93.95	109.70	82.07
Hedmark	Apr	0.29	106.82	125.26	137.27	118.01
Hedmark	May	130.47	113.63	125.08	131.93	121.10
Hedmark	Jun	146.02	123.65	138.99	149.21	134.21
Hedmark	Jul	0.25	120.92	132.60	145.80	127.93
Hedmark	Aug	0.00	112.31	125.50	133.35	115.51
Hedmark	Sep	0.00	54.16	66.78	72.76	60.89
Hedmark	Oct	0.00	27.72	35.21	39.36	32.67
Hedmark	Nov	6.15	4.11	5.90	7.35	5.65
Hedmark	Dec	0.00	0.12	0.63	1.24	0.98
Sør-Trøndelag	Jan	0.00	0.33	1.14	2.40	1.39
Sør-Trøndelag	Feb	4.30	3.12	5.08	7.58	5.55
Sør-Trøndelag	Mar	15.98	53.39	58.44	61.03	54.10
Sør-Trøndelag	Apr	106.28	90.99	98.74	111.42	96.47
Sør-Trøndelag	May	112.46	104.63	111.47	114.15	105.02
Sør-Trøndelag	Jun	114.20	106.44	116.01	119.35	110.73
Sør-Trøndelag	Jul	78.17	77.74	85.76	90.36	84.00
Sør-Trøndelag	Aug	21.43	79.23	90.69	95.91	82.77
Sør-Trøndelag	Sep	0.00	62.92	68.98	74.77	61.21
Sør-Trøndelag	Oct	24.83	19.72	22.46	25.17	20.94

Continued on next page

Table I.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Sør-Trøndelag	Nov	3.35	3.53	7.31	10.73	7.88
Sør-Trøndelag	Dec	0.00	0.03	0.07	0.92	0.54
Oslo	Jan	0.00	0.58	3.48	5.29	3.52
Oslo	Feb	0.00	0.86	14.30	27.59	15.23
Oslo	Mar	22.21	26.04	31.89	38.97	35.52
Oslo	Apr	122.93	89.68	114.59	157.25	121.17
Oslo	May	139.83	103.61	127.31	181.80	139.04
Oslo	Jun	156.56	120.93	140.81	212.09	159.70
Oslo	Jul	161.32	115.56	141.24	184.46	136.27
Oslo	Aug	124.50	84.26	121.25	142.00	100.72
Oslo	Sep	67.59	40.24	64.24	77.90	54.05
Oslo	Oct	32.83	15.41	29.93	39.06	25.74
Oslo	Nov	5.55	2.64	5.16	6.78	4.78
Oslo	Dec	0.00	0.34	1.12	1.94	1.67
Vestfold	Jan	5.49	4.07	6.76	8.66	6.35
Vestfold	Feb	38.73	28.73	36.25	40.70	31.51
Vestfold	Mar	91.98	81.06	94.24	109.12	93.86
Vestfold	Apr	0.00	109.05	118.84	133.74	112.87
Vestfold	May	0.00	115.09	133.16	150.90	121.29
Vestfold	Jun	0.00	123.35	143.88	168.30	129.47
Vestfold	Jul	0.00	121.36	136.77	158.02	121.42
Vestfold	Aug	0.00	112.44	122.30	134.68	107.35
Vestfold	Sep	0.00	65.74	74.77	81.74	66.11
Vestfold	Oct	39.78	28.61	35.89	44.84	32.67
Vestfold	Nov	7.56	4.89	7.38	8.59	6.44
Vestfold	Dec	0.00	0.90	2.92	5.32	3.38
Telemark	Jan	0.00	0.02	2.69	6.95	3.81
Telemark	Feb	30.50	18.28	29.58	35.02	25.03
Telemark	Mar	98.28	66.34	88.91	108.45	87.00
Telemark	Apr	102.74	102.11	120.62	139.20	121.45
Telemark	May	127.94	120.38	134.67	150.87	136.45
Telemark	Jun	136.80	128.63	137.25	155.60	142.45
Telemark	Jul	0.00	113.21	134.60	147.50	125.91
Telemark	Aug	0.00	93.12	111.29	131.29	107.12
Telemark	Sep	74.91	53.82	65.18	86.53	62.24
Telemark	Oct	37.02	24.76	33.69	42.05	30.60
Telemark	Nov	7.74	4.45	7.12	8.86	6.20
Telemark	Dec	0.00	0.03	1.20	2.99	1.69
Oppland	Jan	0.00	0.01	1.07	2.69	2.09
Oppland	Feb	0.00	2.19	9.28	28.20	15.03
Oppland	Mar	87.89	60.67	84.40	97.04	72.54
Oppland	Apr	0.13	92.71	120.14	128.34	102.58
Oppland	May	0.14	69.50	113.46	125.06	95.89
Oppland	Jun	0.15	96.82	124.58	133.78	108.99
Oppland	Jul	0.22	105.18	125.30	128.56	107.94
Oppland	Aug	0.28	102.02	113.05	118.08	99.30
Oppland	Sep	0.16	50.84	63.17	65.59	55.52
Oppland	Oct	0.09	24.86	32.32	37.58	28.32
Oppland	Nov	5.06	3.78	4.89	5.44	4.37

