UiA University of Agder

Investigation of Outdoor Performance of Silicon Photovoltaic Modules at Different Installation Angles

SIGRID LANGHOLM LARSEN

SUPERVISOR
Anne Gerd Imenes

University of Agder, 2021
Faculty of Engineering and Science
Department of Engineering Sciences

Abstract

The objective of this master thesis is to analyse and compare the performance of two different PV technologies mounted at different tilt angles and orientations. The PV modules are installed on the roof of the J5 building in the University of Agder, located in Grimstad, Norway. The analysis was carried out by comparing 13 months of measured data (from March 2020 to March 2021). The technologies assessed are mono crystalline silicon and poly crystalline silicon, mounted at a tilt angle of 10° East/West, 45° South and 90° South. Data recorded include global and diffuse horizontal irradiance, irradiance in the plane-of-array and other weather parameters, PV module temperature, DC output power, current and voltage for the PV modules. To assess the performance, performance ratio, temperature corrected performance ratio and specific yield has been calculated for all modules on a monthly basis. Results give the DC specific yield for 10° tilt modules between 920 - 984 kWh/kWp. The two modules mounted on the 45° tilt rack and the BIPV modules all had low data availability due to missing data points. To account for this data are estimated for the missing data point using linear regression. The DC specific yield before estimating data for missing data points for 45° are 988 kWh/kWp (mono c-Si 45°), 1050 kWh/kWp (poly c-Si 45°), 762 kWh/kWp (BIPV normal), and 760 kWh/kWp (BIPV grey). Comparing the two PV technologies at different tilt angles and orientations, there is not detected a substantial difference in performance between the East- and West-facing modules at a 10° tilt. The PR for the higher tilted modules are on average slightly better than for the 10° tilt modules. The best indication of module performance in the various installation angles are when estimating output power for missing data. From these results the PR_{DC} values of the 10° tilt modules are in the range 0.88 - 0.90 (mono c-Si E/W) and 0.88-0.92 (poly c-Si W/E), while for the 45° tilt modules the values are 0.92 (mono c-Si) and 0.95 (poly c-Si), and for both the normal and grey BIPV modules at 90° tilt the value is 0.94.

PVsyst software is used to simulate five mono c-Si modules at 10° tilt and East-facing. This is because in the system at UiA five modules are connected in a string to one inverter. Design parameters such as module tilt, orientation, shading and inter-row spacing is defined to obtain as accurate simulation as possible. The results show a good correlation between simulation and calculated results. AC energy output for the simulation was 1436 kWh/year, compared to DC output for the real system at 1451 kWh/year. PVsyst simulation can be used to investigate other installation angles and orientations than the ones analysed in this work.

Preface

This master's thesis finalizes my engineering masters degree in Renewable Energy from the University of Agder (UiA). Previous to this two-year master's degree, I finished a bachelor's degree in Renewable Energy from UiA.

The motivation for this master's thesis is the constantly growing utilization of solar power in Norway. I had my first course in solar energy during my master's which sparked an interest in this clean energy utilization. Writing this thesis has been a challenging and educational experience, both professional and personal.

I would like to extend my gratitude to my supervisor, Anne Gerd Imenes, who introduced me to this thesis. She provided data an gave me the theoretical platform to produce the necessary results. Her guidance and assistance have been of invaluable help. Lastly, I would also like to thank my friends and family for support and encouragement along the way.

Sigrid L. Larsen

Sigrid Langholm Larsen

Grimstad, May 2021

Mandatory declaration

1.	I/We hereby declare that my/our report is my/our own work and that I/We have not used any other sources or have received any other help than mentioned in the thesis.	☒
2.	I/we further declare that this thesis: - has not been used for an other exam at another department/university/university college in Norway of abroad;	☒
3.	I/we am/are aware that violation of the above is regarded as cheating and may result in cancellation of exams and exclusion from universities and colleges in Norway, see Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	☒
4.	I/we am/are aware that all submitted theses may be checked for plagiarism.	☒
5.	I/we am/are aware that the University of Agder will deal with all cases where there is suspicion of cheating according to the university's guidelines for dealing with cases of cheating.	☒
6.	I/we have incorporated the rules and guidelines in the use of sources and references on the library's web pages.	☒

Publishing Agreement

Authorization for electronic publishing of the thesis.
Author(s) have copyrights of the thesis. This means, among other
things, the exclusive right to make the work available to the general public (Åndsverkloven. §2).
All theses that fulfill the criteria will be registered and published
in Brage Aura and on UiA's web pages with author's approval.
Theses that are not public or are confidential will not be published.

I hereby give the University of Agder a free right to
make the task available for electronic publishing: ☒YES □ NO

Is the thesis confidential? □YES ☒NO
(confidential agreement must be completed and signed by the Head of the Department)
- If yes:
Can the thesis be published when the confidentiality period is over? □YES □NO

Is the task except for public disclosure? □ YES ☒NO
(contains confidential information. see Offl. §13/Fvl. §13)

Contents

List of Figures

List of Tables

Nomenclature

α	Absorption coefficient	
ΔT_{cnd}	Conduction temperature drop	
η	Efficiency	[%]
A	Area	$[m^2]$
a	Empirical constant reflecting the increase of module temperature with sunlight	
AC	Alternating current	A
AM	Air mass	
AOI	Angle of incident	
b	Empirical constant reflecting the effect of wind speed on the module temperature	[s/m]
$BAPV$	Building Applied Photovoltaic	
$BIPV$	Building Integrated Photovoltaics	
$c-Si$	crystalline silicon	
DC	Direct current	A
E	Irradiance	$[W/m^2]$
E_A	Energy output DC	[W]
E_F	Net energy output AC	[W]
E_{NOCT}	Irradiance at nominal operating cell temperature	$[800W/m^2]$
E_{POA}	Irradiance in Plane-of-Array	$[W/m^2]$

E_{STC}	Irradiance at standard test conditions	$[1000W/m^2]$
FF	Fill factor	
h	Hour	
H_G	Radiation energy	$[kWh/m^2]$
I_D	Diode current	[A]
I_S	Saturation current	[A]
I_{MPP}	Current at maximum power point	[A]
I_{ph}	Photocurrent	[A]
I_{SC}	Short circuit current	[A]
k	Boltzmann‘s constant	$[6.63 \cdot 10^{-34} J/K]$
L_C	Generator loss	
L_S	System loss	
m	Ideality factor	
MPP	Maximum Power Point	[W]
$NOCT$	Nominal Operating Cell Temperature	
P_{STC}	Power at standard test conditions	[W]
P_{sun}	Constant irradiance from the Sun	$3.845 \cdot 10^{2}6$ W
$Pmpp$	Peak power	[W]
POA	Plane-of-array	
PR	Performance ratio	
PV	Photovoltaic	

q	Elementary charge	$[1.6 \cdot 10^{-19} As]$
SF	Spectral factor	
SR	Spectral response	
T	Temperature	$[°C]$
T_a	Ambient temperature	$[°C]$
T_C	PV cell temperature	$[°C]$
T_m	Module back surface temperature	$[°C]$
$T_{modules}$	Module temperature at standard test conditions	$[25\ °C]$
T_{Tcorr}	Temperature corrected performance ratio	W
TC	Temperature coefficient	$[\%/°C]$
U_0	Constant heat transfer component	$[W/m^2K]$
U_1	Convective heat transfer component	$[W/m^3sk]$
V	Voltage	[V]
V_T	Thermal voltage	[V]
V_{MPP}	Voltage at maximum power point	[V]
V_{OC}	Open circuit voltage	[V]
WS	Wind speed	[m/s]
Y_A	Generation yield	$[Wh/W_p]$
Y_F	Final yield	$[Wh/W_p]$
Y_R	Reference yield	

1 Introduction

Solar energy is today the most rapidly growing renewable energy source, closely followed by wind energy [1]. An increasing number of countries have introduced solar power as a means to reduce greenhouse gasses and reach the United Nations Goal 7 "Ensure access to affordable, reliable, sustainable and modern energy for all" [2]. During the last few years, the photovoltaic (PV) market in Norway has also grown substantially. From 2017 - 2018 the amount of energy produced by solar power increased by 29 % [3]. With this growth comes a need for further research and knowledge on the performance of PV systems under Nordic conditions. A common misunderstanding is that the Nordic climate is not suited for solar energy, as there is not as many sunny days as further South in Europe. However, PV systems have been used for many years in off-grid systems used in cottages around the country and recently also increasingly in larger grid-connected systems. In fact, the Nordic climate is well suited for photovoltaic use, with good sun conditions and colder temperatures [4]. The interest in PV installation in Norway is rapidly growing in both private homes and the industrial sector.

Crystalline silicon solar cells have been the most dominant technology in the PV market since it's invention in the 1950's [5]. During the last decade, the crystalline silicon (c-Si) cells has accounted for 80 - 90 % [5] of the market share for photovoltaics, and it is expected to continue in this lead. SUSOLTECH (Sustainable Solar Cell Technology) is a Norwegian research centre working to increase efficiency and decrease cost of crystalline silicon solar cells [6]. University of Agder (UiA) has joined the leading Norwegian research groups to cooperate in their work [7]. The objective of the UiA project is to research performance of silicon based PV modules in Norway, using monitoring data from an outdoor installation in Grimstad.

The performance of a photovoltaic module is highly dependent on location and environmental factors, as well as installation configuration. The amount of energy produced by the system is directly linked to the amount of sun that reaches the panel. To obtain the best performance in terms of the highest annual energy yield, PV modules must be oriented and tilted at an optimum angle for the location. However, PV systems may also be optimized for the best cost-efficiency, such as in East/West configurations, or for building integrated photovoltaics (BIPV). The impact on the monthly variation in energy yield and the performance of these installation configurations will be further investigated in this thesis.