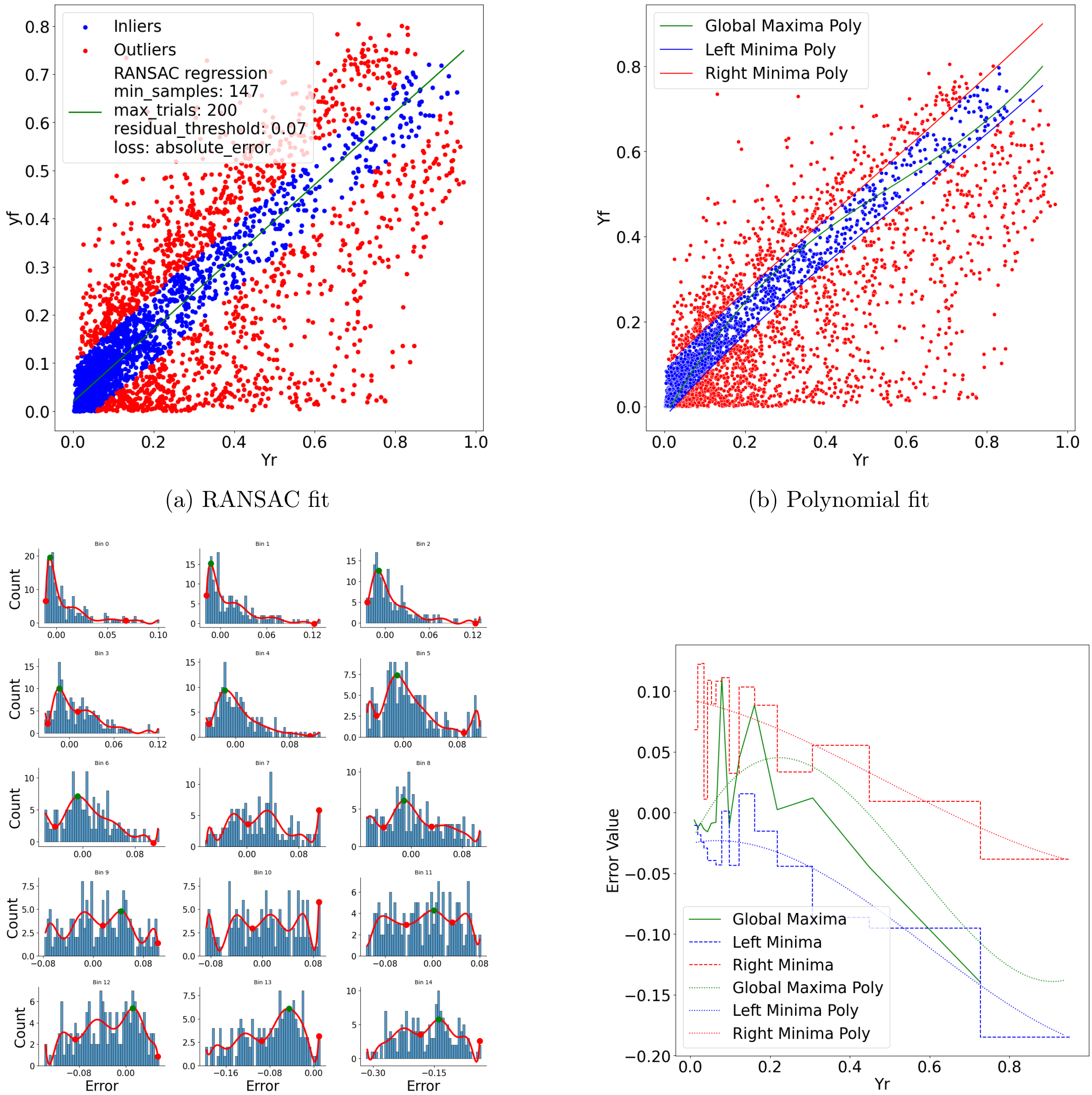
Continued on next page

Table I.1 Continued from previous page

Fylke	Month	Peak Weibull	Q1	Median	Q3	Mean
Oppland	Dec	0.00	0.05	0.34	0.88	0.64

Appendix J

Clustering Examples



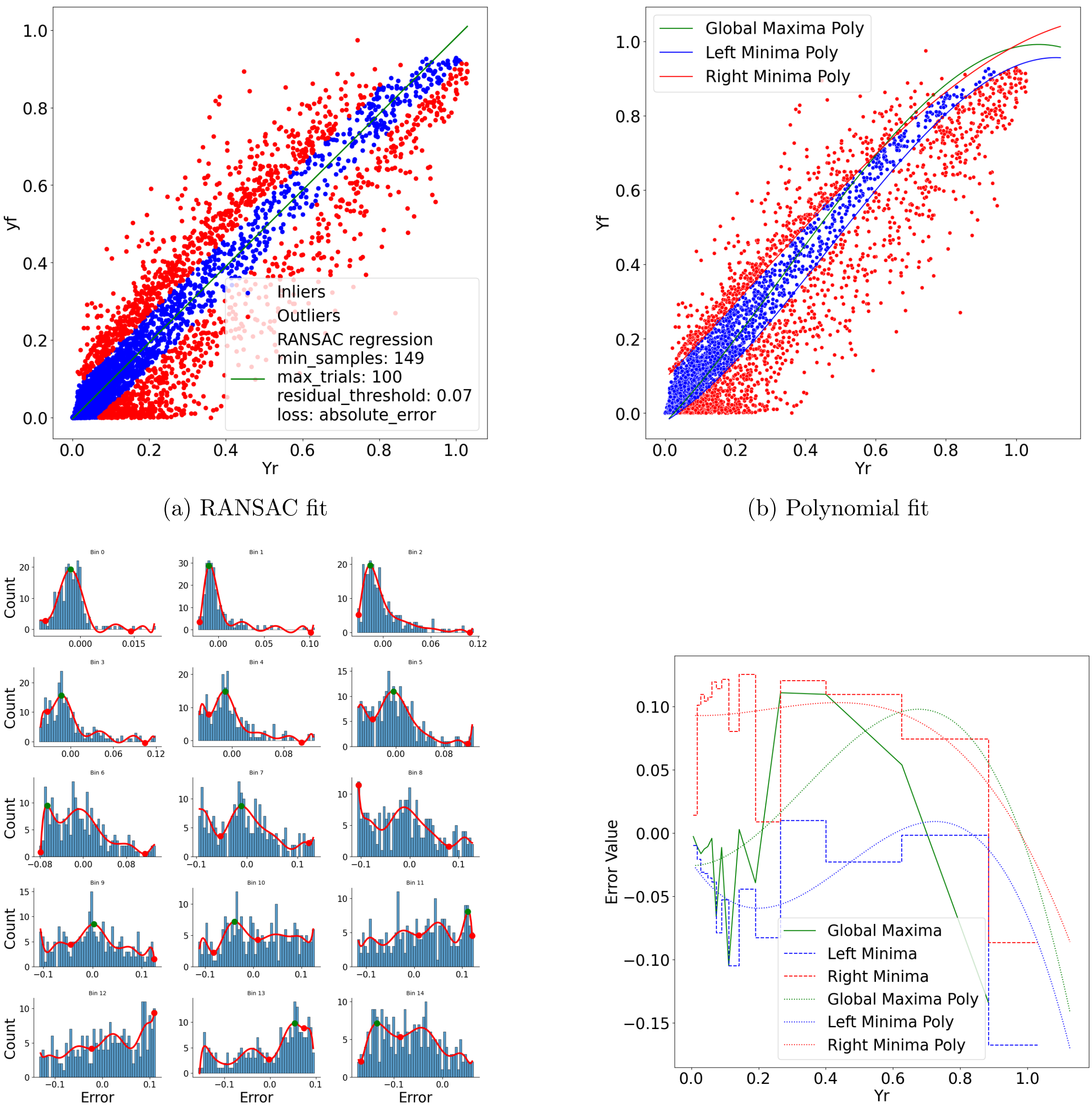
(a) RANSAC fit

(b) Polynomial fit

(c) Histograms

(d) Polynomial borders

Figure J.1: Appendix: clustering example 1



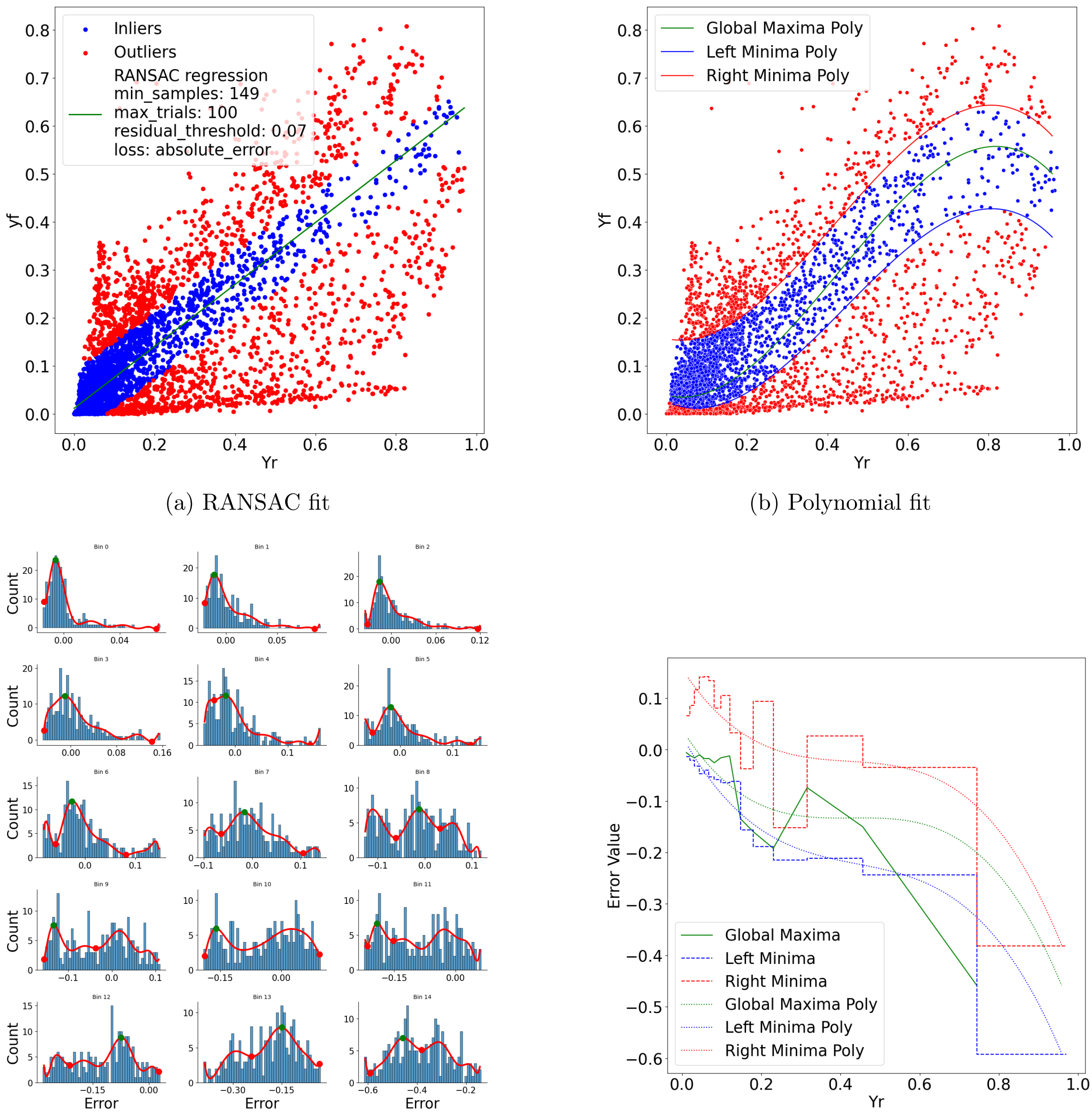
(a) RANSAC fit

(b) Polynomial fit

(c) Histograms

(d) Polynomial borders

Figure J.2: Appendix: clustering example 2



(a) RANSAC fit

(b) Polynomial fit

(c) Histograms

(d) Polynomial borders

Figure J.3: Appendix: clustering example 3

Appendix K

Code: Downloading Weather Data from CAMS

```

# -*- coding: utf-8 -*-
"""
Created on Tue Mar 21 11:21:51 2023

@author: marti
"""

import os
import fnmatch
import json
import pandas as pd
import re
import numpy as np
import math
import re
import glob
import pvlib
from datetime import datetime
from requests.exceptions import ReadTimeout
import requests
from tabulate import tabulate
import matplotlib.pylab as plt
import seaborn as sns

#%% getting CAMS weather data


######################## Cheking for daylight saving (this method has ...
    not been used)
#Path to folder
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_loca
#Setting folderpath as file
files = os.listdir(folder_path)

#Finding all data in the folder
unique_keys = []
for file in files:
    match = re.search(r'plant_(\d+)_location\.parquet', file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)
```

```

unique_keys = list(set(unique_keys))

no_daylight_pressent = []
daylight_pressent = []
for key in unique_keys:
    filename = f'{folder_path}\\plant_{key}_location.parquet'
    print(filename)
    daylight = pd.read_parquet(filename)

    # scheck for duplicate in the datetime column
    if daylight['datetime'].duplicated().any():
        daylight_pressent.append(key)
        print('the dataframe column has daylight saving present')
    else:
        no_daylight_pressent.append(key)
        print('the dataframe column does not have daylight saving ...
            present')

############################### Downloading weather data from CAMS

#setting up path to weather folder
weather_folder_path = "C:\\Users\\marti\\Desktop\\IFE\Værdata"
for key in unique_keys:
    print(key)
    # Check if the file already exists in the weather folder
    weather_filename = f'{weather_folder_path}\\cams_data_{key}.parquet'

    # cheking if weather data has already been downloaded
    if not os.path.exists(weather_filename):
        print(f"Downloading weather data for key: {key}")
        try:
            # getting metadata: colecting lon, lat, time
            metadata_location = ...
                f'{folder_path}\\plant_{key}_location.parquet'
            metadata = pd.read_parquet(metadata_location, columns = ...
                ["datetime","lat","lon"])

            # locating metadata: if there is metadata in the fiile: ...
                continue
            if not metadata.lat.empty and not metadata.lon.empty: # if ...
                metadata has information
                lat = metadata.lat.iloc[0]
                lon = metadata.lon.iloc[0]
                start_date = ...
                    datetime.strptime(metadata.datetime.min(), ...
                    "%Y-%m-%dT%H:%M:%S").strftime('%Y-%m-%d')
                start_date = pd.Timestamp(start_date, tz='Europe/Oslo')
                end_date = datetime.strptime(metadata.datetime.max(), ...
                    "%Y-%m-%dT%H:%M:%S").strftime('%Y-%m-%d')
                end_date = pd.Timestamp(end_date, tz='Europe/Oslo')

                #downloading weather data
                weather_data = pvlib.iotools.get_cams(latitude = lat, ...
                    longitude = lon, start = start_date, end = ...
                    end_date, email='martinkk@uia.no', identifier = ...
                    'cams_radiation', integrated = True, timeout = 45  )

                #extracing usfull information from weather data metadata
                weather_data_df = weather_data[0]
                weather_data_df["altitude"] = weather_data[1]["altitude"]
```

```
                weather_metadata_df = ...
                   pd.DataFrame.from_dict(weather_data[1], ...
                   orient='index').T

                #saving file
                weather_data_filename = ...
                   f'C:\\Users\\marti\\Desktop\\IFE\Værdata\\cams_data_{key}.parquet
                weather_metadata_filname = ...
                   f'C:\\Users\\marti\\Desktop\\IFE\Værdata\\cams_metadata_{key}.pa

                weather_data_df.to_parquet(weather_data_filename, ...
                   index=False)
                weather_metadata_df.to_parquet(weather_metadata_filname, ...
                   index=False)

        #error message
        except ReadTimeout:
                print(f"timout for key: {key}")
         #error message
        except requests.HTTPError as e:
                print(f"coordinates not found for key:{key}: ...
                   lat,long:({lat}, {lon}), error: {e}")
         #data aloready downloaded
    else:
        print(f"weather data already downloaded for key: {key}")



#%% Merging

######################## Finding the dwonloaded files
#folder path
folder_path = 'C:\\Users\\marti\\Desktop\\IFE\Værdata'
#defining folderpath as files
files = os.listdir(folder_path)

#finding data in folder
unique_keys = []
for file in files:
    match = re.search(r'cams_data_(\d+)\.parquet', file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)

unique_keys = list(set(unique_keys))

####################################
#################################### getting pvdata

# Set file path and name
file_path = ...
   "C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_new_capa
pvdata = pd.read_parquet(file_path)

####################################
#################################### merging data
merged_dfs = [] # setting up list

for key in unique_keys:
    print(f"key: {key}")
    #loading weather by the use of key
```

```
    cams = ...
       pd.read_parquet(f'C:\\Users\\marti\\Desktop\\IFE\Værdata\\cams_data_{key}.pa
    #adding key to CAMS
    cams["key"] = key

    #getting date
    cams["datetime"] = cams['Observation ...
       period'].str.extract(r'(\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}\.\d)')
    # converting time to datetime
    cams['datetime'] = pd.to_datetime(cams['datetime'])
    # drop unised column
    cams = cams.drop(columns=['Observation period'])

    #loading pvdata
    pvdata_filter = pvdata[pvdata["key"] == key].copy()

    #adding datetime
    pvdata_filter['datetime'] = pd.to_datetime(pvdata_filter['datetime'])

    #merging
    merged = pd.merge(pvdata_filter, cams, on=['key',"datetime"], ...
       how='left')

    #adding merged to the data in the previus loop
    merged_dfs.append(merged)

#merging all merged_dfs
new_pvdata = pd.concat(merged_dfs, axis=0, ignore_index=True)

#savind data
new_pvdata.to_parquet("C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2
```

Appendix L

Code: Merging Solcellespesialisten‘s Files, Adding Geolocation Data, Refining Capacity Data

```
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 14 17:53:55 2023

@author: marti
"""

import os
import fnmatch
import json
import pandas as pd
import re
import numpy as np
import math
import glob
import pvlib
from datetime import datetime
from requests.exceptions import ReadTimeout
import requests
from tabulate import tabulate
import matplotlib.pylab as plt
import seaborn as sns

import reverse_geocoder as rg # from ...
    https://pypi.org/project/reverse_geocoder/
import pandas as pd
import folium
from folium.plugins import MarkerCluster
from folium.plugins import HeatMap
import pyarrow.parquet as pq

#%%

def process_dataframe_by_chunks_to_parquet(df, key_column, ...
    date_column, chunk_size, aggregations, output_file_prefix):
    df = df.copy()
    df['chunk_id'] = np.arange(len(df)) // chunk_size
    df[date_column] = pd.to_datetime(df[date_column])

    for (chunk_id, key), group_df in df.groupby(['chunk_id', key_column]):
```

```
        print(f"Processing chunk_id {chunk_id}, key {key}")
        df_hourly = group_df.set_index(date_column, ...
            drop=False).resample('H').agg(aggregations)
        output_file = ...
            f"{output_file_prefix}_chunk_{chunk_id}_key_{key}.parquet"
        df_hourly.to_parquet(output_file, engine='pyarrow')


def load_parquet_files_to_dataframe(input_file_prefix):
    files = glob.glob(f"{input_file_prefix}_chunk_*.parquet")
    num_files = len(files)
    dataframes = []

    for i, file in enumerate(files):
        print(f"Loading file {i + 1} of {num_files}")
        df = pd.read_parquet(file, engine='fastparquet')
        dataframes.append(df)

    print("concatinating df")
    combined_df = pd.concat(dataframes)
    return combined_df


#%%
# file_path to all files
file_path = ...
    "C:\\Users\\marti\\Desktop\\IFE\\OneDrive_2023-03-13\\Sunpoint ...
    Merged_data"

# list to store values
all_data_indices = []


for filename in os.listdir(file_path):
    if fnmatch.fnmatch(filename, 'plant_*_Metadata.csv'):
        # get name
        index = int(filename.split('_')[1])
        # check for matching plant name
        plant_filename = f'plant_{index}.json'
        if plant_filename in os.listdir(file_path):
            all_data_indices.append(index)


# print maching results
print('Matching indices:')
print(all_data_indices)
#%



all_data = []
plant_df = []
output_folder = ...
    "C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_1"

if not os.path.exists(output_folder):
    os.makedirs(output_folder)

for index in all_data_indices:


```

```
    print(index)

    plant_filename = f'plant_{index}.json'
    metadata_filename = f"plant_{index}_Metadata.csv"


    plant_path = os.path.join(file_path, plant_filename)
    metadata_path = os.path.join(file_path, metadata_filename)

    #loading plant data
    with open(plant_path, 'r') as f:
        plant_data = json.load(f)

    #loading metadata
    metadata_data = pd.read_csv(metadata_path)

    #adding key
    metadata_data.insert(0, "key", index)

    #convert json to df with loop
    for lst in plant_data:
        temp_df = pd.DataFrame(lst)
        #add key
        temp_df.insert(0, "key", index)
        # Add metadata
        for col in metadata_data.columns:
            temp_df[col] = metadata_data.at[0, col]

        # merging list from this itteration with last itteration
        plant_df.append(temp_df)

    # merge and save output
    merged_data = pd.concat(plant_df, axis=0)
    output_file = os.path.join(output_folder, ...
       f"plant_{index}_merged.parquet")
    merged_data.to_parquet(output_file)
    plant_df = []  #clear list for next itteration


merged_data.columns



#%% Making hourly data

#folder path
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_1'
#defining folderpath as files
files = os.listdir(folder_path)

#finding data in folder
unique_keys = []
for file in files:
    match = re.search(r'plant_(\d+)_merged\.parquet', file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)

unique_keys = list(set(unique_keys))
```

```

#ssetting up aggregation method
aggregations = {
    'key': 'first',
    'timedate': "first",
    "Capacity": "first",
    'acproduction': 'mean',
    'dailyproduction': 'last',
    'totalproduction': 'last',
    'vnom': 'mean',
    'vl1': 'mean',
    'vl2': 'mean',
    'vl3': 'mean',
    'il1': 'mean',
    'il2': 'mean',
    'il3': 'mean',
    'frequency': 'mean',
    'runhours': 'last',
    'temperature': 'mean',
    'mocked': 'first',
    'mppt': 'first',
    "lat": "first",
    "lon": "first",

}




# for loop to loop thue all parquet files in the selected folder
for key in unique_keys:
    filename = f'{folder_path}\\plant_{key}_merged.parquet'
    print(filename)
    subset = pd.read_parquet(filename)

    subset['year'] = pd.to_datetime(subset['timedate']).dt.year
    subset['month'] = pd.to_datetime(subset['timedate']).dt.month
    subset['date'] = pd.to_datetime(subset['timedate']).dt.day
    subset['hour'] = pd.to_datetime(subset['timedate']).dt.hour

    sorted_data = subset.sort_values(['key', 'timedate'])
    aggregated_data = sorted_data.groupby(['year', 'month', 'date', ...
       'hour']).agg(aggregations).reset_index()

    # save the data
    new_filename = ...
       f'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Ra
    aggregated_data.to_parquet(new_filename, index=False)





#%% adding location data

#setting path to folder
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_hou
#setting name of files
file_pattern = os.path.join(folder_path, 'plant_*_hourly.parquet')
files = glob.glob(file_pattern)
```

```python

#getting files
data_list = []
for file in files:
    print(file)
    df = pd.read_parquet(file, columns=['key', 'lat', 'lon'])
    data_list.append(df)

print(f'Number of files: {len(data_list)}')


coordinates = pd.concat(data_list, ignore_index=True)

#finding the uniqeu keys in the df
coordinates = coordinates.drop_duplicates(subset='key', keep='first')

#lists for later use
no_location_key = []
coordinate_results_list = []

#Finding lat and long in data
for index, row in coordinates.iterrows():
    try:
        lat = str(row['lat'])
        lon = str(row['lon'])
        key = row.key
        coordinates = (lat,lon)
        results = rg.search(coordinates)
        print(key)


        results_dict = {'Key': key, **results[0]}

        coordinate_results_list.append(results_dict)
    except:
        print(f"no location data for key: {key}")
        no_location_key.append(key)

# convert lists into df
location_data = pd.DataFrame(coordinate_results_list)
location_data = location_data.rename(columns={'name': 'city'})
location_data = location_data.rename(columns={'admin1': 'Fylke'})
location_data = location_data.rename(columns={'admin2': 'kommune'})
location_data = location_data.rename(columns={'cc': 'country'})

#deliting lat and long
location_data = location_data.drop(columns = ["lat","lon"])

#setting up df to store missing locations
latexdf = pd.DataFrame(columns=["Number of instances"])
latexdf.loc["missing coordinates", "Number of instances"] = 0
latexdf.loc["missing city", "Number of instances"] = 0
latexdf.loc["location not in Norway", "Number of instances"] = 0

#saving data if location is found into folder: ...
    IFE_Data_13.03.2023_merged_Raw_location
for file in files:
    print(file)
    #load file
    df = pd.read_parquet(file)
    #loacte key
```

```
    key = int(os.path.basename(file).split('_')[1])

    # merge loaction with data
    merged_df = pd.merge(df, location_data, left_on='key', ...
        right_on='Key', how='left')

    # drop key to avoid duplicate
    merged_df = merged_df.drop(columns=['Key'])

    #saving file if it do not have missing location data
    if key not in no_location_key:

        # Filter wrong locations
        missing_coordinates = merged_df["lat"].isnull()
        missing_city = (~merged_df["lat"].isnull()) & ...
            merged_df["city"].isnull()
        location_not_in_norway = (merged_df["country"] != "NO")

        # Count instances for each condition
        latexdf.loc["missing coordinates", "Number of instances"] += ...
            int(missing_coordinates.any())
        latexdf.loc["missing city", "Number of instances"] += ...
            int(missing_city.any())
        latexdf.loc["location not in Norway", "Number of instances"] ...
            += int(location_not_in_norway.any())

        # Drop rows based on filter conditions
        merged_df = merged_df[~(missing_coordinates | missing_city | ...
            location_not_in_norway)]

        # Save the new data into new folder
        new_filename = ...
            f'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merge
        merged_df.to_parquet(new_filename, index=False)


        ############################################

    ############################filtering wrong locations
    if key in no_location_key:
        #the file is not save
        print(f"missing location information in key: {key}, file not ...
            saved")



#storing missing values as latex table
file_path = "C:\\Users\\marti\\Desktop\\IFE\\Tabeller\\Location_table.tex"
with open(file_path, 'w') as f:
    f.write(latexdf.to_string())



#%% renaming columns
############## making list of cites in folder
#folder path
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_loca
#defining folderpath as files
files = os.listdir(folder_path)

```

```
#finding data in folder
unique_keys = []
for file in files:
    match = re.search(r'plant_(\d+)_location\.parquet', file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)

unique_keys = list(set(unique_keys))


for key in unique_keys:
    filename = f'{folder_path}\\plant_{key}_location.parquet'
    print(filename)
    naming_df = pd.read_parquet(filename)

    # apply column renaming
    naming_df = naming_df.rename(columns={
                                        'key': 'key',
                                        "timedate": 'datetime',
                                        'date': 'date',
                                        'time': 'time',
                                        "Capacity": "capacity[w]",
                                        'delta': 'delta',
                                        'acproduction': ...
                                            'acproduction[wh]',
                                        'dailyproduction': ...
                                            'dailyproduction[kwh]',
                                        'totalproduction': ...
                                            'totalproduction[kwh]',
                                        'monthTotalproduction': ...
                                            'monthtotalproduction[kwh]',
                                        'yearTotalproduction': ...
                                            'yeartotalproduction[kwh]',
                                        'vnom': 'vnom',
                                        'vl1': 'vl1',
                                        'vl2': 'vl2',
                                        'vl3': 'vl3',
                                        'il1': 'il1',
                                        'il2': 'il2',
                                        'il3': 'il3',
                                        'frequency': 'frequency',
                                        'runhours': 'runhours',
                                        'temperature': 'temperature',
                                        'mocked': 'mocked',
                                        'mppt': 'mppt',
                                        "lat": "lat",
                                        "lon": "lon"
                                        })

    # save the updated dataframe with the same filename
    naming_df.to_parquet(filename, index=False)


#%% calculating spesific


#folder path
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_loca
#defining folderpath as files
```

```
files = os.listdir(folder_path)

#finding data in folder
unique_keys = []
for file in files:
    match = re.search(r'plant_(\d+)_location\.parquet', file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)

unique_keys = list(set(unique_keys))

########## getting files
folder_path = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_loca
#setting name of files
file_pattern = os.path.join(folder_path, 'plant_*_location.parquet')
files = glob.glob(file_pattern)


#getting files
data_list = []
for file in files:
    print(file)
    df = pd.read_parquet(file)
    data_list.append(df)

print(f'Number of files: {len(data_list)}')

#merging list into df
pvdata = pd.concat(data_list, ignore_index=True)

"""
####################################
#################################### Removing days, months, and year ...
    without power production

pvdata['datetime'] = pd.to_datetime(pvdata['datetime'])

# Removing days where production is 0
pvdata_day_orignial_1 = pvdata.copy()
pvdata = pvdata[pvdata.groupby([pd.Grouper(key='datetime', freq='Y'), ...
    pd.Grouper(key='datetime', freq='M'), pd.Grouper(key='datetime', ...
    freq='D'), 'key'])['acproduction[wh]'].transform(lambda x: ...
    x.ne(0).any())]
day_len = pvdata.copy()
num_days_removed = (len(pvdata_day_orignial_1) - len(pvdata))/24

# Removing months where production is 0
pvdata__month_orignial = pvdata.copy()
pvdata = pvdata[pvdata.groupby([pd.Grouper(key='datetime', freq='Y'), ...
    pd.Grouper(key='datetime', freq='M'), ...
    'key'])['acproduction[wh]'].transform(lambda x: x.ne(0).any())]
month_len = pvdata.copy()
num_months_removed = (len(pvdata__month_orignial) - len(pvdata))/24

# Removing years where production is 0
pvdata_orignial = pvdata.copy()
pvdata = pvdata[pvdata.groupby([pd.Grouper(key='datetime', ...
    freq='Y')])['acproduction[wh]'].transform(lambda x: x.ne(0).any())]
year_len = pvdata.copy()
```

```
num_years_removed_yearly = (len(pvdata_orignial) - len(pvdata))/24

# saving result to latex file
table = [["variabel", "Number of rows deleted", "number of days deleted"],
        ['Total rows before deletion', len(pvdata_day_orignial_1), ...
           'Number of days'],
         ['days', len(day_len), num_days_removed],
         ['month', len(month_len), num_months_removed],
         ['year', len(year_len), num_years_removed_yearly]
        ]

with ...
    open("C:\\Users\\marti\\Desktop\\IFE\\Tabeller\\pvdata_ife_raw_data_yearly_no_pow
    'w') as f:
     f.write(tabulate(table, tablefmt='latex_booktabs'))
"""
####################################
#################################### Spesific yield

#calculating new key grouper with new data
pvdata['datetime'] = pd.to_datetime(pvdata['datetime'])
key_group = pvdata.groupby('key')

 #setting up list for later use
yearly_wh_list = []
#calculating yearly spesific yeld
for key, df in key_group:
    #print(key, year.year)
    capcaity = df.loc[df.index[0], "capacity[w]"]
    yearly_wh_value = df["acproduction[wh]"].sum()
    yearly_wh_list.append((key, yearly_wh_value, capcaity))

yearly_wh_df = pd.DataFrame(yearly_wh_list, columns=['key', ...
    'yearly_Wh', "capacity[w]"])

#converting to capacity[kWp]
#yearly_wh_df["capacity[kwp]"] = yearly_wh_df["capacity[w]"] / 1000
#calculating spesific year [kWh/y / kWp]
yearly_wh_df["yearly_spesific_yield"] = yearly_wh_df["yearly_Wh"] / ...
    yearly_wh_df["capacity[w]"]
#%%
#Ploting spesific yield
fig, ax = plt.subplots(figsize = (12,12))
x = yearly_wh_df.reset_index().index
sns.scatterplot(data=yearly_wh_df, x=x,y="yearly_spesific_yield" , ...
    alpha=1,)
#plt.title('Yearly Spesific yield', size=25)
#plt.legend(title='installation number', fontsize=12, title_fontsize=25)
plt.xlabel('Installation number', size=25)
plt.ylabel('Spesific yield [kWh/kWp]', size=25)
plt.xticks(fontsize = 25)
ax.set_ylim([0, 2500])
plt.yticks(fontsize = 25)
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Raw_data\\spesific_yield.png",
plt.clf()
plt.close()

yearly_wh_df["yearly_spesific_yield"].describe()
```

```
pvdata = pd.merge(pvdata, yearly_wh_df[['key', ...
    'yearly_spesific_yield', "yearly_Wh"]], on=['key'])



#%%Adjusting capcaity

###################################
################################### Adjusting capcaity

#creating empty columns for later use
pvdata['capacity_adjusted[kwp]'] = np.nan
pvdata['spesific_yield_adjusted'] = np.nan
pvdata['plot'] = np.nan

pvdata_key_group = pvdata.groupby('key')



#assuming capacity is in Watt
modified_groups = []
for row, group in pvdata_key_group:

    if (group['capacity[w]'].max() < 70_000) and ...
        (group['yearly_spesific_yield'].min()< 2500):
        # save 200 as not adjusted
        group["plot"] = 200
        # capacity adjustment
        group['capacity_adjusted[kwp]'] = (group['capacity[w]']/1000)
        # calculating new spesific yield
        group['spesific_yield_adjusted'] = ((group['yearly_Wh']/1000) ...
            / group['capacity_adjusted[kwp]'])

    elif (group['yearly_spesific_yield'].min() > 2500) and ...
        (group["capacity[w]"].min() < 200):
        #capacity adjustment
        group['capacity_adjusted[kwp]'] = ((group['capacity[w]']/1000) ...
            * 1000)
        #calculate new spesific yield
        group['spesific_yield_adjusted'] = ((group['yearly_Wh']/1000) ...
            / (group['capacity_adjusted[kwp]']))
        ## save 100 as adjustment
        group["plot"] = 1000 # was divided
    elif group['yearly_spesific_yield'].max() < 5:
        # adjusting capcaity
        group['capacity_adjusted[kwp]'] = ((group['capacity[w]']/1000) ...
            / 1000)
        # calculate new spesific yield
        group['spesific_yield_adjusted'] = (group['yearly_Wh']/1000) / ...
            group['capacity_adjusted[kwp]']
        # save -1000 as adjustment
        group["plot"] = -1000 # was multiplied
    else:
        group["plot"] = 200 # was multiplied
        group['capacity_adjusted[kwp]'] = (group['capacity[w]']/1000)
        group['spesific_yield_adjusted'] = ((group['yearly_Wh']/1000) ...
            / group['capacity_adjusted[kwp]'])

    #appending data to the lists
    modified_groups.append(group)
#merging the lists
```

```
pvdata = pd.concat(modified_groups)


#Renaming spesific yield to yearly spesific yield
pvdata.rename(columns={'yearly_spesific_yield': 'old_spesific_yield'}, ...
    inplace=True)
pvdata.rename(columns={'spesific_yield_adjusted': ...
    'yearly_spesific_yield'}, inplace=True)
pvdata.rename(columns={'capacity_adjusted[kwp]': 'capacity[kwp]'}, ...
    inplace=True)



#### Dubble checking if the cites got adjusted similarly over the years

# Group by key and check if all values in the "plot" column are similar
groups = pvdata.groupby('key')
for key, group in groups:
    if len(group) == 1:
        continue#print(f"Key '{key}' has only one value and is ...
            excluded from the analysis.")
    else:
        std_dev = group['plot'].std()
        if std_dev < 0.1:
            print(f"All values in the 'plot' column for key '{key}' ...
                are similar.")
        else:
            print(f"Not all values in the 'plot' column for key ...
                '{key}' are similar.")

df = pvdata.drop_duplicates(subset='key', keep='first')
#storing result information
table = [['Total cites multiplied', (df["plot"] == 1_000).sum()],
         ['Total cites divided', (df["plot"] == -1_000).sum()],
         ['Total cites unchanged', (df["plot"] == 200).sum()],
         ]

with ...
    open("C:\\Users\\marti\\Desktop\\IFE\\Tabeller\\pvdata_ife_raw_data_altered_cap.t
    'w') as f:
     f.write(tabulate(table, headers=['Metric', 'Value'], ...
        tablefmt='latex_booktabs'))

# Ploting spesific yield differnce between log and actual
fig, ax = plt.subplots(figsize = (12,12))
sns.barplot(data=df, x='key',y="plot")
plt.title('Factor between recorded and actual value', size=20)
plt.xlabel('Site', size=15)
plt.ylabel('Factor', size=15)
plt.xticks([])
#ax.set_ylim([0, 10000])
plt.yticks(fontsize = 15)
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Raw_data\\Raw_site_capacity_lo
plt.clf()
plt.close()

df["yearly_spesific_yield"].describe()

#Ploting spesific yield
fig, ax = plt.subplots(figsize = (12,12))
x = df.reset_index().index
```

```
sns.scatterplot(data=df, x=x,y="yearly_spesific_yield" , alpha=1,)
#plt.title('Yearly Spesific yield', size=25)
#plt.legend(title='installation number', fontsize=12, title_fontsize=25)
plt.xlabel('Installation number', size=25)
plt.ylabel('Spesific yield [kWh/kWp]', size=25)
plt.xticks(fontsize = 25)
ax.set_ylim([0, 2500])
plt.yticks(fontsize = 25)
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Raw_data\\adjusted_spesific_yi
plt.clf()
plt.close()

####################################
#################################### Using adjusted ...
    capacity_adjusted[kwp] to calculate monthly spesific yield

#calculating new key group with new data
key_group = pvdata.groupby(['key', pd.Grouper(key='datetime', freq='M')])

monthly_wh_list = [] #setting up list for later use
#calculating yearly spesific yeld
for (key, date), df in key_group:
    print(key, date.month, date.year)
    capcaity = df.loc[df.index[0], "capacity[kwp]"]
    monthly_wh_value = df["acproduction[wh]"].sum()
    print(monthly_wh_value)
    monthly_wh_list.append((date.year, date.month, key, ...
        monthly_wh_value, capcaity))

monthly_wh_df = pd.DataFrame(monthly_wh_list, columns=["year", ...
    "month", 'key', 'monthly_Wh', "capacity[kwp]"])

#calculating spesific year [kWh/y / kWp]
monthly_wh_df["monthly_spesific_yield"] = (monthly_wh_df["monthly_Wh"] ...
    /1000)  / monthly_wh_df["capacity[kwp]"]

#storing result in pvdata
pvdata = ...
    pvdata.merge(monthly_wh_df[["monthly_spesific_yield","key","year","month"]], ...
    on=["key","year","month"], how='left')

#ploting
pvdata_hourly_unique_plot = pvdata.groupby(['key', 'year', ...
    'month']).agg('last')[['capacity[kwp]', 'monthly_spesific_yield', ...
    "datetime", "yearly_spesific_yield"]]
pvdata_hourly_unique_plot['month_rounded'] = ...
    pvdata_hourly_unique_plot['datetime'].dt.to_period('M').dt.to_timestamp()


#Ploting spesific yield
fig, ax = plt.subplots(figsize = (12,12))
sns.scatterplot(data=pvdata_hourly_unique_plot, ...
    x='month_rounded',y="monthly_spesific_yield" , alpha=1, ...
    palette='rocket')
#plt.title('Monthly Spesific yield', size=25)
#plt.legend(title='Capacity [kWp]', fontsize=12, title_fontsize=15)
plt.xlabel('Capacity [kWp]', size=25)
plt.ylabel('Monthly spesific yield [kwh/kWp]', size=15)
plt.xticks(fontsize = 25)
#ax.set_ylim([0, 10000])
plt.yticks(fontsize = 25)
```

```
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Raw_data\\Raw_data_spesific_yi
plt.clf()
plt.close()

#Ploting histogram of capacity
plot = pvdata.drop_duplicates(subset='key')

#breaking axis
break_point_1 = 100
break_point_2 = 50

data_before_break = plot[plot['capacity[kwp]'] <= break_point_1]
data_after_break = plot[plot['capacity[kwp]'] > break_point_1]

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 8), sharey=True, ...
    gridspec_kw={'width_ratios': [1, 1]})
# ploting data before break
sns.histplot(data=data_before_break, x='capacity[kwp]', ax=ax1, ...
    alpha=1, palette='rocket', bins = 100)
ax1.set_xlim(0, break_point_1)

# setting title information for ax1
ax1.set_xlabel('Capacity [kWp]', fontsize=25)
ax1.set_ylabel('PV installations', fontsize=25)
ax1.tick_params(axis='x', labelsize=25)
ax1.tick_params(axis='y', labelsize=25)

# Pploting the data after the break
sns.histplot(data=data_after_break, x='capacity[kwp]', ax=ax2, ...
    alpha=1, palette='rocket',bins = 20)
#ax2.set_xlim(break_point_2, data_after_break['capacity[kwp]'].max())

# setting title information for ax2
ax2.set_xlabel('Capacity [kWp]', fontsize=25)
ax2.set_ylabel('PV installations', fontsize=25)
ax2.tick_params(axis='x', labelsize=25)
ax2.tick_params(axis='y', labelsize=25)
#saving plot
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Raw_data\\Raw_capacity_distrub
plt.clf()
plt.close()

plot["capacity[kwp]"].describe()
####################################
#################################### Saving new data

# Save the aggregated data
new_filename = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_new_capa
pvdata.to_parquet(new_filename, index=False)
```