1.1 Problem Statement

This master thesis aims to compare the performance of two types of different silicon-based photovoltaic modules mounted at different tilt angles and orientations.

The specific objectives of this master thesis are:

- To analyse the monitoring data and calculate specific yield and performance ratio for the PV technologies in the different installation configurations (i) 10° East/West (flat roof); (ii) 45° South (free rack), (iii) 90° South (BIPV).
- To investigate the effect temperature have on the performance of PV technologies at different tilt angles and orientations.
- To use a software tool to predict performance of the PV modules and compare the results with measured data.

This master thesis is written as a further work for ENE503 Energy Research project "Investigation of temperature influence on performance of different PV technologies under Nordic conditions" [8] written by the author as a part of the master's degree in Renewable Energy at UiA.

1.2 Key Assumptions and Limitations

Some key assumptions and limitations have been set for this master thesis. Not all aspects of PV performance and installations are considered in this thesis, therefore only the key parts of the PV system are described in chapter 2 *Theory*. The data used for analyses are collected from March 2020 to March 2021. This was found to be the time period with overlapping data for all PV systems. For the 45° and 90° modules there were periods with missing data due to downtime for the inverter or monitoring system. For these time periods the power data was estimated using linear regression based on the measured irradiance.

The performance analysis conducted in this thesis are based on DC values. The 10° East/West DC data are measured using optimizers, while the 45° and 90° tilt modules are from current voltage (IV) curve monitoring, where the value is kept at maximum power point (P_{MPP}) between IV-sweeps.

Another assumption made is assuming that the module temperature measured is representative for each module in the PV-array. The temperature may vary between modules within the same installation, and between the top and bottom of the same module. The deviance is dependent on the installation type, but is typically between 3 - 5 K [9]. The temperature i measured at the back of one East-facing module and one West-facing module for each PV system at 10° tilt. For the 45° tilt module the temperature is measured at the back of each different PV technology. Lastly, for the 90° tilt modules the temperature is measured at the top and bottom of each module, where the module temperature used in this thesis is a mean value between the two measurements.

1.3 Structure of the Thesis

Chapter 1 An introduction to the thesis is given in chapter 1. Here the problem statement is given, and the key assumptions and limitations are addressed.

Chapter 2 Chapter 2 presents a theoretical background to PV technology, solar radiation, and to the PV analysis conducted in the thesis.

Chapter 3 A literature review, giving a summary of former work conducted on the topic of PV performance is given in chapter 3. This chapter builds on chapter 2, further elaborating on the subject of the problem statement.

Chapter 4 Chapter 4 gives an overview of the case study at the University of Agder. It also describes the method for the performance analysis in this thesis.

Chapter 5 Chapter 6 describes the simulation software PVsyst, and how this software is set up to simulate parts of the system in the case study.

Chapter 6 In chapter 5 the results of the performance analysis and PVsyst simulations are presented, described and discussed. The results are mainly presented in plots.

Chapter 7 The thesis is concluded in chapter 7, and a short suggestion for further work is given.

2 Theory

This chapter presents some theoretical background of photovoltaic systems, solar radiation, and methods of PV system analysis. Large parts of the theory chapter is based on the previous work done by the author in ENE503 Energy Research Project at UiA [8]. However, all external sources are specified.

2.1 PV Technology Types

There are several PV technologies on the market, however, in this report only mono- and poly-crystalline silicon will be considered. Also a description of building integrated photovoltaics will be included.

Figure 2.1 shows an overview of the efficiency development for crystalline silicon cells [10]. The chart shows the highest confirmed conversion efficiency for research cells dating from 1976 to 2020. The most recent world record is marked with a flag at the right side of the chart displaying the efficiency and the symbol for the technology. The record for mono- and poly crystalline silicon is 26.1 % and 23.3 %, respectively. [10] [8]

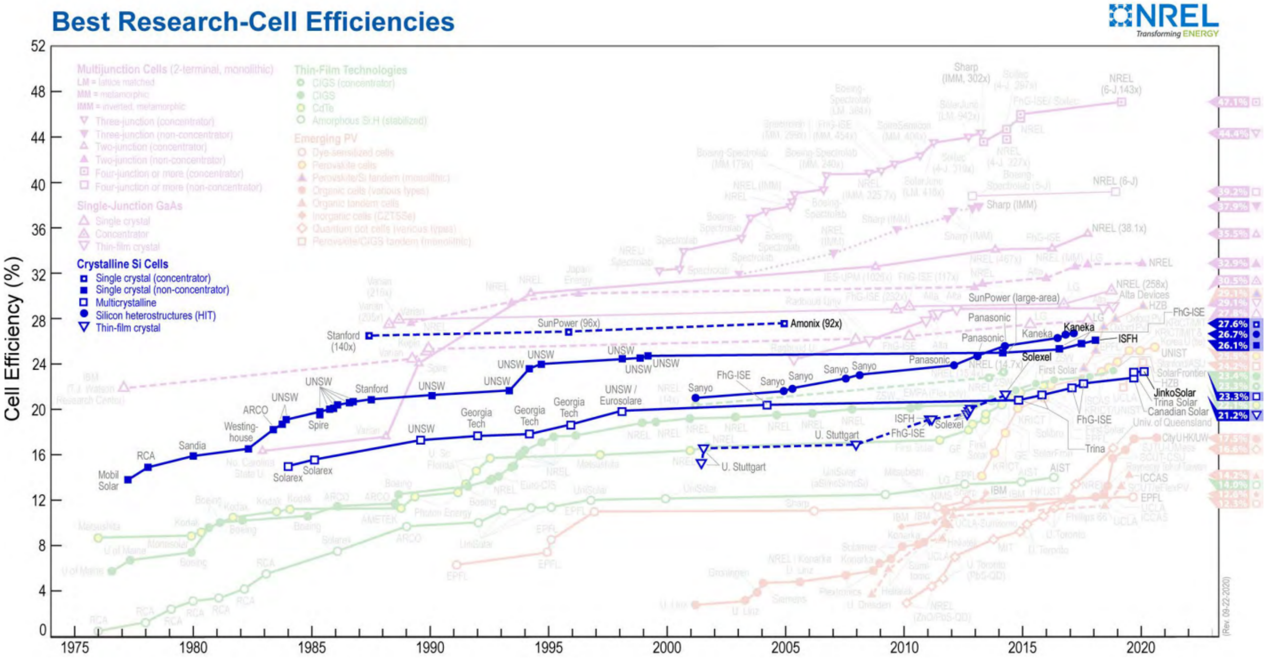


Figure 2.1: Efficiency chart for crystalline silicon cells [10].

2.1.1 Mono Crystalline Silicon

Mono crystalline silicon cells, or mono c-Si, are the most common type of solar cell used today. The cells consist, as the name implies, of a single crystal of silicon [11]. The mono c-Si modules are the PV technology with the highest efficiency, however, this results in a higher cost [12]. Usually, the efficiency is between 17-22 % [12] in commercial use, however, an efficiency of 25 % has been obtained in the lab [13].

2.1.2 Poly Crystalline Silicon

Poly crystalline silicon cells, or poly c-Si, are produced in the same way as mono c-Si cells, however, in a poly c-Si cell the silicon is not a single crystal, hence the name poly. The wafers in a poly c-Si cell are made by melting several fragments of silicon together [14, p 103]. Poly c-Si cells may therefore also be referred to as "multi-crystalline" silicon cells [15]. The efficiency of poly c-Si cells is between 15 % - 17% for commercial use [12]. Due to the many crystals in each cell, the cells have a lot of grain boundaries, and therefore less freedom for the electrons to move. This is the reason for the lower efficiency for poly c-Si cells compared to mono c-Si [15]. An advantage, however, is that poly c-Si cells are cheaper than mono c-Si cells.

Figure 2.2 shows a mono c-Si and a poly c-Si cell side by side. Here the difference in material is very clear, with the multi-crystal silicon in the poly c-Si cell compared to the single crystal in the mono c-Si cell.

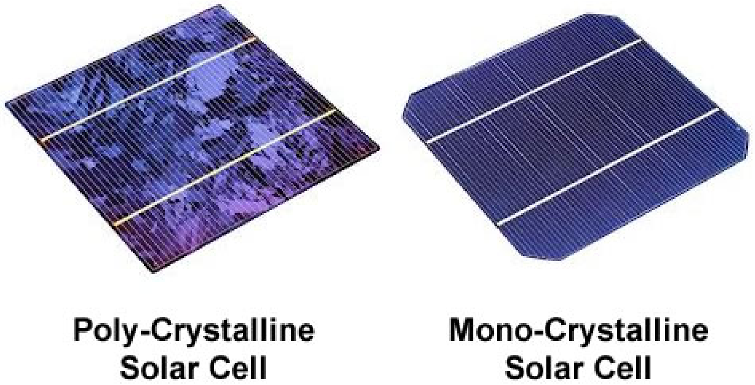


Figure 2.2: Comparison of poly c-Si (left) and mono c-Si (right) cells side by side [16].

2.1.3 Building Integrated Photovoltaics

Building integrated photovoltaic or BIPV is when photovoltaic materials are used to replace conventional building materials in parts of a building structure. This is in contrast to building applied photovoltaics (BAPV), which is when photovoltaic modules are added on the building and do not have any direct effect on the building structure. BIPV systems may be part of the roof, skylights, or facade. The building sector account for approximately 39 % of the global energy consumption [17]. BIPV systems may therefore be a great way to reduce the amount of needed energy from the grid. BIPV systems integrated into the facade are gaining increasing interest. This is because the facade surfaces usually are larger than the roof, in addition, facade systems may provide a more even energy production throughout the day, due to different facade orientations [18].