Appendix M

Code: Finding Missing Timestamps

```
# -*- coding: utf-8 -*-
Created on Thu Mar 16 13:42:45 2023

@author: marti

import os
import fnmatch
import json
import pandas as pd
import re
import numpy as np
import math
import glob
import pvlib
from datetime import datetime
from requests.exceptions import ReadTimeout
import requests
#from dataprep.eda import create_report
from tabulate import tabulate
import matplotlib.pylab as plt
import seaborn as sns
from datetime import timedelta
import reverse_geocoder as rg
import pandas as pd
import folium
from folium.plugins import MarkerCluster
from folium.plugins import HeatMap


#%% Loading data

####################################
#################################### Loading data

parquet_file = ...
    "C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_weather_5\\pvda

# Read the parquet file into a DataFrame
pvdata = pd.read_parquet(parquet_file)

#%%

####################################
#################################### finding missing timestamp on ...
    5-min interval basis
```

```
############finding keys.
#folder path
folder_path = "C:\\Users\\marti\\Desktop\IFE\\Værdata"
#defining path
files = os.listdir(folder_path)

#finding data in folder
unique_keys = []
for file in files:
    match = re.search(r"cams_data_(\d+)\.parquet", file)
    if match:
        key = int(match.group(1))
        unique_keys.append(key)

unique_keys = list(set(unique_keys))

########## getting files

#Funtion to make expected timestamp
def generate_timestamp_expected(year, month):
    start_date = pd.Timestamp(year, month, 1)
    days_in_month = start_date.days_in_month
    end_date = start_date + timedelta(days=days_in_month)
    return pd.date_range(start=start_date, end=end_date, freq="5min", ...
        closed="left")

#making emty df
missing_timestamps_info = pd.DataFrame(columns=["key", "year", ...
    "month", "missing_intervals"])

pvdata_folder = ...
    "C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13.03.2023_merged_Raw_1"

for key in unique_keys:
    filename = f"{pvdata_folder}\\plant_{key}_merged.parquet"
    print(filename)
    subset = pd.read_parquet(filename)

    # Extract year and month from the "datetime" column
    subset["datetime"] = pd.to_datetime(subset["timedate"])
    subset["year"] = subset["datetime"].dt.year
    subset["month"] = subset["datetime"].dt.month

    # Group by year and month
    grouped = subset.groupby(["year", "month"])


    for (year, month), group in grouped:
        timestamp_expected = generate_timestamp_expected(year, month)
        existing_timestamps = group["datetime"]
        missing_timestamps = ...
            timestamp_expected[~timestamp_expected.isin(existing_timestamps)]
        missing_intervals = len(missing_timestamps)

        # Saving information
        missing_timestamps_info = ...
            missing_timestamps_info.append({"key": key,
                                                                      "year": ...
                                                                          year,
```

```
                                                                    "month": ...
                                                                       month,
                                                                    "missing_intervals
                                                                       missing_interva
                                                                   ignore_index=True)

#finding missing days
missing_timestamps_info["missing_days"] = ...
    missing_timestamps_info["missing_intervals"] / (24 * 60 /5)

print(missing_timestamps_info)


#%%
####################################
#################################### finding missing timestamp on ...
    hourly basis

# function to find missing hours
def monthly_hours(year, month):
     days = pd.date_range(start=f"{year}-{month:02d}-01", ...
         periods=pd.Timestamp(year, month, 1).days_in_month, freq="D")
     return days.shape[0] * 24

df = pvdata.copy()

expected_hours = df.groupby(["key", "year", ...
    "month"]).size().reset_index(name="expected_hours")
#hours availabe
expected_hours["monthly_hours"] = expected_hours.apply(lambda row: ...
    monthly_hours(row["year"], row["month"]), axis=1)
#percentage
expected_hours["percent_available"] = ...
    (expected_hours["expected_hours"] / ...
    expected_hours["monthly_hours"]) * 100
filtered_df = expected_hours[expected_hours["percent_available"] >= ...
    90]# removin month if it has less than 30

print(filtered_df)

expected_hours["percent_available_bins"] = ...
    pd.cut(expected_hours["percent_available"], bins=[0, 10, 50, 90, ...
    95, 99, 100], labels=["0-10%", "10-50%", "50-90%", "90-95%", ...
    "95-99%", "99-100%"], duplicates="drop")

table = ...
    expected_hours["percent_available_bins"].value_counts().sort_index(ascending=Fal
table.columns = ["Category", "Count"]

with ...
    open("C:\\Users\\marti\\Desktop\IFE\\Tabeller\\missing_timestamp.tex", ...
    "w") as f:
     f.write(table.to_latex(index=False))

#Ploting histogram of filtered_df
fig, ax = plt.subplots(figsize = (12,12))
sns.displot(data=filtered_df, x="percent_available")
# Set the x and y labels and the title
plt.title("Number of installations per municipality", size=20)
plt.legend(title="Municipality", fontsize=12, title_fontsize=15)
plt.xlabel("Month", size=15)
```

```
plt.ylabel("Count", size=15)
plt.xticks(fontsize = 15)
#ax.set_ylim([0, 10000])
plt.yticks(fontsize = 15)
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Spesific_yield\\countplot_komm
plt.clf()
plt.close()


#%% Removing innstalations based on visual inspection of map placement

#removing innstalation placed in the ocean
pvdata = pvdata[~((pvdata['lat'] == 61.05) & (pvdata['lon'] == 4.17))]

#removing innstalation where there are less than 10 innstalations
#getting first row of each instalation
first_occurrence_data = pvdata.drop_duplicates(subset="key", keep="first")

#finding number of instalation in each Fylke
first_occurrence_data["county_count"] = ...
    first_occurrence_data.groupby("Fylke")["Fylke"].transform("count")

filtered_data = ...
    first_occurrence_data[first_occurrence_data["county_count"] >= 10]

#Removing pv instalations where less than 10 is availabe
filtered_keys = filtered_data['key']
pvdata_filtered = pvdata[pvdata['key'].isin(filtered_keys)]

#saving new df
pvdata_filtered.to_parquet("C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\IFE_Data_13

#%% Ploting distrobution of Fylke

####################################
#################################### Ploting distrobution of Fylke

#Ploting countplot of fylke
plot = pvdata.drop_duplicates(subset=["key", "month"])
plot_top_10_kommunes = plot["Fylke"].value_counts().nlargest(10).index
plot = plot[plot["Fylke"].isin(plot_top_10_kommunes)]
fig, ax = plt.subplots(figsize = (12,12))
sns.countplot(data=plot, x="month", hue= "Fylke")
# Set the x and y labels and the title
plt.title("Number of installations per county", size=20)
plt.legend(title="County", fontsize=12, title_fontsize=15)
plt.xlabel("Month", size=15)
plt.ylabel("Count", size=15)
plt.xticks(fontsize = 15)
#ax.set_ylim([0, 10000])
plt.yticks(fontsize = 15)
plt.savefig("C:\\Users\\marti\\Desktop\\IFE\\Figurer\\Spesific_yield\\countplot_fylk
plt.clf()
plt.close()

unique_table = pvdata.drop_duplicates(subset=["key","month"])
#Group by fylke and month
table = unique_table.groupby(["Fylke", ...
    "month"]).size().reset_index(name="count")

#pivot table to make it easier to read
```

```
table_pivot = table.pivot_table(values="count", index="Fylke", ...
    columns="month")

#Replace NAN with 0
table_pivot = table_pivot.fillna(0)

# Print the table
print(table_pivot)
```

Appendix N

Code: Inference of Tilt and Azimuth for Solcellespesialisten's Data

Parts of this code are from a previous research article [13]. The unaltered code can be found at [77]. The original copyright and license notice [87] is included here:


```
# -*- coding: utf-8 -*-

Created on Thu Apr 10 15:16:14 2023

@author: marti



import pvlib
import matplotlib.pyplot as plt
import pandas as pd
import datetime
import math
import numpy as np
from math import sqrt
import scipy.interpolate
import glob
import os
```

```python
import multiprocessing
import concurrent.futures
from concurrent.futures import ProcessPoolExecutor
from functools import partial
import pytz
import re

import os
import shutil

#defining input and output data
input_file = "C:/Users/marti/Desktop/IFE/pvdata_weather_filtered.parquet"
output_folder = "C:/Users/marti/Desktop/IFE/Data"
"""
# Delete all files in the output folder
for filename in os.listdir(output_folder):
    file_path = os.path.join(output_folder, filename)
    if os.path.isfile(file_path):
        os.unlink(file_path)
"""

#reading file
df = pd.read_parquet(input_file)

# Get unique key values
unique_keys = df["key"].unique()


#saving each PV instalation in a seperate file, to avoid high memory ...
    usage later
for key in unique_keys:
    filtered_df = df[df["key"] == key]
    output_file = os.path.join(output_folder, f"{key}.parquet")
    filtered_df.to_parquet(output_file)




#setting up new input and output folders
source_folder = "C:/Users/marti/Desktop/IFE/Data"
dest_folder = "C:/Users/marti/Desktop/IFE/csv"
new_folder = "C:/Users/marti/Desktop/IFE/Ikke_regnet"

for file in os.listdir(new_folder):
    file_path = os.path.join(new_folder, file)
    try:
        if os.path.isfile(file_path):
            os.unlink(file_path)
    except Exception as e:
        print(f"failed to delete {file_path}. Reason: {e}")

pattern = re.compile(r"^\d+\.parquet$")

xlsx_files = [f for f in os.listdir(source_folder) if pattern.match(f)]
print(f"Filtered xlsx_files: {xlsx_files}")

for xlsx_file in xlsx_files:
    csv_file = xlsx_file.replace(".parquet", ".csv")
    if not os.path.exists(os.path.join(dest_folder, csv_file)):
        print(f"Copying {xlsx_file} to {new_folder}")
        shutil.copy(os.path.join(source_folder, xlsx_file), ...
```

```
            os.path.join(new_folder, xlsx_file))
#saving data
folder_paths = "C:/Users/marti/Desktop/IFE/Ikke_regnet"
file_paths = glob.glob(folder_paths + "/*")
print(f"Files in {folder_paths}: {file_paths}")


#%% Loading data

#input_data = pd.read_excel("C:\\Users\\marti\\OneDrive - ...
    Universitetet i ...
    Agder\\Master-MartinKrebsKristiansen-Solutvikling\\Martin-J5-data\\SolarLog-ACdat
"""
# specify the file
parquet_file = ...
    'C:\\Users\\marti\\Desktop\\IFE\\Sammenslått\\Tilt_azimuth_hourly\\{key}.parquet

# open the file
parquet_table = pq.read_table(parquet_file)

# convert the Parquet table to a df
df = parquet_table.to_pandas()
file_path = "C:/Users/marti/Desktop/IFE/Ikke_regnet/17.parquet"
"""


#%%
#setting up process to run multiple files simultaneously
def process_file(index, file_path):
    #printing file number
    print(f"Processing file: {file_path}")
    pvdata = pd.read_parquet(file_path)


    # selecting latitude
    lat = pvdata.lat[0]

    # selecting longdiude
    lon = pvdata.lon[0]

    input_data = pvdata.copy()



    #remane Pac1 to AC_S45
    input_data = input_data.rename(columns={'acproduction[wh]': 'AC_S45'})


    #input_data["ghi"] = input_data.GHI_Avg
    #input_data["dhi"] = input_data.DHI_Avg
    #input_data["dni"] = input_data.DNI_Avg


    # Calculate solar position.
    solpos = pvlib.solarposition.get_solarposition(input_data.index, ...
        lat, lon)
    input_data['zenith'] = solpos['apparent_zenith']
    input_data['azimuth'] = solpos['azimuth']
    input_data.head(24)


```

```
########################################################################################
########################################################################################

    #selecting irradiance data
    GHI = input_data['ghi'].resample('1D').sum()
    DHI = input_data['dhi'].resample('1D').sum()
    GHIDHI = pd.DataFrame({'daily_ghi' : GHI, 'daily_dhi' : DHI})
    GHIDHI['clear_sky_index'] = DHI / GHI
    GHIDHI.head()


    # Step 2: Pick out clearest day of each month

    # Obtain the date of the monthly clearest days
    GHIDHI['time'] = pd.to_datetime(GHIDHI.index)
    GHIDHI['YYYY'] = GHIDHI['time'].dt.year
    GHIDHI['MM'] = GHIDHI['time'].dt.month
    GHIDHI['DD'] = GHIDHI['time'].dt.day
    GHIDHI_sort = GHIDHI.sort_values(by='clear_sky_index', axis=0, ...
        ascending=True)
    for i in range(1, 13):
        a = GHIDHI_sort.loc[GHIDHI_sort['MM'] == i].head(1)
        locals()['clearest_day_M{}'.format(i)] = a['DD']
        print('The clearest day of month {} is {}.{}'.format(i, i, ...
            a['DD'][0]))

    # Select the input data of the monthly clearest days
    input_data['time'] = pd.to_datetime(input_data.index)
    input_data['MM'] = input_data['time'].dt.month
    input_data['DD'] = input_data['time'].dt.day
    for i in range(1, 13):
        a = locals()['clearest_day_M{}'.format(i)]
        a = a.reset_index(drop=True)
        locals()['M{}'.format(i)] = input_data.loc[(input_data['MM'] ...
            == i) & (input_data['DD'] == a[0])]


#Step 3: Evaluate curve mismatch between normalized plane-of-array ...
    irradiance and PV output


    solar_constant = 1366.1
    method = 'spencer'
    epoch_year = 2022  # year of measurement data (not used)
    model_am = 'kastenyoung1989'
    albedo = 0.2
    surface_type = None
    model = 'perez'
    model_perez = 'allsitescomposite1990'
    for i in range(1, 13):
        print('calculating month {}/12'.format(i))
        monthly_data = locals()['M{}'.format(i)]
        yyyy = monthly_data.time.dt.year[0]
        mm = monthly_data.time.dt.month[0]
        dd = monthly_data.time.dt.day[0]
        day_of_year = datetime.date(yyyy, mm, dd)
        dni_extra = pvlib.irradiance.get_extra_radiation(day_of_year, ...
            solar_constant, method, epoch_year)
        air_mass = ...
            pvlib.atmosphere.get_relative_airmass(monthly_data.zenith, ...
```

```
            model_am)
        air_mass.fillna(0, inplace=True)
        AC_norm = (monthly_data.AC_S45 - monthly_data.AC_S45.min()) / ...
            (monthly_data.AC_S45.max() - monthly_data.AC_S45.min())
        locals()['result_M{}'.format(i)] = []

    # Calulating plane of irradiance for every possible range (0-360 ...
        Azimuth, 0-91 Tilt)
        surface_tilt_list = range(0, 91, 1)
        surface_azimuth_list = range(0, 360, 1)

        for surface_tilt in surface_tilt_list:
            print(f"start { surface_tilt}")
            for surface_azimuth in surface_azimuth_list:

                poa_cal = ...
                    pvlib.irradiance.get_total_irradiance(surface_tilt, ...
                    surface_azimuth, monthly_data.zenith,
                                                  monthly_data.azimuth
                                                      monthly_data.dni
                                                      monthly_data.ghi
                                                  monthly_data.dhi, ..
                                                      dni_extra, ...
                                                      air_mass, ...
                                                      albedo, ...
                                                      surface_type, ..
                                                      model,
                                                  model_perez)
                poa = poa_cal["poa_global"]
                poa_norm = ((poa_cal["poa_global"] - ...
                    poa_cal["poa_global"].min()) /
                            (poa_cal["poa_global"].max() - ...
                                poa_cal["poa_global"].min()))
                error = []
                for j in range(len(poa_norm)):
                    #removing data where solar angle is over 70 degrees
                    if monthly_data.zenith[j] < 70:
                        error.append(AC_norm[j] - poa_norm[j])
                if len(error)>0:
                    squaredError = []
                    absError = []
                    for val in error:
                        squaredError.append(val * val)  # (Error)^2
                        absError.append(abs(val))  # Abs(Error)
                    RMSE = sqrt(np.nansum(squaredError) / ...
                        len(squaredError))  # RMSE
                    MAE = np.nansum(absError) / len(absError)  # MAE
                    dic = {'surface_tilt' : surface_tilt, ...
                        'surface_azimuth' : surface_azimuth, 'RMSE' : ...
                        RMSE, 'MAE' : MAE}
                    locals()['result_M{}'.format(i)].append(dic)
        locals()['result_M{}'.format(i)] = ...
            pd.DataFrame(locals()['result_M{}'.format(i)])
        # Save the DataFrame to a Parquet file
        # Save the DataFrame to a Parquet file
        file_name = os.path.basename(file_path).split('.')[0]
        output_file_path = ...
            f"C:/Users/marti/Desktop/IFE/Results/result_{file_name}_M{i}.parquet"
        locals()['result_M{}'.format(i)].to_parquet(output_file_path)
```

```
    # Step 4: Generate and overlap monthly results
    z_list = [0.5,1,2,3,4,5,6,7,8,9,10,15,20,25,30,35]
    all_results_df = pd.DataFrame(columns = ["z_var", "tilt", "azimuth"])
    folder_name = f"C:/Users/marti/Desktop/IFE/Figure/{file_name}"
    os.makedirs(folder_name)
    for z_var in z_list:
        yearly_result = pd.DataFrame()
        for i in range(1, 13):
            result = locals()['result_M{}'.format(i)]
            if len(result) > 0:
                z = result["RMSE"]
                threshold = np.percentile(z, z_var)  # Calulating ...
                    result of vaying prosentile.

                yearly_result = ...
                    yearly_result.append(result.loc[result["RMSE"] <= ...
                    threshold,["surface_azimuth","surface_tilt"]], ...
                    sort=True)
                yearly_result["point"] = ...
                    yearly_result["surface_azimuth"].map(str) + ',' + ...
                    yearly_result["surface_tilt"].map(str)
        x = yearly_result["point"].tolist()

        # Overlap monthly results.
        yearly_count = []
        for azimuth in range(0,360):
            for tilt in range(0,91):
                count = x.count(str(azimuth)+','+str(tilt))
                dic = ...
                    {'surface_tilt':tilt,'surface_azimuth':azimuth,'count':count}
                yearly_count.append(dic)
        yearly_count = pd.DataFrame(yearly_count)
        yearly_count.sort_values(by='count', axis=0, ascending=False)

        #  Step 5: Obtain the final result of PV orientation estimation

        a = np.radians(yearly_count["surface_azimuth"])
        b = yearly_count["surface_tilt"]
        z = yearly_count["count"]
        xi = np.linspace(a.min(), a.max(), 100)
        yi = np.linspace(b.min(), b.max(), 100)
        theta,r = np.meshgrid(xi, yi)
        zi = scipy.interpolate.griddata((a, b), z, (theta, r), ...
            method='linear')


        fig, ax = plt.subplots(subplot_kw=dict(projection='polar'))
        cset = ax.contourf(theta,r,zi,[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, ...
            11, 12], cmap=plt.cm.jet)
        ax.set_theta_direction(-1)
        ax.set_theta_zero_location('N')
        ax.set_rgrids(np.arange(30, 120, 30))
        ax.set_thetagrids(np.arange(0, 360, 45), ...
            ('N','NE','E','SE','S','SW','W','NW'))
        ax.tick_params(labelsize=20)
        position=fig.add_axes([0.89, 0.1, 0.03, 0.8])
        cb=plt.colorbar(cset,ticks=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, ...
            11, 12], cax=position)
        cb.ax.tick_params(labelsize=20)
        cb.set_label('# of overlaps', rotation=270, fontsize=15)
```

```
        fig.savefig(f"C:/Users/marti/Desktop/IFE/Figure/{file_name}/{file_name}_zvar



        count_max = yearly_count.loc[:, "count"].max()
        overlap_tilt = yearly_count.loc[yearly_count['count'] == ...
            count_max, 'surface_tilt'].mean()
        overlap_azimuth = yearly_count.loc[yearly_count['count'] == ...
            count_max, 'surface_azimuth'].mean()
        print('Final derivation ...
            result:','\n','tilt:',overlap_tilt,'\n','azimuth:',overlap_azimuth)

        results_df = pd.DataFrame({"z_var": [z_var], 'tilt': ...
            [overlap_tilt], 'azimuth': [overlap_azimuth]})
        all_results_df = all_results_df.append(results_df, ...
            ignore_index=True)
    all_results_df.to_csv(f"C:/Users/marti/Desktop/IFE/csv/{file_name}.csv", ...
        index=False)


print(f"Total number of files to process: {len(file_paths)}")

#Initiating multiple files simultaneously
if __name__ == '__main__':
    file_paths = file_paths  # path of files to presess

    with concurrent.futures.ProcessPoolExecutor(max_workers=10) as ...
        executor:
        executor.map(process_file, range(len(file_paths)), file_paths)
```