A common challenge with BIPV systems is elevated temperature of the modules. To solve this, the modules are ventilated at the back. The outdoor air enters the system at the bottom and exits at the top. The air absorbs the heat from the modules to increase efficiency and lifetime. Some applications utilize the heated air by directing it into the building to reduce the heating load. These systems are called BIPVT systems (Building integrated photovoltaic and thermal) [19].

2.2 PV System

2.2.1 PV Module

Multiple solar cells interconnected in the same plane, and encapsulated to protect the cells from the environment, constitute a photovoltaic module or panel. The typical voltage of a single solar cell is 0.6 V. By connecting them in series a higher voltage can be obtained while the current remains the same (see chapter 2.2.2).

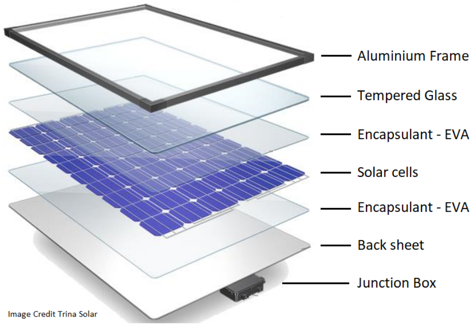


Figure 2.3: Structure of a PV module [20].

Depending on the desired power output of the module, the number of single PV cells wired into an array varies. A typical number of cells in a PV module is 36, 48, 60 or 72 [14, p 142]. The cells in a module are typically series connected to obtain the required output voltage [14, p 141]. Several PV modules are interconnected to form a PV system. Figure 2.3 shows an overview of the typical components of a PV module structure. The module is usually covered by an aluminum frame with a glass top and a polymer backsheet. For some PV modules, both the front and the back surfaces are made of glass, for instance BIPV modules.

2.2.2 PV Strings

There are two different ways to string (or wire) PV modules together, in series or parallel. By connecting the modules in series, the voltage increases with the number of panels, but the current stays the same [14]. When stinging in series, a wire from the positive terminal of one module is connected to a wire from the negative terminal of the next module, and so on. One drawback of connecting PV modules in series is that a shaded module will decrease the current throughout the entire string (see chapter 2.4) [21]. Figure 2.4 illustrates how the modules are series-connected.

The second way of connecting PV panels is in parallel. When connecting in parallel, all the positive terminals are connected on one string, and the negative terminals are connected on another [21]. This is illustrated in figure 2.5. Using this method, the current increases with each additional panel, however, the voltage remains the same. Thus, if one module is shaded, the other panels in the string may operate as normal [21].

A typical way of wiring PV modules is by adding a fixed number of modules in a string to get the desired output power and then connecting several strings in parallel [21].

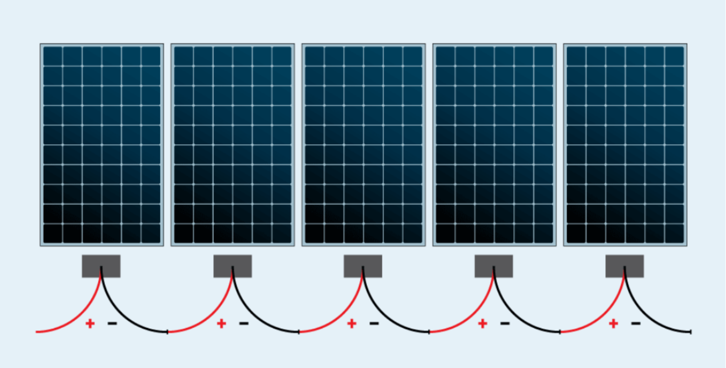


Figure 2.4: When connecting PV modules in series, the positive terminal of one module is connected to the negative terminal of the next module [22].

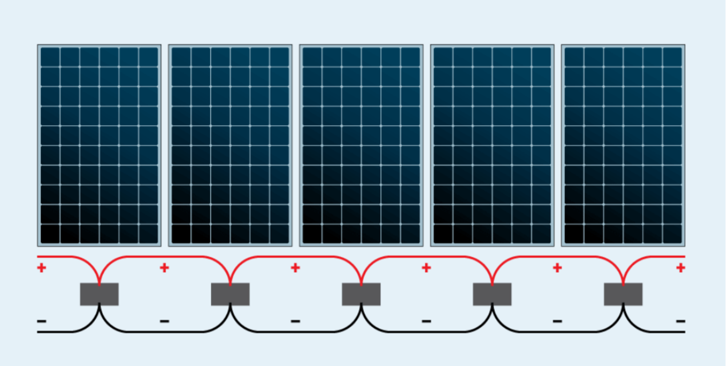


Figure 2.5: When connecting PV modules in parallel, all positive terminals are connected on one string, and the negative terminals are connected on another [22].

PV modules produce direct current (DC), while the current inserted into the grid is alternating current (AC). A DC-AC conversion with an inverter is therefore necessary in a PV system. The inverter also has other tasks than to convert the power for DC-AC. The inverter is responsible for feeding the current into the grid with the same frequency as the grid (50Hz in Norway). The output power of the PV module is constantly changing with the fluctuating sun radiation and temperature. The inverter is responsible for always operating the module in the maximum power point, or MPP [14]. The efficiency of the inverter is a measure of how well it inverts the DC current to AC current, and it is defined as:

$$\eta_{inv} = \frac{P_{AC}}{P_{DC}} \tag{2.1}$$

were P_{AC} is the output power and P_{DC} is the input power. The inverter should achieve high efficiency, above 95 % [14].

Another way for MPP tracking is by DC/DC optimizers, which are installed on each PV module individually. They are DC to DC converters, that ensures the maximum output for each PV module before feeding the power into the inverter. They also allow you to register data from the modules [23].

2.3 Characteristic Performance of a Solar Cell

The performance of a solar cell is characterized by the peak power (P_{mpp}), short circuit current (I_{sc}), open-circuit voltage (V_{oc}) and fill factor (FF). These can be determined by analyzing the -V curve of the solar cell, given in the right side of figure 2.6.

The maximum power point is the operating point on the I-V curve at which the maximum power is produced. The voltage and current associated with the MPP are called V_{MPP} and I_{MPP}. The equation for the maximum power point is given in eq. (2.2) [14, p 82].

$$MPP = I_{MPP} \cdot V_{MPP} \tag{2.2}$$

Fill factor is another parameter used to determine the maximum power of the solar cell and is a measurement of the quality of the cell. FF is defined as the ratio between the maximum power to the product of V_{OC} and I_{SC}, as shown in eq. (2.3) [24]. This ratio is visualized in the right side of figure 2.6. Graphically the fill factor is the area under the *MPP* working point. Typical values for silicon cells are between 0.75 and 0.85. [14, p 83]

$$FF = \frac{V_{MPP} \cdot I_{MPP}}{V_{OC} \cdot I_{SC}} \tag{2.3}$$

The curve in the first quadrant (the positive x-y direction), in figure 2.6 (left side) is considered as the I-V curve of the solar cell (right side) [14, p 80-81]. The area below the I-V curve shows the maximum power the solar cell can produce with the maximum current and voltage. The right side of figure 2.6 shows the I-V curve in more detail.

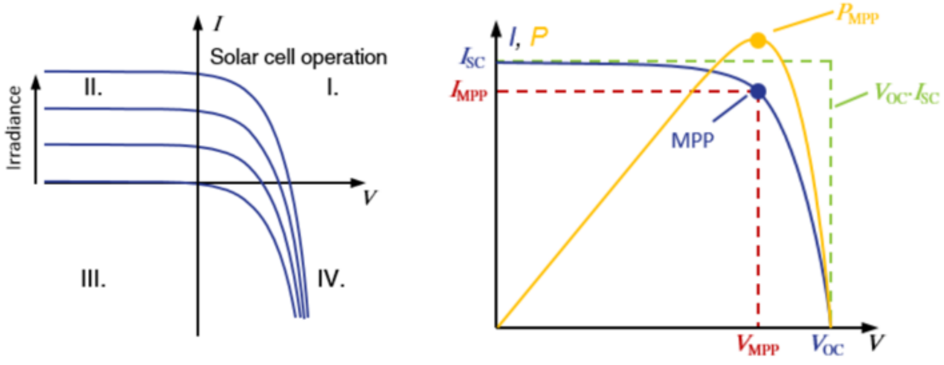


Figure 2.6: (On the left) The characteristic curve for a solar cell, (on the right) IV-curve for a solar cell [14, p 81, 83].

Figure 2.7 shows the electrical equivalent circuit for a solar cell to schematically represent the behavior of a real, illuminated, solar cell. This is the standard model, also called the single-diode model. The solar cell is represented by a current source in parallel with a diode, representing the p-n junction. The p-n junction is a boundary between the two semiconductor materials. It separates the negatively charged electrons from the positively charged holes, and only allow flow in one direction [14]. The resistances R_{sh} and R_s describe the ohmic losses and losses due to leakage currents [14].

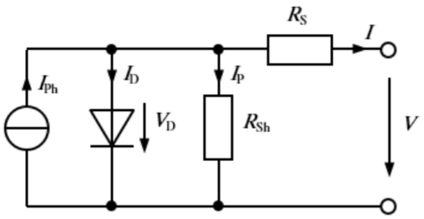


Figure 2.7: Equivalent circuit for electrical description of solar modules [14].