Appendix O

Code: RANSAC and Clustering

```python

# -*- coding: utf-8 -*-
"""
Created on Tue Apr 18 11:16:58 2023

@author: marti
"""

import pandas as pd
import os
import pvlib
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import glob
from scipy.signal import argrelextrema
import matplotlib.ticker as ticker
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import RANSACRegressor
from sklearn.model_selection import RandomizedSearchCV


#%% Load data and merge

# Read the parquet file
parquet_file = ...
    'C:/Users/marti/Desktop/IFE/pvdata_weather_filtered.parquet'
pvdata = pd.read_parquet(parquet_file)

# Get the list of files in the folder
csv_folder = 'C:/Users/marti/OneDrive/Dokumenter/Master/IFE/csv'

#loading file based on the key
for unique_key in pvdata['key'].unique():
    key = int(unique_key)
    print(key)
    file_name = f'{key}.csv'
    csv_path = os.path.join(csv_folder, file_name)

    if os.path.exists(csv_path):
        #loading csv file
        csv_df = pd.read_csv(csv_path)

        #add information from the csv to the parquet
        for col in csv_df.columns:
```

```
            if col not in pvdata.columns:
                pvdata[col] = None
            #adding inforamtion
            pvdata.loc[pvdata['key'] == key, col] = csv_df.at[11, col]

# Remove tilt or azimuth values which are NAN
pvdata = pvdata.dropna(subset=['tilt', 'azimuth'])

#saving files
output_directory = 'C:/Users/marti/Desktop/IFE/orientation_sammenslått'

# Deleting files for next run
files = glob.glob(os.path.join(output_directory, '*'))
for f in files:
    os.remove(f)

#group by key
grouped_data = pvdata.groupby('key')

for key, group in grouped_data:
    output_file_path = os.path.join(output_directory, f"{key}.parquet")
    group.to_parquet(output_file_path)


#saving file
#pvdata.to_parquet('C:/Users/marti/Desktop/IFE/orientation_sammenslått_kombined/all_

#Debug line
#file_path = ...
   "C:\\Users\\marti\\Desktop\\IFE\\orientation_sammenslått\\11.parquet"

#%% Applying filter by cluster

# itterate over evey csv file
for file_name in os.listdir(output_directory):
    file_path = os.path.join(output_directory, file_name)
    file_name = file_name.replace('.csv', '')
    print(file_name)

    # read file
    input_data = pd.read_parquet(file_path)

    #solar variables
    solar_constant = 1366.1
    method = 'spencer'
    model_am = 'kastenyoung1989'
    albedo = 0.2
    surface_type = None
    model = 'perez'
    model_perez = 'allsitescomposite1990'

    lat = input_data["lat"][0]
    lon = input_data["lon"][0]

    surface_tilt = input_data["tilt"][0]
    surface_azimuth = input_data["azimuth"][0]


    # Calculate solar position.
    solpos = pvlib.solarposition.get_solarposition(input_data.index, ...
       lat, lon)
```

```
    input_data['zenith'] = solpos['apparent_zenith']
    input_data['azimuth'] = solpos['azimuth']
    input_data.head(24)


    day_of_year = input_data.index.to_series().dt.dayofyear

    dni_extra = pvlib.irradiance.get_extra_radiation(day_of_year, ...
        solar_constant, method)
    air_mass = ...
        pvlib.atmosphere.get_relative_airmass(input_data.zenith, model_am)
    air_mass.fillna(0, inplace=True)


    poa_cal = pvlib.irradiance.get_total_irradiance(surface_tilt, ...
        surface_azimuth, input_data.zenith,
                                                    input_data.azimuth, ...
                                                        input_data.dni, ...
                                                        input_data.ghi,
                                                    input_data.dhi, ...
                                                        dni_extra, ...
                                                        air_mass, albedo, ...
                                                        surface_type, model,
                                                    model_perez)




    input_data = pd.merge(input_data, poa_cal, left_index=True, ...
        right_index=True, how='inner')

    #%% Calculating input data

    input_data["yf"] =  (input_data["acproduction[wh]"]/1000) / ...
        input_data.capacity_kwp

    input_data = input_data.loc[input_data['yf'] != 0]

    input_data["yr"] = input_data.poa_global / 1000

    input_data["pr"] = input_data["yf"] / input_data["yr"]

   #debugplot
   # sns.scatterplot(data=input_data, x="yr", y="yf")


    #%% error

    input_data["error"] = input_data["yf"] - input_data["yr"]

    #%% step 1. inliers using Ran-Sa_c


    #creating RANSAG regressor
    ransac = RANSACRegressor()

    #parameter for the grid search
    param_grid = {
        'min_samples': list(range(10, 150)),
        'max_trials': [100, 200, 300, 500, 700, 1000, 1500],
        'residual_threshold': np.arange(0.07, 0.15, 0.01),
```

```
        'loss': ['absolute_error'],
    }

    # executing GridSearchCV
    #grid_search = GridSearchCV(ransac, param_grid, ... scoring='neg_mean_squared_error', cv=5, n_jobs=-1)
    random_search = RandomizedSearchCV(ransac, param_grid, ... scoring='neg_mean_squared_error', n_iter=150, cv=5, n_jobs=-1, ... random_state=42)


    input_data = input_data.dropna(subset=['poa_global'])
    x = input_data["yr"].values.reshape(-1, 1)
    y = input_data["yf"].values.reshape(-1, 1)

    # fit the x,y coordinates
    #grid_search.fit(x, y)
    random_search.fit(x, y)

    #loacate best fit
    best_ransac = random_search.best_estimator_

    # print best parameters
    print("Best hyperparameters:", random_search.best_params_)

    # print slope and intercept
    print('Intercept:', best_ransac.estimator_.intercept_)
    print('Slope:', best_ransac.estimator_.coef_)

    # locate innlier data
    inlier_mask = best_ransac.inlier_mask_

    input_data['inlier_ransac'] = inlier_mask

    #making grid of input values
    x_grid = np.linspace(x.min(), x.max(), 100).reshape(-1, 1)

    # predicting output values
    y_pred = best_ransac.predict(x_grid)

    #plot input data points, and the RANSAC regression
    fig, ax = plt.subplots(figsize = (12,12))
    plt.scatter(x[inlier_mask], y[inlier_mask], color='blue', ... label='Inliers')
    plt.scatter(x[~inlier_mask], y[~inlier_mask], color='red', ... label='Outliers')
    best_params = random_search.best_params_
    line_label = f"RANSAC regression\nmin_samples: ... {best_params['min_samples']}\nmax_trials: ... {best_params['max_trials']}\nresidual_threshold: ... {best_params['residual_threshold']}\nloss: {best_params['loss']}"

    plt.plot(x_grid, y_pred, color='green', linewidth=2, label=line_label)
    plt.legend(fontsize=25, title_fontsize=25)
    plt.xlabel("Yr", size=25)
    plt.ylabel("yf", size=25)
    plt.xticks(fontsize = 25)
    #ax.set_ylim([0, 350])
    #ax.set_xlim([0, 1])
    plt.yticks(fontsize = 25)
```

```
    plt.savefig(f"C:\\Users\\marti\\Desktop\\filtered non ...
       zero\\Figure\\{file_name}_RANSAC_.png", bbox_inches="tight")
    plt.clf()
    plt.close()


    #%% Step 2.polynomial regression
    #this step only uses the inlier data from step 1


    # copying inliers
    inliers = input_data[inlier_mask]

    #setting number of bins
    num_bins = 10
    inliers['error_bins'] = pd.qcut(inliers['yr'], q=num_bins, ...
       labels=False, precision=0)

    def optimize_polyfit(x, y, max_degree=10):
        min_mse = float('inf')
        best_degree = 1
        best_coeffs = None

        for degree in range(1, max_degree+1):
            coeffs = np.polyfit(x, y, degree)
            poly_func = np.poly1d(coeffs)

            y_pred = poly_func(x)
            mse = mean_squared_error(y, y_pred)

            if mse < min_mse:
                min_mse = mse
                best_degree = degree
                best_coeffs = coeffs

        return best_coeffs

    from scipy.optimize import root_scalar

    #finding where the polynomal line crosses 0
    def find_zero_crossing(poly_func, min_x, max_x):
        if np.sign(poly_func(min_x)) * np.sign(poly_func(max_x)) > 0:
            return None
        zero_crossing = root_scalar(poly_func, method='brentq', ...
           bracket=[min_x, max_x])
        if zero_crossing.converged:
            return zero_crossing.root
        return None

    #fit and plot histograms
    def fit_poly_and_plot_hist(data, **kwargs):
        ax = plt.gca()
        sns.histplot(data=data, x='error', bins=50, ax=ax)
        counts, bin_edges = np.histogram(data['error'], bins=50)
        bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2
        best_coeffs = optimize_polyfit(bin_centers, counts)
        poly_func = np.poly1d(best_coeffs)

        #find local maxima and minima
        local_maxima = argrelextrema(poly_func(bin_centers), np.greater)
        local_minima = argrelextrema(poly_func(bin_centers), np.less)
```

```

        #find global maximum
        global_maximum_index = np.argmax(poly_func(bin_centers))
        global_maximum = bin_centers[global_maximum_index], ...
           poly_func(bin_centers[global_maximum_index])

        #find the local minima with largest difference in y-value
        left_minima = None
        right_minima = None
        max_diff = float('-inf')

        local_minima_y = poly_func(bin_centers[local_minima])

        #selecting minima
        for i in range(len(local_minima_y) - 1):
            diff = local_minima_y[i + 1] - local_minima_y[i]
            if diff > max_diff:
                max_diff = diff
                minima = bin_centers[local_minima][i], local_minima_y[i]
                next_minima = bin_centers[local_minima][i + 1], ...
                   local_minima_y[i + 1]

                if minima[0] < global_maximum[0]:
                    left_minima = minima
                else:
                    right_minima = minima
                if next_minima[0] < global_maximum[0]:
                    left_minima = next_minima
                else:
                    right_minima = next_minima

        if left_minima is None:
            left_minima = bin_centers.min(), poly_func(bin_centers.min())
        if right_minima is None:
            right_minima = bin_centers.max(), poly_func(bin_centers.max())

        #Plot the polynomial curve
        x_plot = np.linspace(bin_centers.min(), bin_centers.max(), 100)
        y_plot = poly_func(x_plot)
        ax.plot(x_plot, y_plot, '-', color="red", linewidth=3)

        #plot global maximum and the local minima point
        ax.plot(*global_maximum, 'go', markersize=10, color="green")
        ax.plot(*left_minima, 'bo', markersize=10, color="red")
        ax.plot(*right_minima, 'bo', markersize=10, color="red")

        return global_maximum[0], left_minima[0], right_minima[0]


    g = sns.FacetGrid(inliers, col='error_bins', col_wrap=3, ...
       sharex=False, sharey=False, height=4)

    #Plot histograms, fit the polynomial regression
    g.map_dataframe(fit_poly_and_plot_hist)

    g.set_axis_labels("Error", "Count", fontsize=25)
    g.set_titles("Bin {col_name}", fontsize=25)
    for axes in g.axes.flat:
        axes.tick_params(axis='both', labelsize=15)
        axes.xaxis.set_major_locator(ticker.MaxNLocator(3))

```

```
#save
plt.savefig(f"C:\\Users\\marti\\Desktop\\filtered non ...
    zero\\Figure\\{file_name}FacetGrid_histogram.png", ...
    bbox_inches="tight")

# removefig
plt.clf()
plt.close()




#%% Step 3. Group threshold
global_maxima_x = []
left_minima_x = []
right_minima_x = []
yr_values = []
mid_yr_values = []

#plot individual histograms
def store_and_plot(data, **kwargs):
    g_max_x, l_min_x, r_min_x = fit_poly_and_plot_hist(data, **kwargs)
    global_maxima_x.append(g_max_x)
    left_minima_x.append(l_min_x)
    right_minima_x.append(r_min_x)
    return data['yr'].mean()

g = sns.FacetGrid(inliers, col='error_bins', col_wrap=3, ...
    sharex=False, sharey=False)
g.map_dataframe(lambda data, **kwargs: ...
    yr_values.append(store_and_plot(data, **kwargs)))

for idx in range(len(yr_values) - 1):
    mid_yr = (yr_values[idx] + yr_values[idx + 1]) / 2
    mid_yr_values.append(mid_yr)

last_mid_yr = yr_values[-1] + (mid_yr_values[-1] - mid_yr_values[-2])
mid_yr_values.append(last_mid_yr)


#create stair-like coordinates
def create_stair_x_coordinates(x_values, max_x):
    stair_x_values = []
    for idx in range(len(x_values) - 1):
        stair_x_values.extend([x_values[idx], x_values[idx + 1]])
    stair_x_values.extend([x_values[-1], max_x])
    return stair_x_values

def create_stair_y_coordinates(y_values):
    stair_y_values = []
    for idx in range(len(y_values) - 1):
        stair_y_values.extend([y_values[idx], y_values[idx]])
    stair_y_values.extend([y_values[-1], y_values[-1]])
    return stair_y_values

####################
#################### Creating poly y value
# Fit 3rd-degree polynomials
global_poly_coeff = np.polyfit(mid_yr_values, global_maxima_x, 3)
left_poly_coeff = np.polyfit(mid_yr_values, left_minima_x, 3)
right_poly_coeff = np.polyfit(mid_yr_values, right_minima_x, 3)
```

```

    # Create polynomial functions
    global_poly_func = np.poly1d(global_poly_coeff)
    left_poly_func = np.poly1d(left_poly_coeff)
    right_poly_func = np.poly1d(right_poly_coeff)

    plt.xlim(min(yr_values), max(inliers['yr']))

    x_poly = np.linspace(min(mid_yr_values), mid_yr_values[-1], 100)

    max_x = max(inliers['yr'])

    #plot
    plt.figure(figsize=(12, 12))
    plt.plot(yr_values, global_maxima_x, 'g-', label='Global Maxima')
    plt.plot(create_stair_x_coordinates(yr_values, max_x), create_stair_y_coordinates(left_minima_x), 'b--', label='Left Minima')
    plt.plot(create_stair_x_coordinates(yr_values, max_x), create_stair_y_coordinates(right_minima_x), 'r--', label='Right Minima')

    #plot the polynomial functions
    plt.plot(x_poly, global_poly_func(x_poly), 'g:', label='Global Maxima Poly')
    plt.plot(x_poly, left_poly_func(x_poly), 'b:', label='Left Minima Poly')
    plt.plot(x_poly, right_poly_func(x_poly), 'r:', label='Right Minima Poly')


    plt.tick_params(axis='both', labelsize=25)
    plt.xlabel('Yr', fontsize=25)
    plt.ylabel('Error Value', fontsize=25)
    plt.legend( fontsize=25, title_fontsize=25)


    plt.savefig(f"C:\\Users\\marti\\Desktop\\filtered non zero\\Figure\\{file_name}Polyline.png", bbox_inches="tight")

    # To show the plot
    plt.clf()
    plt.close()


    #%% Flipping the curve

    global_y = global_poly_func(x_poly) + x_poly
    left_poly_y = left_poly_func(x_poly) + x_poly
    right_poly_y = right_poly_func(x_poly) + x_poly

    """#debug plot
    plt.figure()
    plt.plot(x_poly, global_y, 'g', label='Global Maxima Poly')
    plt.plot(x_poly, left_poly_y, 'b', label='Left Minima Poly')
    plt.plot(x_poly, right_poly_y, 'r', label='Right Minima Poly')

    sns.scatterplot(data=input_data, x="yr", y="yf")
    plt.xlabel('Yr', fontsize=25)
    plt.ylabel('Yf', fontsize=25)
    plt.show()
```

```
    """
    #%% Selecting inliers

    inlier_poly = []


    # finding inliers
    for index, row in input_data.iterrows():
        yf = row["yf"]
        yr = row["yr"]
        left_threshold = left_poly_func(yr) + yr
        right_threshold = right_poly_func(yr) + yr

        inlier_poly.append(left_threshold <= yf <= right_threshold)

    #creating new column with inlier information True/False
    input_data["inlier_poly"] = inlier_poly

    #Scatterplot
    plt.figure(figsize=(12, 12))
    sns.scatterplot(data=input_data, x="yr", y="yf", ...
       hue="inlier_poly", palette=['red', 'blue'], legend=False)

    plt.plot(x_poly, global_y, 'g', label='Global Maxima Poly')
    plt.plot(x_poly, left_poly_y, 'b', label='Left Minima Poly')
    plt.plot(x_poly, right_poly_y, 'r', label='Right Minima Poly')

    plt.tick_params(axis='both', labelsize=25)
    plt.xlabel('Yr', fontsize=25)
    plt.ylabel('Yf', fontsize=25)
    plt.legend( fontsize=25, title_fontsize=25)
    plt.savefig(f"C:\\Users\\marti\\Desktop\\filtered non ...
       zero\\Figure\\{file_name}_FacetGrid_histogram.png", ...
       bbox_inches="tight")

    plt.clf()
    plt.close()


    output_path = f"C:/Users/marti/Desktop/filtered non ...
       zero/data/{file_name}"
    #saving information to file
    input_data.to_parquet(output_path)
```

Bibliography

[1] M. Buvik, J. Cabrol, D. Spilde, E. Skaansar, A. Roos, and Å. Grytli Tveten, *Norsk og nordisk effektbalanse fram mot 2030.* [Online]. Available: https://publikasjoner.nve.no/rapport/2022/rapport2022_20.pdf (Accessed: 8-5-2023).

[2] Solcellespesialisten, *PROSJEKTER | Solcellespesialisten.* [Online]. Available: https://www.solcellespesialisten.no/prosjekter (Accessed: 11-5-2023).

[3] K. Mertens, *Photovoltaics: Fundamentals, technology, and Practice*, 2nd ed. Hoboken, NJ: Wiley, 2019, ISBN: 9781119401049.

[4] D. Atsu, I. Seres, and I. Farkas, "The state of solar PV and performance analysis of different PV technologies grid-connected installations in Hungary," *Renewable and Sustainable Energy Reviews*, vol. 141, p. 110 808, May 2021, ISSN: 1364-0321. DOI: 10.1016/J.RSER.2021.110808.

[5] M. B. Øgaard, Å. Skomedal, and J. H. Selj, "PERFORMANCE EVALUATION OF MONITORING ALGORITHMS FOR PHOTOVOLTAIC SYSTEMS,"

[6] C. A. M. Service, *CAMS solar radiation time-series: data documentation - Copernicus Knowledge Base - ECMWF Confluence Wiki.* [Online]. Available: https://confluence.ecmwf.int/display/CKB/CAMS+solar+radiation+time-series%3A+data+documentation (Accessed: 24-4-2023).

[7] C. A. M. Service, *CAMS solar radiation time-series.* [Online]. Available: https://ads.atmosphere.copernicus.eu/cdsapp#!/dataset/cams-solar-radiation-timeseries?tab=overview (Accessed: 11-5-2023).

[8] "Copernicus Atmosphere Monitoring Service User Guide to the CAMS Radiation Service (CRS)," DOI: 10.5194/amt-6-2403-2013.

[9] Z. Qu, A. Oumbe, P. Blanc, *et al.*, "Fast radiative transfer parameterisation for assessing the surface solar irradiance: The Heliosat-4 method," *Meteorologische Zeitschrift*, vol. 26, no. 1, pp. 33–57, Feb. 2017, ISSN: 16101227. DOI: 10.1127/METZ/2016/0781.

[10] pvlib, *pvlib.iotools.get_ cams — pvlib python 0.9.4 documentation.* [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.iotools.get_cams.html#id8 (Accessed: 6-3-2023).

[11] A. R. Gonçalves, A. T. Assireu, F. R. Martins, *et al.*, "Enhancement of Cloudless Skies Frequency over a Large Tropical Reservoir in Brazil," *Remote Sensing 2020, Vol. 12, Page 2793*, vol. 12, no. 17, p. 2793, Aug. 2020, ISSN: 2072-4292. DOI: 10.3390/RS12172793. [Online]. Available: https://www.mdpi.com/2072-4292/12/17/2793/htm%20https://www.mdpi.com/2072-4292/12/17/2793.

[12] R. Mondragón, J. Alonso-Montesinos, D. Riveros-Rosas, *et al.*, "Attenuation Factor Estimation of Direct Normal Irradiance Combining Sky Camera Images and Mathematical Models in an Inter-Tropical Area," *Remote Sensing 2020, Vol. 12, Page 1212*, vol. 12, no. 7, p. 1212, Apr. 2020, ISSN: 2072-4292. DOI: 10.3390/RS12071212. [Online]. Available: https://www.mdpi.com/2072-4292/12/7/1212/htm%20https://www.mdpi.com/2072-4292/12/7/1212.

[13] B. Meng, R. C. Loonen, and J. L. Hensen, "Data-driven inference of unknown tilt and azimuth of distributed PV systems," *Solar Energy*, vol. 211, pp. 418–432, Nov. 2020, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2020.09.077.

[14] pvlib, *pvlib.irradiance.get_total_irradiance — pvlib python 0.9.5 documentation.* [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.irradiance.get_total_irradiance.html (Accessed: 23-4-2023).

[15] R. Perez, P. Ineichen, R. Seals, J. Michalsky, and R. Stewart, "Modeling daylight availability and irradiance components from direct and global irradiance," *Solar Energy*, vol. 44, no. 5, pp. 271–289, Jan. 1990, ISSN: 0038-092X. DOI: 10.1016/0038-092X(90)90055-H.

[16] P. G. Loutzenhiser, H. Manz, C. Felsmann, P. A. Strachan, T. Frank, and G. M. Maxwell, "Empirical validation of models to compute solar irradiance on inclined surfaces for building energy simulation," *Solar Energy*, vol. 81, no. 2, pp. 254–267, Feb. 2007, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2006.03.009.

[17] J. D. Foley, M. A. Fischler, and R. C. Bolles, "Graphics and Image Processing Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography," 1981.

[18] S. Choi, T. Kim, and W. Yu, "Performance Evaluation of RANSAC Family," DOI: 10.5244/C.23.81.

[19] T. Opsahl, *Lecture 3.3 Robust estimation with RANSAC.* [Online]. Available: https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v17/forelesninger/lecture_3_3_robust_estimation_with_ransac.pdf.

[20] R. Turner, A. Samaranayaka, and C. Cameron, "Parametric vs nonparametric statistical methods: which is better, and why?,"

[21] J. P. Verma, "Data analysis in management with SPSS software," *Data Analysis in Management with SPSS Software*, pp. 1–481, Dec. 2013. DOI: 10.1007/978-81-322-0786-3/COVER.

[22] StatsDirect, *One Way Analysis of Variance (ANOVA) - StatsDirect.* [Online]. Available: https://www.statsdirect.com/help/analysis_of_variance/one_way.htm (Accessed: 10-5-2023).

[23] J. Taylor, J. Leloux, L. M. H. Hall, A. M. Everard, J. Briggs, and A. Buckley, "PERFORMANCE OF DISTRIBUTED PV IN THE UK: A STATISTICAL ANALYSIS OF OVER 7000 SYSTEMS,"

[24] A. Nanda, D. B. B. Mohapatra, A. P. K. Mahapatra, A. P. K. Mahapatra, and A. P. K. Mahapatra, "Multiple comparison test by Tukey's honestly significant difference (HSD): Do the confident level control type I error," *International Journal of Statistics and Applied Mathematics*, vol. 6, no. 1, pp. 59–65, Jan. 2021, ISSN: 24561452. DOI: 10.22271/MATHS.2021.V6.I1A.636.

[25] L. Statistics, *Kruskal-Wallis H Test in SPSS Statistics | Procedure, output and interpretation of the output using a relevant example.* [Online]. Available: https://statistics.laerd.com/spss-tutorials/kruskal-wallis-h-test-using-spss-statistics.php (Accessed: 6-5-2023).

[26] E. G. M. Hui, "Learn R for applied statistics: With data visualizations, regressions, and statistics," *Learn R for Applied Statistics: With Data Visualizations, Regressions, and Statistics*, pp. 1–243, Jan. 2018. DOI: 10.1007/978-1-4842-4200-1.

[27] M. Rouncefield, "Combinations, Probabilities and Sample Size. Investigations into the Mann-Whitney (/test," *TEACHING MATHEMATICS AND ITS APPLICATIONS*, vol. 17, no. 4, 1998. [Online]. Available: https://academic.oup.com/teamat/article/17/4/159/1707396.

[28] A. Dinno, "Nonparametric pairwise multiple comparisons in independent groups using Dunn's test," *Stata Journal*, vol. 15, no. 1, pp. 292–300, Apr. 2015, ISSN: 15368734. DOI: 10.1177/1536867X1501500117.

[29] D. Palejev and M. Savov, "On the Convergence of the Benjamini–Hochberg Procedure," *Mathematics 2021, Vol. 9, Page 2154*, vol. 9, no. 17, p. 2154, Sep. 2021, ISSN: 2227-7390. DOI: 10.3390/MATH9172154. [Online]. Available: https://www.mdpi.com/2227-7390/9/17/2154/htm%20https://www.mdpi.com/2227-7390/9/17/2154.

[30] O. M. Midtgard, T. O. Sætre, G. Yordanov, A. G. Imenes, and C. L. Nge, "A qualitative examination of performance and energy yield of photovoltaic modules in southern Norway," *Renewable Energy*, vol. 35, no. 6, pp. 1266–1274, Jun. 2010, ISSN: 0960-1481. DOI: 10.1016/J.RENENE.2009.12.002.

[31] M. S. Adaramola and E. E. Vågnes, "Preliminary assessment of a small-scale rooftop PV-grid tied in Norwegian climatic conditions," *Energy Conversion and Management*, vol. 90, pp. 458–465, Jan. 2015, ISSN: 0196-8904. DOI: 10.1016/J.ENCONMAN.2014.11.028.

[32] A. Ameur, A. Berrada, K. Loudiyi, and M. Aggour, "Forecast modeling and performance assessment of solar PV systems," *Journal of Cleaner Production*, vol. 267, p. 122 167, Sep. 2020, ISSN: 0959-6526. DOI: 10.1016/J.JCLEPRO.2020.122167.

[33] A. Fezzani, I. Hadj-Mahammed, A. Kouzou, *et al.*, "Energy Efficiency of Multi-Technology PV Modules under Real Outdoor Conditions—An Experimental Assessment in Ghardaïa, Algeria," *Sustainability 2022, Vol. 14, Page 1771*, vol. 14, no. 3, p. 1771, Feb. 2022, ISSN: 2071-1050. DOI: 10.3390/SU14031771. [Online]. Available: https://www.mdpi.com/2071-1050/14/3/1771/htm%20https://www.mdpi.com/2071-1050/14/3/1771.

[34] J. Ascencio-Vásquez, J. C. Osorio-Aravena, K. Brecl, E. Muñoz-Cerón, and M. Topič, "Typical Daily Profiles, a novel approach for photovoltaics performance assessment: Case study on large-scale systems in Chile," *Solar Energy*, vol. 225, pp. 357–374, Sep. 2021, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2021.07.007.

[35] S. Tamrakar, M. Mustafa, and R. Riise, "Feasibility study for utilization of solar energy in the arctic areas," *IOP Conference Series: Materials Science and Engineering*, vol. 700, no. 1, Nov. 2019, ISSN: 1757899X. DOI: 10.1088/1757-899X/700/1/012066. [Online]. Available: https://www.researchgate.net/publication/337543181_Feasibility_study_for_utilization_of_solar_energy_in_the_arctic_areas.

[36] T. Haumann, "A Brief Look at the Performance of PV in Norway," 2016.

[37] Solargis, *Solar resource maps and GIS data for 200+ countries | Solargis*. [Online]. Available: https://solargis.com/maps-and-gis-data/download/norway (Accessed: 3-3-2023).

[38] A. G. Imenes, H. G. Beyer, K. Boysen, J. O. Odden, and R. E. Grundt, "Performance of grid-connected PV system in Southern Norway," *2015 IEEE 42nd Photovoltaic Specialist Conference, PVSC 2015*, Dec. 2015. DOI: 10.1109/PVSC.2015.7355823.

[39] A. G. Imenes, "Performance of BIPV and BAPV installations in Norway," *Conference Record of the IEEE Photovoltaic Specialists Conference*, vol. 2016-November, pp. 3147–3152, Nov. 2016, ISSN: 01608371. DOI: 10.1109/PVSC.2016.7750246.

[40] S. Lindig, J. Ascencio-Vasquez, J. Leloux, D. Moser, and A. Reinders, "Performance Analysis and Degradation of a Large Fleet of PV Systems," *IEEE Journal of Photovoltaics*, vol. 11, no. 5, pp. 1312–1318, Sep. 2021, ISSN: 21563403. DOI: 10.1109/JPHOTOV.2021.3093049.