The single-diode model can be represented by eq. (2.4) [14, p 81] where I_{ph} is the photocurrent [A], I_D is the diode current [A], I_S is the dark saturation current [A], V is the voltage [V], m is the ideal factor (usually between 1 and 2), and V_T is the thermal voltage [V]. These terms are described below.

$$I = I_{Ph} - I_D = I_{Ph} - I_S \cdot (e^{\frac{V}{m \cdot V_T}} - 1) \tag{2.4}$$

The photocurrent is directly proportional to the short circuit current, I_{SC}. The short circuit current is the maximum current obtained when the voltage, V, is zero (hence zero resistance, no voltage drop across an external load). This current is directly proportional to the irradiance [14, p 81-82]. The short circuit current is given in eq. (2.5).

$$I_{SC} = I_{Ph} - I_S \cdot (e^0 - 1) = I_{Ph} \tag{2.5}$$

The dark saturation current (I_S) is the very small current that flows through a solar cell even when no light, or no photons, has entered. It is a measurement of recombination in a photovoltaic cell. [25]

The open-circuit voltage (V_{OC}) is the maximum voltage at zero current (I = 0, no current can flow in an open circuit), and is given in eq. (2.6) [26] The open-circuit voltage varies with the natural logarithm of the irradiance [14, p 82].

$$V_{OC} = m \cdot V_T \cdot ln\left(\frac{I_{SC}}{I_S} + 1\right) \tag{2.6}$$

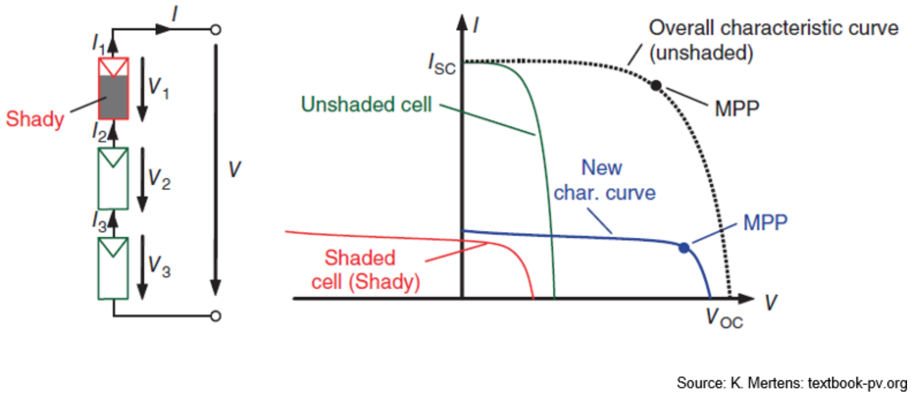
The thermal voltage (V_T) describes the voltage produced within the p-n junction due to temperature in the PV cell. It can be determined by the following eq. (2.7):

$$V_T = \frac{k \cdot T}{q} \tag{2.7}$$

where k is the Boltzmann constant ($8.6175 \cdot 10^-5\, eV K^{-1}$), T is the temperature [K], and q is the electronic charge ($1.6 \cdot 10^-19 A\, s$) [14].

2.4 Shading

Shading of PV modules may cause great problems. Shading of just one cell can reduce the power generated, for all cells in the substring, to zero [27]. When a solar cell is shaded, the current in the string of cells in the PV module reduces to the current level of the shaded cell. The effect of shading is determined by how the panels are wired. Figure 2.8 shows the effect of partial shading in series connection. Because the cells are wired in series, the current must go through the shaded cell. The current has to be the same overall cells, thus, it is reduced to the lowest I_{SC}. For cells wired in parallel, shading does not have as large effects. [14] [27]



Source: K. Mertens: textbook-pv.org

Figure 2.8: Partial shading of a solar cell with series connection (left), and the effect of shading on the power generation (right) [14, p 142].

Shading, in addition to reducing the power generated by the module, may lead to damage of the cell or the encapsulation of the module. The unshaded cells trying to push power through the shaded cell cause massive heating of the shaded cell, also called a hot spot [14]. The shaded cell becomes reversed bias and behave as a resistance. To prevent hot spots, bypass diodes may be used. A bypass diode is a diode coupled in parallel, but with opposite polarity to the solar cell (see figure 2.9). The bypass diode has a threshold voltage of 0.7 V [14]. If the solar cell is shaded, the current can flow in the external circuit, and the cell will not over-heat. [28]

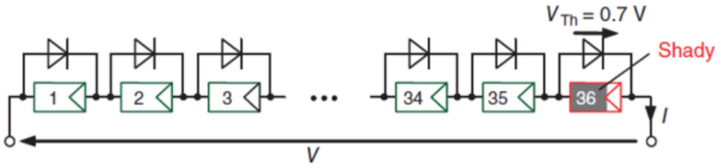


Figure 2.9: Bypass diodes connected in parallel with the solar cells [14].

2.5 Performance Evaluations

2.5.1 Standard Test Conditions (STC)

Standard test conditions are the conditions for which all solar modules are tested. By using a fixed set of conditions solar modules of different types and manufacturers can be compared and rated against each other. The capacity of the solar modules measured under these conditions is the rated power, or nominal power, of the module [14, p 15] [29]. STC is defined as follows:

- Irradiance: E=E_{STC}=1000 W/m^2
- Temperature of the solar module: $T_{cell} = 25°C$
- Standard light spectrum: AM 1.5 (see chapter 2.8.2)

2.5.2 Efficiency

A PV module's efficiency is a measurement of the module's ability to convert sunlight into electric energy. Eq. (2.8) show the equation to calculate the efficiency of a PV module [14].

$$\eta = \frac{P_{MPP}}{E \cdot A} = \frac{FF \cdot V_{OC} \cdot I_{SC}}{E \cdot A} \tag{2.8}$$

where E is the irradiance in the plane of the solar module [W/m^2], and A is the area of the solar module [m^2].

2.6 Yield

Yield is one of the most important calculations for energy production in PV systems [30], with a unit [Wh]. Yield primarily depends on the radiation energy, H_G, arriving at the generator but also on the system component efficiencies and losses. Yield can be calculated for different parts of the energy production (see figure 2.10) [14]. The reference yield, generator yield, and final yield are defined below. Specific yield is a measure of the total energy generated per Wp, with the unit [Wh/Wp].

2.6.1 Reference Yield

Reference yield, Y_R, is calculated by dividing the total in-plane irradiation by the full irradiance of the Sun under standard test conditions. If the radiation energy, H_G is divided by 1000 W/m^2, the result will be in the unit hours [h]. The unit hours refer to the number of hours during which the Sun would have to shine with full force on the generator to generate the solar energy H_G. The reference yield does not take losses into account [14, p 278] and can be defined as:

$$Y_R = \frac{H_G}{E_{STC}} \tag{2.9}$$

where H_G is the radiation energy [Wh/m^2] and E_{STC} is the irradiance at STC [1000 W/m^2].

2.6.2 Generator Yield

Generator yield, Y_A, is the yield calculated at the PV side of the generation, taking generator losses, L_C (array capture losses) into account. The unit of Y_A is [Wh/Wp] [14, p 278], and it is defined as follows:

$$Y_A = \frac{E_A}{P_{STC}} \tag{2.10}$$

where E_A is the energy output DC [Wh] and P_{STC} is the nominal power of the PV system.

The generator losses represent the losses due to array operations. They include losses for temperature higher than T_{STC}, if the modules are partly shaded, if the modules are not operating in MPP, ohmic losses in DC lines, etc. [31] [14, p. 279]

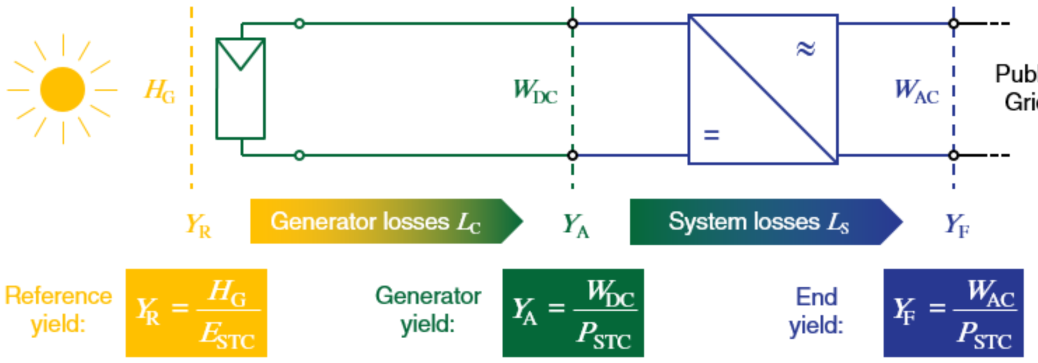
2.6.3 Final Yield

The final yield is calculated at the inverter AC side of the generation, taking system losses, L_S into account. The equation for final yield is given in eq. (2.11) [14, p 278], and the unit is Wh/Wp. The final yield is the entire PV system's net energy output divided by the rated kW (DC) installed. [31, p 36]

$$Y_F = \frac{E_F}{P_{STC}} \tag{2.11}$$

where E_F is net energy output AC [Wh] and P_{STC} is the nominal power of the PV system [Wp].

The system losses are losses caused by the inverter and assoiated cabling. This includes losses due to inverter efficiency less than 100 %, the inverter is under dimensioned, ohmic losses in the AC lines, or cable losses. [14, p 279]



Source: K. Mertens: textbook-pv.org

Figure 2.10: Yields and losses for a PV system [14].

2.6.4 Performance Ratio (*PR*)

The performance ratio is defined in IEC 61742 [31] as a measurement of the PV system's performance and how efficiently the PV system converts sunlight into power. The PR compares the final yield to the reference yield, or the ratio between produced output power to theoretically available energy output under STC conditions This gives the PR based onAC values (PR_{AC}). A typical value for PR is between 75 % to 85 % [14, p 279].

$$PR_{AC} = \frac{Y_F}{Y_R} \tag{2.12}$$

The PR may also be calculated using DC values. This is given by eq. (2.13). This result in a higher PR value, due to the avoidance of losses from DC to AC conversion.

$$PR_{DC} = \frac{Y_A}{Y_R} \tag{2.13}$$

Several factors may affect the value of the performance ratio. These are environmental factor and some other types of factors [32]. The environmental factors are temperature, solar irradiation, PV module shading or soiling. Other factors that may influence the PR are measurement period if the period is too short, conduction losses, the efficiency factor of the PV module, the efficiency factor of the inverter, the difference in PV technology and measurement gage, degradation of the PV module, and orientation of the measurement gage [32]. Unlike the specific energy production [kWh/kWp/year], the PR is not dependent on the meteo input or plane orientation, this allows for comparison of performance of systems at different locations and orientations [33].

2.6.5 Temperature Corrected Performance Ratio

The performance ratio is directly affected by the temperature of the PV module. Depending on location globally, the system may experience seasonal variations due to climate. To compare PR of different systems, it is necessary to calculate a temperature corrected PR (PR_{corr}) [34]. A common method is to correct the performance ratio to the temperature at STC values ($25°C$). This solves the problem of seasonal variations. However, it often results in a higher PR. The equation for a STC corrected PR is given by first temperature correcting the output power (eq. (2.14)) and then using this P_{Tcorr} to correct the performance ratio (eq. (2.15)).

$$P_{Tcorr} = \frac{P}{1 + TC(T_m - 25°C)} \tag{2.14}$$

where P is the output DC power [W], TC is the temperature coefficient for power provided by the manufacturer [$\%/°C$], and T_m is the PV module temperature [$°C$].

$$PR_{corr} = \frac{P_{Tcorr}/P_{STC}}{E/E_{STC}} \tag{2.15}$$

where P_{Tcorr} is the temperature corrected output power [W], E is the global irradiance [W/m^2], and E_{STC} is the irradiance at STC [$1000W/m^2$].

2.7 Module Temperature

Solar cell performance decreases with an increase in temperature, due to an increase in carrier recombination rate with an increase in temperature. When the temperature of the PV module increases, the current of the module increase slightly, while the voltage decrease substantially. This reaction is visualised in the IV-curve in figure 2.11. Both the electric efficiency and the power output depend linearly on the temperature of the PV module.

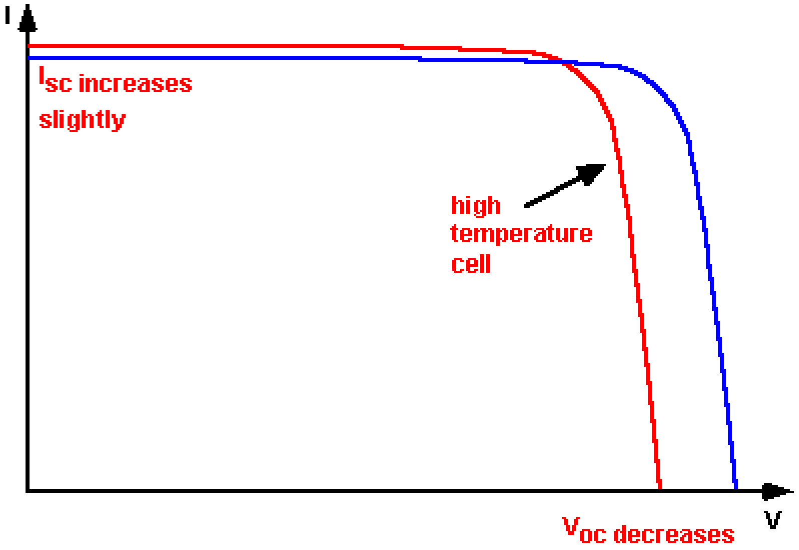


Figure 2.11: IV-curve with the increase in module temperature.

2.7.1 Sandia Temperature Model

The Sandia transfer model is an empirically-based model developed by King and Boyson [35]. This model is divided into two equations, determining the module back surface temperature eq. (2.16), and determining the internal cell operating temperature eq. (2.17). Values recommended for the empirically determined coefficients (a, b) given by King and Boyson [35] are given in table 2.1. If needed, these values may be recomputed for different designs. [36]

Table 2.1: Empirical convective heat transfer coefficients [36].

Module type	**Mount**	**a**	**b**	ΔT_{cnd}
Glass/cell/glass	Open rack	-3.47	-0.0594	3
Glass/cell/glass	Close-roof mount	-2.98	-0.0471	1
Glass/cell/polymer sheet	Open rack	-3.56	-0.0750	3
Glass/cell/polymer sheet	Close-roof mount	-2.81	-0.0455	0
polymer/thin-film/steel	Open rack	-3.58	-0.1130	3

$$T_m = E_{POA} * exp(a + b * WS) + T_a \tag{2.16}$$

where E_{POA} is the global irradiance in the plane-of-array $[W/m^2]$, a is the empirical constant reflecting the increase of module temperature with sunlight, b is the empirical constant reflecting the effect of wind speed on the module temperature [s/m], WS is the wind speed [m/s], and T_a is the ambient temperature $[°C]$.

Once the module back temperature is determined, the cell temperature can be calculated using eq. (2.17).

$$T_C = T_m + \frac{E_{POA}}{E_{STC}} \cdot \Delta T_{cnd} \tag{2.17}$$

where T_m is the module back surface temperature $[°C]$ and ΔT_{cnd} is the conduction temperature drop $[°C]$.

2.8 Solar Radiation

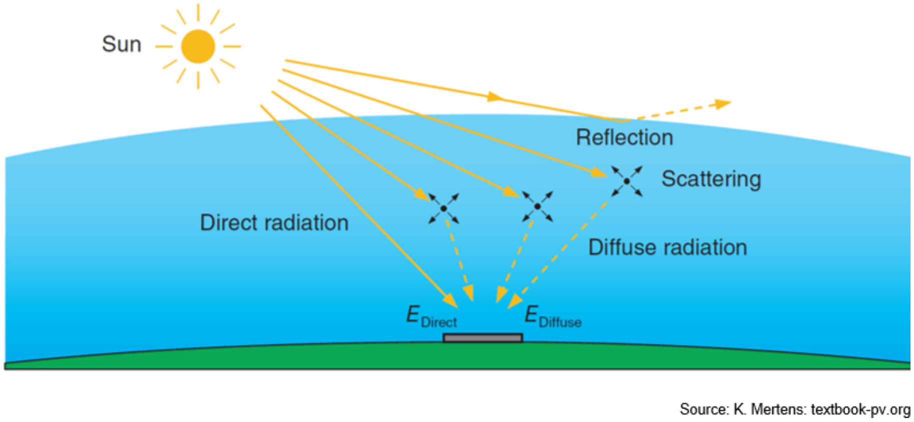
2.8.1 Irradiance (E)

The Sun constantly radiate an amount of $P_{Sun} = 3.845 \cdot 10^{26} W$ in all directions [14]. When referring to irradiance, E, or power density, this is the amount of the Sun's radiation that reaches a specific point here on Earth. Total radiation is calculated as the sum of direct and diffuse radiations, eq. (2.18), and is referred to as global radiation [14, p 23].

$$E_G = E_{Direct} + E_{Diffuse} \tag{2.18}$$

Direct radiation is the amount of radiation that comes directly from the Sun. Due to reflection, absorption and scattering processes as the sunlight travels through the atmosphere, only about 61 % of the Sun's total radiation is received here on Earth. [14, p 26]

Diffuse radiation is radiation from the Sun that has been scattered in the atmosphere but still reaches the Earth's surface. Even though only 61 % of the Sun's radiation reaches the Earth's surface as direct radiation, the final radiation is higher due to diffuse radiation. In some places on Earth, diffuse radiation may contribute as much as 60 % of the annual global radiation [14].



Source: K. Mertens: textbook-pv.org

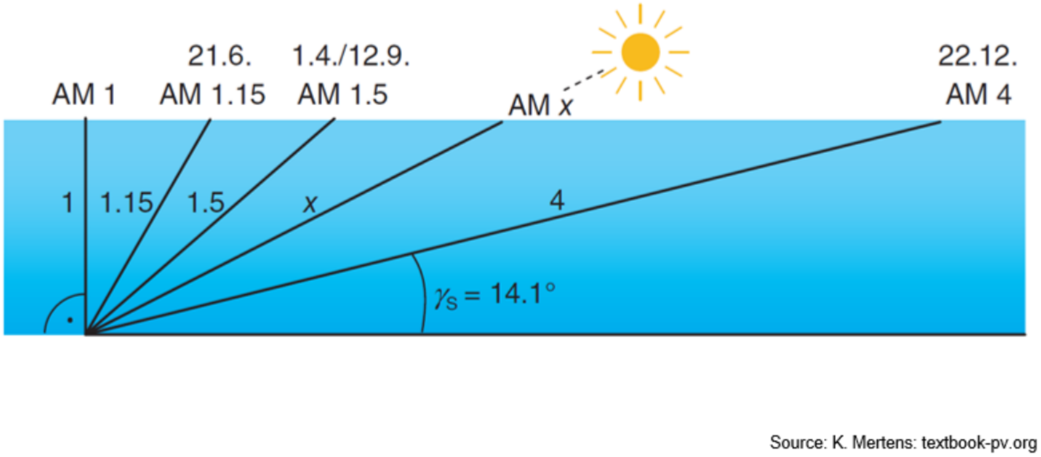
Figure 2.12: Global irradiation, sum of direct and diffuse radiation [14].