[41] S. Lindig, D. Moser, A. J. Curran, *et al.*, "International collaboration framework for the calculation of performance loss rates: Data quality, benchmarks, and trends (towards a uniform methodology)," *Progress in Photovoltaics: Research and Applications*, vol. 29, no. 6, pp. 573–602, Jun. 2021, ISSN: 1099-159X. DOI: 10.1002/PIP.3397. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/pip.3397%20https://onlinelibrary.wiley.com/doi/abs/10.1002/pip.3397%20https://onlinelibrary.wiley.com/doi/10.1002/pip.3397.

[42] Y. Zhao, B. Lehman, R. Ball, J. Mosesian, and J. F. De Palma, "Outlier detection rules for fault detection in solar photovoltaic arrays," *Conference Proceedings - IEEE Applied Power Electronics Conference and Exposition - APEC*, pp. 2913–2920, 2013. DOI: 10.1109/APEC.2013.6520712.

[43] O. Tsafarakis, K. Sinapis, and W. G. Van Sark, "PV System Performance Evaluation by Clustering Production Data to Normal and Non-Normal Operation," *Energies 2018, Vol. 11, Page 977*, vol. 11, no. 4, p. 977, Apr. 2018, ISSN: 1996-1073. DOI: 10.3390/EN11040977. [Online]. Available: https://www.mdpi.com/1996-1073/11/4/977/htm%20https://www.mdpi.com/1996-1073/11/4/977.

[44] S. Lindig, A. Louwen, D. Moser, and M. Topic, "Outdoor PV System Monitoring—Input Data Quality, Data Imputation and Filtering Approaches," *Energies 2020, Vol. 13, Page 5099*, vol. 13, no. 19, p. 5099, Sep. 2020, ISSN: 1996-1073. DOI: 10.3390/EN13195099. [Online]. Available: https://www.mdpi.com/1996-1073/13/19/5099/htm%20https://www.mdpi.com/1996-1073/13/19/5099.

[45] M. G. Deceglie, L. Micheli, and M. Muller, "Quantifying Soiling Loss Directly from PV Yield," *IEEE Journal of Photovoltaics*, vol. 8, no. 2, pp. 547–551, Mar. 2018, ISSN: 21563381. DOI: 10.1109/JPHOTOV.2017.2784682.

[46] J. K. Selj, E. S. Marstein, Å. Skomedal, *et al.*, "General, Robust and Scalable Methods for String Level Monitoring in Utility Scale PV Systems Characterization of light induced degradation View project GENERAL, ROBUST AND SCALABLE METHODS FOR STRING LEVEL MONITORING IN UTILITY SCALE PV SYSTEMS," DOI: 10.4229/EUPVSEC20192019-5BO.5.4. [Online]. Available: https://www.researchgate.net/publication/337113457.

[47] M. Deceglie, E. S. Marstein, Å. Skomedal, H. Haug, and E. S. Marstein, "Iterative and Self-Consistent Estimation of Degradation and Soiling Loss in PV Systems-a Case Study," DOI: 10.4229/EUPVSEC20202020-5DO.1.3. [Online]. Available: https://www.researchgate.net/publication/344930548.

[48] pvlib, *pvlib.location.Location.get_ clearsky — pvlib python 0.9.5 documentation.* [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.location.Location.get_clearsky.html#pvlib.location.Location.get_clearsky (Accessed: 9-4-2023).

[49] RdTools, *Degradation and soiling example with clearsky workflow — RdTools 2.1.4+0.g996f843.dirty documentation.* [Online]. Available: https://rdtools.readthedocs.io/en/stable/examples/degradation_and_soiling_example_pvdaq_4.html?highlight=clear#Clear-sky-workflow (Accessed: 15-4-2023).

[50] J. Leloux, L. Narvarte, A. Desportes, and D. Trebosc, "Performance to Peers (P2P): A benchmark approach to fault detections applied to photovoltaic system fleets," *Solar Energy*, vol. 202, pp. 522–539, May 2020, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2020.03.015.

[51] F. Harrou, A. Dairi, B. Taghezouit, and Y. Sun, "An unsupervised monitoring procedure for detecting anomalies in photovoltaic systems using a one-class Support Vector Machine," *Solar Energy*, vol. 179, pp. 48–58, Feb. 2019, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2018.12.045.

[52] T. Hu, M. Zheng, J. Tan, L. Zhu, and W. Miao, "Intelligent photovoltaic monitoring based on solar irradiance big data and wireless sensor networks," *Ad Hoc Networks*, vol. 35, pp. 127–136, Dec. 2015, ISSN: 1570-8705. DOI: 10.1016/J.ADHOC.2015.07.004.

[53] J. D. De Guia, R. S. Concepcion, H. A. Calinao, S. C. Lauguico, E. P. Dadios, and R. R. P. Vicerra, "Application of Ensemble Learning with Mean Shift Clustering for Output Profile Classification and Anomaly Detection in Energy Production of Grid-Tied Photovoltaic System," *ICITEE 2020 - Proceedings of the 12th International Conference on Information Technology and Electrical Engineering*, pp. 286–291, Oct. 2020. DOI: 10.1109/ICITEE49829.2020.9271699.

[54] S. Ferlito, S. De Vito, and G. Di Francia, "Detect Anomalies in Photovoltaic Systems Using Isolation Forest (Preliminary Results)," *Lecture Notes in Electrical Engineering*, vol. 753, pp. 231–238, 2021, ISSN: 18761119. DOI: 10.1007/978-3-030-69551-4{_}31/TABLES/1. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-69551-4_31.

[55] S. Killinger, D. Lingfors, Y. M. Saint-Drenan, *et al.*, "On the search for representative characteristics of PV systems: Data collection and analysis of PV system azimuth, tilt, capacity, yield and shading," *Solar Energy*, vol. 173, pp. 1087–1106, Oct. 2018, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2018.08.051.

[56] G. Heilscher, F. Meier, C. Hoyer-Klick, S. Lanig, D. Stetter, and H. Ruf, "Active Grid Planning Based on Solar Power Roof Potential Analysis," *27th European Photovoltaic Solar Energy Conference and Exhibition*, no. 3-936338-28-0, pp. 3782–3787, Oct. 2012. DOI: 10.4229/27THEUPVSEC2012-5CO.7.2. [Online]. Available: http://www.eupvsec-proceedings.com/proceedings?paper=15654.

[57] D. Palmer, I. Cole, T. Betts, and R. Gottschalg, "Assessment of potential for photovoltaic roof installations by extraction of roof tilt from light detection and ranging data and aggregation to census geography," *IET Renewable Power Generation*, vol. 10, no. 4, pp. 467–473, Apr. 2016, ISSN: 1752-1424. DOI: 10.1049/IET-RPG.2015.0388. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1049/iet-rpg.2015.0388%20https://onlinelibrary.wiley.com/doi/abs/10.1049/iet-rpg.2015.0388%20https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-rpg.2015.0388.

[58] S. Killinger, N. Engerer, and B. Müller, "QCPV: A quality control algorithm for distributed photovoltaic array power output," *Solar Energy*, vol. 143, pp. 120–131, Feb. 2017, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2016.12.053.

[59] Y. M. Saint-Drenan, S. Bofinger, R. Fritz, S. Vogt, G. H. Good, and J. Dobschinski, "An empirical approach to parameterizing photovoltaic plants for power forecasting and simulation," *Solar Energy*, vol. 120, pp. 479–493, Oct. 2015, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2015.07.024.

[60] Z. Qu, A. Oumbe, P. Blanc, *et al.*, "Fast radiative transfer parameterisation for assessing the surface solar irradiance: The Heliosat-4 method," *Meteorologische Zeitschrift*, vol. 26, no. 1, pp. 33–57, Feb. 2017, ISSN: 16101227. DOI: 10.1127/METZ/2016/0781.

[61] C. A. M. Service, *CAMS solar radiation time-series: data documentation - Copernicus Knowledge Base - ECMWF Confluence Wiki*. [Online]. Available: https://confluence.ecmwf.int/display/CKB/CAMS+solar+radiation+time-series%3A+data+documentation (Accessed: 24-4-2023).

[62] Copernicus, *About Copernicus | Copernicus*. [Online]. Available: https://www.copernicus.eu/en/about-copernicus (Accessed: 15-4-2023).

[63] G. Buster, M. Bannister, A. Habte, *et al.*, "Physics-guided machine learning for improved accuracy of the National Solar Radiation Database," *Solar Energy*, vol. 232, pp. 483–492, Jan. 2022, ISSN: 0038-092X. DOI: 10.1016/J.SOLENER.2022.01.004.

[64] E. S. Hub, *PVGIS typical meteorological year (TMY) generator*. [Online]. Available: https://joint-research-centre.ec.europa.eu/pvgis-online-tool/pvgis-tools/pvgis-typical-meteorological-year-tmy-generator_en (Accessed: 15-4-2023).

[65] pvlib, *pvlib.iotools.get_pvgis_hourly — pvlib python 0.9.5 documentation*. [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.iotools.get_pvgis_hourly.html (Accessed: 15-4-2023).

[66] Z. Qu, A. Oumbe, P. Blanc, *et al.*, "Fast radiative transfer parameterisation for assessing the surface solar irradiance: The Heliosat-4 method," *Meteorologische Zeitschrift*, vol. 26, no. 1, pp. 33–57, 2017, ISSN: 16101227. DOI: 10.1127/METZ/2016/0781.

[67] "Copernicus Atmosphere Monitoring Service Regular Validation Report Issue #38 M-A-M 2022 CAMS2-73 Solar radiation products,"

[68] A. Thampi, *GitHub - thampiman/reverse-geocoder: A fast, offline reverse geocoder in Python*. [Online]. Available: https://github.com/thampiman/reverse-geocoder (Accessed: 11-5-2023).

[69] scikit-learn developers, *Solar Position Algorithm (SPA)*. [Online]. Available: https://midcdmz.nrel.gov/spa/ (Accessed: 1-5-2023).

[70] I. Reda and A. Andreas, "Solar Position Algorithm for Solar Radiation Applications," 2003. [Online]. Available: http://www.osti.gov/bridge.

[71] S. Bishop, *pytz - World Timezone Definitions for Python — pytz 2014.10 documentation*. [Online]. Available: https://pythonhosted.org/pytz/ (Accessed: 18-4-2023).

[72] time and date, *timeanddate.com*. [Online]. Available: https://www.timeanddate.com/ (Accessed: 16-4-2023).

[73] pvlib, *pvlib.solarposition.get_ solarposition — pvlib python 0.9.5 documentation*. [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.solarposition.get_solarposition.html (Accessed: 15-4-2023).

[74] B. Meng, R. C. Loonen, and J. L. Hensen, "Data-driven inference of unknown tilt and azimuth of distributed PV systems," *Solar Energy*, vol. 211, pp. 418–432, Nov. 2020, ISSN: 0038092X. DOI: 10.1016/J.SOLENER.2020.09.077. [Online]. Available: https://gitlab.tue.nl/bp-tue/inference-of-unknown-tilt-and-azimuth.

[75] pvlib, *pvlib.irradiance.get_ extra_ radiation — pvlib python 0.9.5 documentation*. [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.irradiance.get_extra_radiation.html (Accessed: 20-4-2023).

[76] pvlib, *pvlib.atmosphere.get_ relative_ airmass — pvlib python 0.9.5 documentation*. [Online]. Available: https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.atmosphere.get_relative_airmass.html (Accessed: 12-4-2023).

[77] B. Meng, *example.ipynb · master · BP-TUe / Inference of unknown tilt and azimuth · GitLab*. [Online]. Available: https://gitlab.tue.nl/bp-tue/inference-of-unknown-tilt-and-azimuth/-/blob/master/example.ipynb (Accessed: 5-4-2023).

[78] Python, *concurrent.futures — Launching parallel tasks — Python 3.11.3 documentation*. [Online]. Available: https://docs.python.org/3/library/concurrent.futures.html (Accessed: 16-4-2023).

[79] scikit-learn developers, *sklearn.linear_ model.RANSACRegressor — scikit-learn 1.2.2 documentation*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RANSACRegressor.html (Accessed: 16-4-2023).

[80] scikit-learn developers, *sklearn.model_ selection.GridSearchCV — scikit-learn 1.2.2 documentation*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (Accessed: 20-4-2023).

[81] N. Developers, *numpy.polyfit — NumPy v1.24 Manual*. [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html.

[82] S. community, *scipy.signal.argrelextrema — SciPy v1.10.1 Manual*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.argrelextrema.html (Accessed: 16-4-2023).

[83] A. Desportes, D. Trebosc, L. Narvarte Fernández, R. Moreton Villagrá, J. Taylor, and J. Leloux, "Monitoring 30,000 PV Systems in Europe: Performance, Faults, and State of the Art," *31st European Photovoltaic Solar Energy Conference and Exhibition*, vol. 1, pp. 1574–1582, Nov. 2015. DOI: 10.4229/EUPVSEC20152015-5AO.8.1. [Online]. Available: http://www.eupvsec-proceedings.com/proceedings?paper=34857.

[84] U. of Agder, "Collection of unpublished images," Available from the University of Agder (UiA), 2023.

[85] Geonorge, *Norge, Illustrasjonskart - Kartkatalogen*. [Online]. Available: https://kartkatalog.geonorge.no/metadata/norge-illustrasjonskart/a374f867-60c0-4524-9eda-b15ab4d12858 (Accessed: 14-5-2023).

[86] H. te Heesen and V. Herbort, "Development of an algorithm to analyze the yield of photovoltaic systems," *Renewable Energy*, vol. 87, pp. 1016–1022, Mar. 2016, ISSN: 0960-1481. DOI: 10.1016/J.RENENE.2015.07.058.

[87] *LICENSE · master · BP-TUe / Inference of unknown tilt and azimuth · GitLab*. [Online]. Available: https://gitlab.tue.nl/bp-tue/inference-of-unknown-tilt-and-azimuth/-/blob/master/LICENSE.