2.8.2 Air Mass

As explained in 2.8.3 the solar spectrum changes when the light passes through the atmosphere. To describe this, the term air mass AM is used. AM refers to how far the light has traveled compared to the distance to the vertical path through the atmosphere. It is a measure of how absorption, reflection and scattering processes in the atmosphere affects the spectrum and irradiance of the solar radiation. This is illustrated in figure 2.13. The standard spectrum for measuring solar modules is at $AM = 1.5$, abbreviated to $AM1.5$. This is viewed as the average year's spectrum. However, the air mass changes throughout the day, and it can be calculated using eq. (2.19). [14]

$$AM = \frac{1}{sin\gamma_S} \tag{2.19}$$

where AM is the air mass, and γ_S is the sun height angle. Just outside the earths atmosphere the air mass is zero.



Source: K. Mertens: textbook-pv.org

Figure 2.13: Explanation of air mass [14, p 25].

2.8.3 Solar Spectrum

The energy in solar irradiation comes in different wavelengths, and the wavelength distribution changes with the atmosphere of the Earth. The solar spectrum can be visualized in a graph showing irradiation in $\mathrm{W/m^2} \cdot nm$ by wavelength [nm] (see figure 2.14). Visible light for humans lies between 400 nm to 800 nm. The UV (ultra violet) light is the light with lower wavelengts (0-400 nm). This is the light with the highest energy. The IR (infared) light is the light with wavelengths from 800 nm and higher. The solar spectrum changes with location and time of day [37] [14, p 24]. The dashed line show the black body spectrum. This is the idelized curve of the radiation from the Sun. According to Plank's law every hot body gives off radiation to the surroundings. It is the surface temperature that decides determines the spectrum. The surface temperature of the Sun is 5578 K [14].

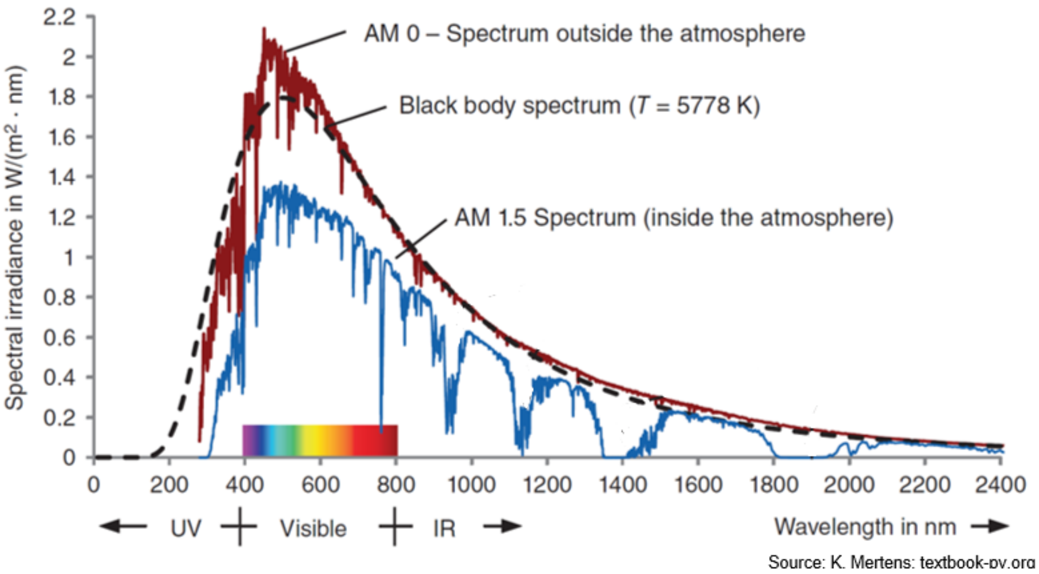


Figure 2.14: Graph of the solar spectrum inside and outside of the Earth's atmosphere [14].

2.9 Spectral Effects on PV Performance

Solar cells are spectrally sensitive, and the spectral irradiance affects the performance of the cell. The spectral effect is the difference between the solar spectrum under which the solar cell operates and the standard solar spectrum. [38]

2.9.1 Spectral Response (SR)

Spectra response (SR) describe the sensitivity of the solar cell to radiation of different wavelengths, or how well the solar cell respond to radiation at different wavelengths. The spectral response decreases at small wavelengths and is limited at long wavelengths because the energy in the photon becomes too small to overcome bandgap of the semiconductor [39]. Figure 2.15 show the spectral response between 400 and 1100 nm for a standard silicon solar cell. [14]

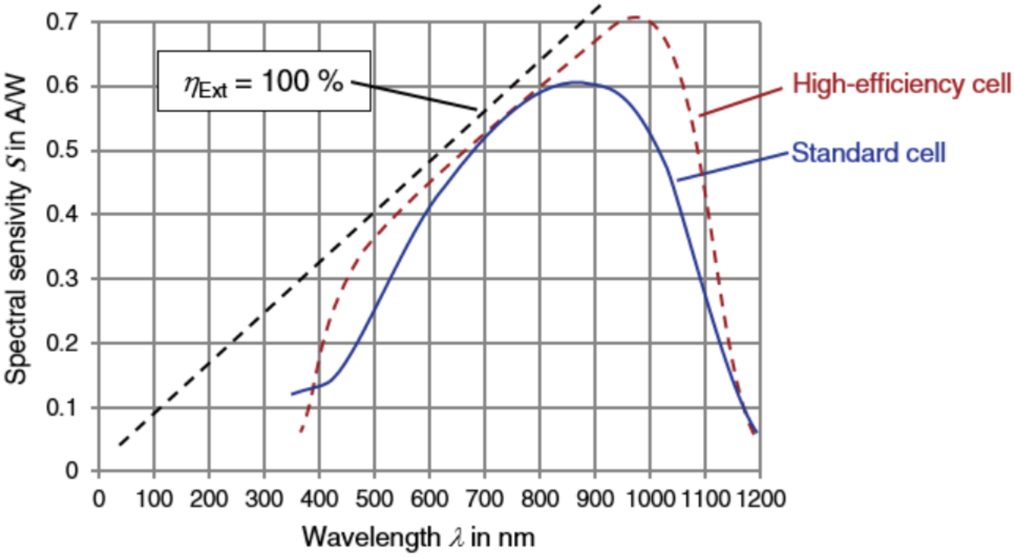


Figure 2.15: Spectral response for a c-Si solar cell (blue), and a high-efficiency cell (red). The $\eta_{Ext} = 100\%$ *is the ideal curve [14].*

2.9.2 Spectral Factor (*SF*)

The actual spectrum of solar radiation under outdoor conditions rarely matches the spectrum under STC. The spectral related mismatches are referred to as spectral mismatch errors. These spectral errors can be analyzed by computing the spectral mismatch factor, or just spectral factor (SF).

The spectral factor described the performance of a PV module operating under STC spectrum with respect to the performance under real incident solar irradiance. It can be used as an estimation of spectral gain or loss due to the deviance from the STC [40]. A SF greater than 1 means spectral gain, while a SF lower than 1 means spectral loss [41]. The SF can be computed using eq. (2.20).

$$SF = \frac{I_{sc} \cdot E_{POA(STC)}}{I_{sc(STC)} \cdot E_{POA}} \tag{2.20}$$

where I_{sc} is the short circuit current at a given time [A], $I_{sc(STC)}$ is the short circuit current under standard test conditions [A], E_{POA} is the irradiance in the plane-of-array at a given time [W/m^2], and $E_{POA(STC)}$ is the irradiance in the plane-of-array under STC [1000 W/m^2].

2.9.3 Solar Incident Angle

The angle of incident (AOI) is the angle between the line that points to the sun and the line normal to the PV module surface, see figure 2.16. The optimum AOI for a PV module is zero, i.e. normally incident light on the plane of array.

The tilt angle (β) is the angle at which the PV module is mounted, relative to the horizontal ground. The tilt angle and orientation may be installed to be either fixed or sun-tracking, i.e., either one- or two-axis tracking. Sun-tracking is classified according to the number of axes used to track the Sun. An advantage of sun-tracking is the possibility to collect more power from the Sun throughout the year. This thesis will focus on fixed-tilt PV. The optimum tilt angle varies throughout the year with the variation in the altitude of the Sun. However, in fixed tilt systems a single tilt angle is chosen as an annual optimum, or based on physical constraints such as roof angle. Some systems allow changing the tilt a few times during the year, to improve annual performance. One method for calculating the optimum tilt angle is to add 15 degrees to the latitude in summer and subtract 15 degrees from the latitude in the winter [42]. However, many PV panels are fixed, and the tilt angle cannot be changed. The orientation of the PV module is often referred to as azimuth angle [14]. It is typically referred to as due South (in the northern hemisphere) or due North (in the southern hemisphere).

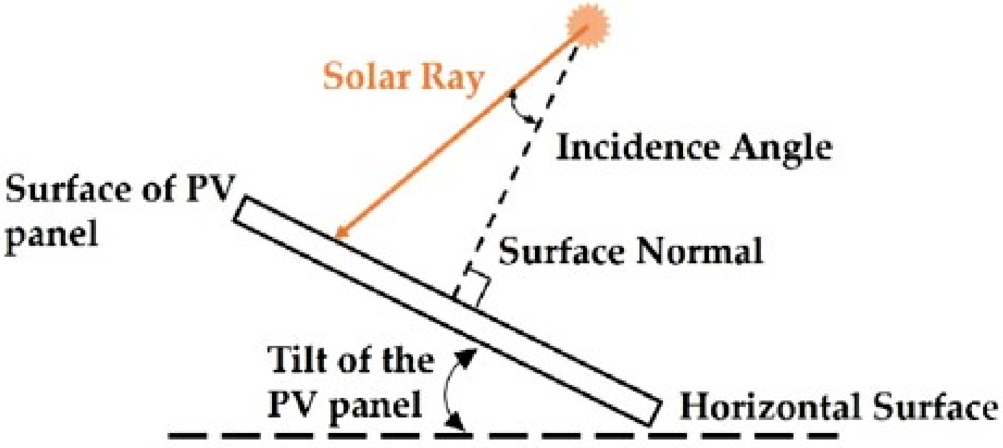


Figure 2.16: Tilt angle and angle of incident for PV module [43].

3 Literature Review

The aim of this report is to investigate PV module performance which, for a given PV technology, is mainly dependent on the local irradiance and temperature conditions. These conditions will again depend on the type of installation, i.e. the orientation and tilt angle of the PV module, and the air-flow around the module. Accurate and consistent evaluations of performance parameters for PV systems are important to evaluate the quality of existing systems and to identify future needs to continue development. Three important performance parameters described in the IEC standard 61724 [44] are generator yield, reference yield, and performance ratio. They define the overall system performance based on solar radiation, energy production, and overall effect on system losses [45] [44].

One of the most important variables for analyzing PV performance is the performance ratio, PR. The PR is a measurement of the performance of the entire PV system. It is the ratio of utilization between output power and power at STC. PR can be calculated using both AC and DC output values for the yield calculations. The difference makes up the conventional losses between the DC output from the PV modules and the AC output from the inverters to the grid. This loss typically has a value from 3 % - 9 % [46]. The PR-values of PV systems in general has grown substantially during the last 40 years [47]. Typical values of AC performance ratio (PR_{AC}) has increased from 50 % - 75 % in the 1980s, to over 80 % today [47]. In [47] Van Sark et al. (2012) asks the question of how high the PR of an entire PV system may actually be. By evaluating losses, degradation, and spectral response, they conclude with a possible PR of over 90 % in today's PV systems. Dhimish (2020) [48] has investigated the DC performance ratio (PR_{DC}) for seven c-Si PV systems scattered over the United Kingdom and Ireland. With measurements over 10 years, they have calculated the monthly PR_{DC}. In conclusion, all PV systems had a mean annual PR_{DC} value above 85 %. The best value was found in Huddersfield at a value of 88.91 %.

Photovoltaic systems have experienced a large increase in Norway over the last few years [3]. In Norway, 22 % of the final energy consumption is used in households, and 37 % in the manufacturing sector [49], thus, encouraging building owners to produce energy in close proximity to the place it will be utilized. The climate in Norway indicates good conditions for solar utilization, with high irradiance and lower temperatures [50, 51]. Some earlier studies have investigated the performance assessment of PV utilization in Norway. Adaramola et al. (2015) [52] investigated a PV system located in Ås municipality in the South-East of Norway. The system consisted of nine poly c-Si modules, with measurements from March 2013 to February 2014. The modules were tilted and fixed at a 37° angle and oriented due South. Adaramola et al. conclude in an annual specific yield of 932 kWh/kWp and a PR_{AC} of 0.83. However, this study does not take temperature correction of PR into account. Pettersen (2019) [53] has assessed the performance of 13 rooftop PV systems in South-East Norway. All systems consist of IBS PolySol c-Si modules and are mounted in an orientation ranging from 130° to 250° (where 0° is defined as North, 90° is East,°180° is South, and 270° is West) with a tilt from 25° to 50 °. By correcting the performance ratio for temperature, he obtained a PR_{corr} between 0.7 and 0.9 for all systems between April to October. Due to lower production caused by factors such as reflection, shadow, and snow covering the modules, the systems experience a lower PR_{corr} in the winter months. During the summer months the East-facing system performed better than the systems oriented West. However, due to the low productions effect on the performance in the winter, the total PR_{corr} did not show any evidence of a better performance for the East-facing system compared to the West-facing. Another study with poly c-Si PV modules is conducted by Imenes (2016) [54] on a flat roof in Vestby, Eastern Norway. The system is mounted at a 10° tilt and placed in alternating rows facing East and West. The reported specific yield was 810 kWh/kWp and the annual system PR_{AC} 0.86. Imenes et al. (2015) [55] present performance results from a PV system mounted on the roof of Agder Energi, in Kristiansand, Norway. This was one of the first grid connected PV systems in Norway. The system consists of c-Si modules, installed at a 20° tilt angle, and an orientation almost due South. The results are based on 2-3 years of data and show a specific yield of 950 kWh/kWp and PR_{AC} of 0.8.

The performance ratio is highly dependent climate conditions at the locations [56, 57]. It has a strong temperature dependency, resulting in large seasonal variations up to around 10 % [36]. To account for the temperature dependency, PR is often corrected to a temperature of $25°C$ (standard test conditions) [36, 58]. This usually results in a higher PR because PV systems typically operate at a higher module temperature than STC. Thus, by correcting the PR to a value of $25°C$, the seasonal temperature variations are eliminated. However, it may overstate the actual expected performance of the system [36]. In [36], National Renewable Energy Laboratory (NREL) studies the possibility to define a site-dependent average cell temperature to which the PR can be corrected. The "weather-corrected" PR considers meteorological aspects that affect the module temperature, such as wind velocity, ambient temperature, and irradiance. This will be a more representative

PR for the given location, which can be used to compare with actual measured PR values but is based on simulation of PV module temperatures from local weather data rather than using actual measured PV module temperatures.

It is well known from earlier studies that an increase in module temperature will have a negative effect on PV module performance [59–62]. PV modules are characterized under STC conditions, however, the real operating conditions rarely correspond to STC. Thus, the efficiency of PV under real conditions is typically lower than the efficiency under STC. The temperature coefficient of a PV module describes how the cell and module output power vary with temperature. The performance of solar cells is largely determined by the open-circuit voltage (V_{OC}), short circuit current (I_{SC}), and fill factor (FF) that determine the efficiency (η). Berthod et al. (2019) [63] establishes the temperature coefficient of the efficiency to be expressed as the sum of the temperature coefficients of open-circuit voltage (β_{Voc}), short-circuit current (β_{Isc}) and fill factor (β_{FF}), as follows.

$$\beta_\eta = \beta_{Voc} + \beta_{Isc} + \beta_{FF} \tag{3.1}$$

In their study, Berthod et al. establish a theoretical model that describes the variation of the temperature coefficients as a function of irradiance. They go on to show that the temperature sensitivity of the solar cell increases at low irradiances. This model does not account for the decrease in β_{Isc} at high irradiances. Nevertheless, is is common to assume a constant temperature coefficient as given by the manufacturer for STC.

The temperature coefficient will vary based on the type of PV technology. Gash et al. (2015) [64] conducted a research of how temperature affects different PV technologies in New Delhi, India. The test was conducted under indoor, laboratory conditions. They tested four different technologies: mono c-Si, poly c-Si, amorphous silicon (a-Si), and CdTe (Cadmium telluride). From the results, they concluded with the highest average temperature coefficient of power for mono c-Si at -0.446 $\%/°C$. For poly c-Si, the average temperature coefficient of power was -0.387 $\%/°C$. The lowest value was measured for CdTe at -0.172 $\%/°C$, making this the best technology with the lowest temperature loss. The mounting configuration of the PV module also plays a role in the effect of temperature on the output performance. The Sandia transfer model is an empirically-based model developed by King and Boyson [35] that attempts to determine the module back surface temperature and the internal cell operating temperature based on empirically determined heat transfer coefficients (see table 2.1). These heat transfer coefficients can be recomputed for different designs according to installation conditions [36]. Such temperature models are useful in cases where PV module temperatures are not available, either due to such data not being recorded, or due to problems with the sensors or missing monitoring data.

Several researchers have investigated the effect of tilt angle and orientation on PV performance and yield, for instance [65–67]. Orientation of PV module (azimuth angle) is a measure of the direction of the panel relative to due North or South [66]. The reference direction is usually due North or South giving the maximum power output, in the Southern and Northern hemisphere, respectively [66, 68]. For fixed mounts, the orientation is most commonly due South [66]. It has also been recommended by several authors [65–67] that a PV system should be installed with a tilt angle equal to the latitude of the site. PV systems with higher tilt angles achieve higher yield performance during the winter, while systems with lower tilt angles achieve higher yield performance during the summer [66, 67]. However, the optimum tilt angle is site-dependent, and some locations experience a weather pattern where winter is cloudier than the summer. Abdallah (2020) [69] has used a mathematical model to calculate the optimum tilt angle on a daily, monthly, seasonal, semi-annual, and annual basis. In addition, they used PVGIS and PVWatts to calculate thesame angles, and compare the results to the mathematical model. The systems investigated are located in two cities in Palestine, Jerusalem and Gaza. They both consist of mono c-Si modules. The results show that a monthly adjustment of optimum tilt angle can generate about 17 % more energy, and a semi-annualy adjustment can generate about 15 % more energy. The yearly optimum tilt angle generates about 10 % more energy than installing the panels at a fixed horizontal surface. Mubarak et al. (2019) [70], argue that even though an orientation due South obtain the highest annual energy output, this may no longer be the optimum orientation. By measuring irradiance with silicon sensors over a three-year period in Germany, they argue that an East and West orientation has the highest self-consumption rate due to better matching with the load. They also include an economic analysis of a PV system without feed-in-tariff that concludes that a South-East, South-West, or East-West facing system has a lower electricity cost than a South facing system [70].

Installations of BIPV systems are, as BAPV systems, rapidly increasing. In [54] Imenes (2016) investigates several BIPV systems in different parts of Norway. The systems perform with different results based on location, design, site conditions, and shading challenges. However, the annual specific yield typically varies from 700-900 kWh/kWp. In his master thesis Frivold (2018) [71] compares the performance of seven different PV systems in different parts of Norway. All these systems have mono c-Si modules mounted vertically on a South facade. For some of the systems he compares BIPV modules mounted on the facade to BAPV modules mounted on the roof of the same building. The data used are from May 2017 to April 2018, and the results give an annual specific yield that varies from 500 to 770 kWh/kWp for the BIPV systems mounted on South-facing facades. When comparing the specific yield for the facade mounted system to the roof-mounted system the results show that the facade produced 80 - 98 % of the roof modules. In addition, the facade energy production was more evenly distributed throughout the year than the roof-mounted system.

In conclusion, this literature review shows that there is a large range of performance values reported for PV modules installed in different locations and in different configurations. Hence, further documentation of performance from various installations will be helpful as a reference for system comparisons and technology evaluations.

4 Method

This chapter will first describe the case study of this thesis, then data availability and filtering, and lastly the method used for data analysis.

4.1 Description of the Photovoltaic Systems at UiA Grimstad

The measurements for this study were performed at the University of Agder (UiA), Campus Grimstad. The PV systems are located at the roof of the J5 building, at a latitude of 58.334630 ° N and longitude of 8.575214 ° E.

4.1.1 PV Modules

The PV systems at the J5 building at UiA are divided into two parts. The first part is a grid-connected system with the modules mounted at 10° tilt angle in an East/West configuration. The second part is a research system, monitored with 1-minute I-V sweeps and held at MPP between sweeps, with the modules mounted south-facing at a 45° rack, as well as two BIPV modules mounted at a 90° tilt angle. Orientation is described according to the same convention as used by the software program PVsyst, with 0° due South and -90° due west. All analyses are done using DC values.

The PV modules mounted at a 10° tilt angle are divided into two arrays consisting of IBC mono c-Si and IBC poly c-Si modules. Each array consists of 40 modules. The modules are mounted with an alternating East-West, with an azimuth angle of -96.8° for the East-facing modules and 83.2° for the West-facing modules. The main part of the performance analysis is conducted on four modules from the middle of the PV array. These modules were selected to avoid shadow from nearby objects. Figure 4.1 and 4.2 show a photograph of the modules and a schematic overview of the chosen modules. The thick line surrounding the PV modules is the fence. In addition, the three modules circled with a stippled line are used when investigating performance on modules exposed to shadow from the fence, see chapter 4.5.

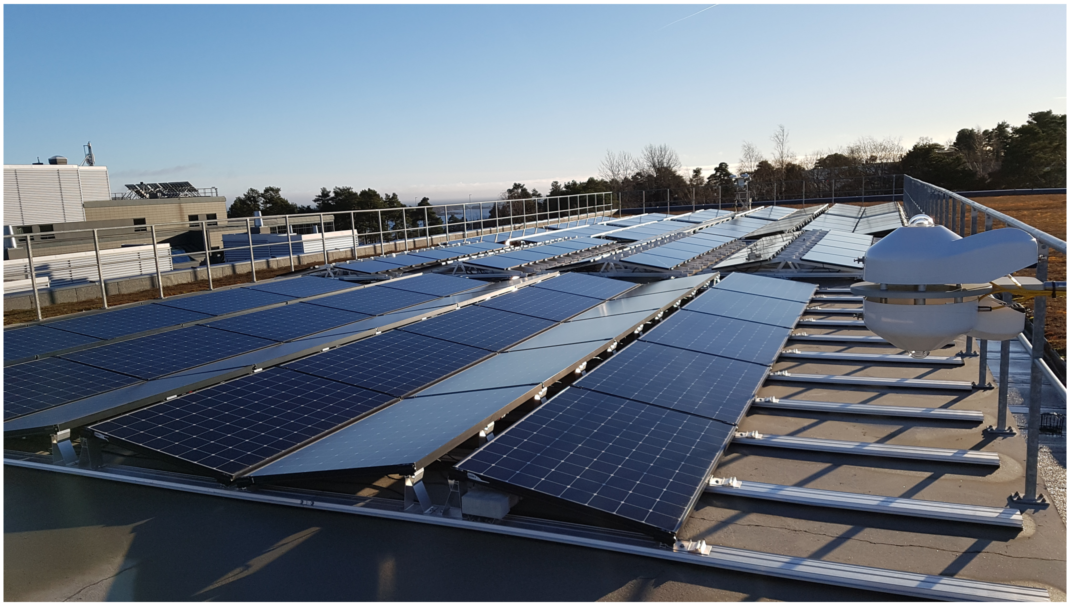


Figure 4.1: Picture of the PV modules mounted East/West with a 10° angle at the roof of the University of Agder. Picture taken from North looking South.

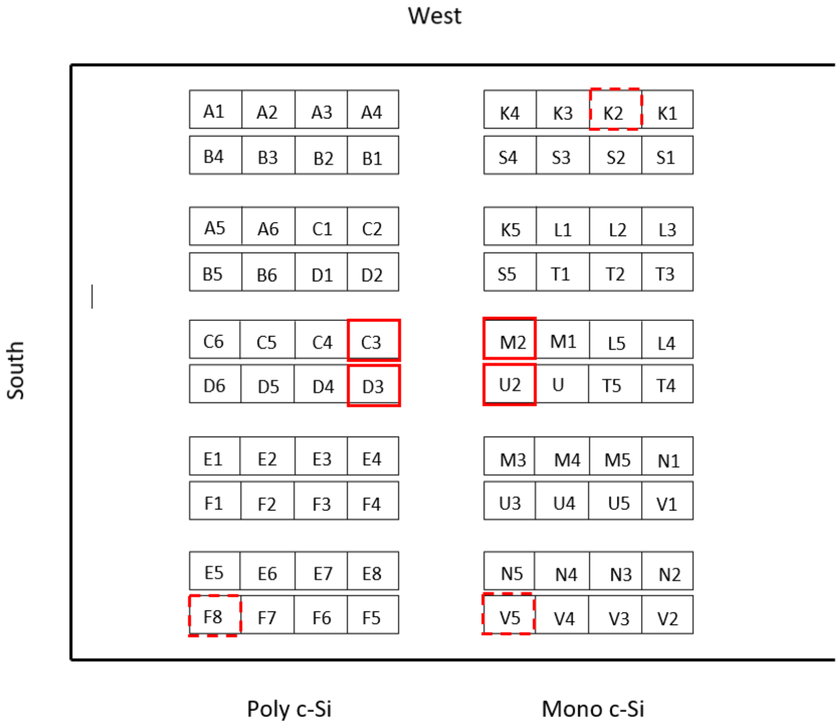


Figure 4.2: The four modules used for analysis, representing the 10° East and West configurations without shading from the surrounding fence, indicated by the red solid boxes. In addition, three modules influenced by shading from the surrounding fence are indicated by the stippled boxes.

At the 45° rack, there are several different PV modules mounted. However, this thesis will investigate the IBS Solar mono c-Si and the IBC Solar poly c-Si modules as shown in figure 4.4. These modules are identical to the analysed modules in the 10° tilt grid-connected system. The two modules mounted at 90° are building-integrated PV modules (BIPV). Both BIPV modules are poly c-Si, however, the module shown in the left in fig 4.5 is tinted grey to simulate the color of the facade cladding used at the University of Agder in Grimstad. All modules on the rack and the BIPV modules are mounted at an orientation of -6.8°. The BIPV poly-Si modules are produced by Ertex and have different nominal power compared to the IBC poly-Si modules. These modules were installed as part of an international research collaboration in the IEA PVPS Task 15 program [72].

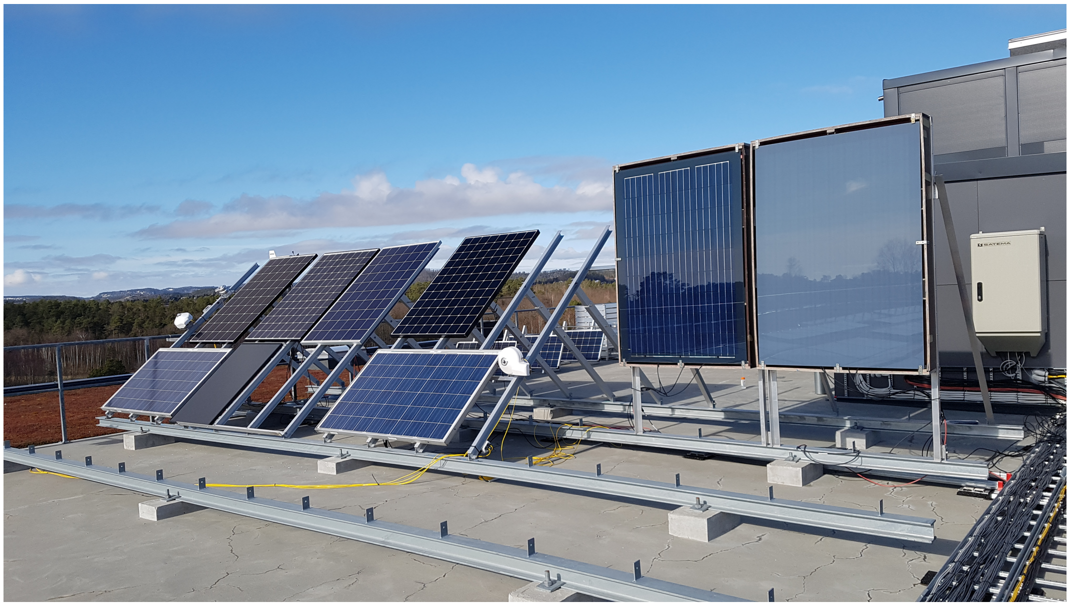


Figure 4.3: Picture of the PV modules mounted on the 45° tilt rack and the vertical BIPV modules at the University of Agder.

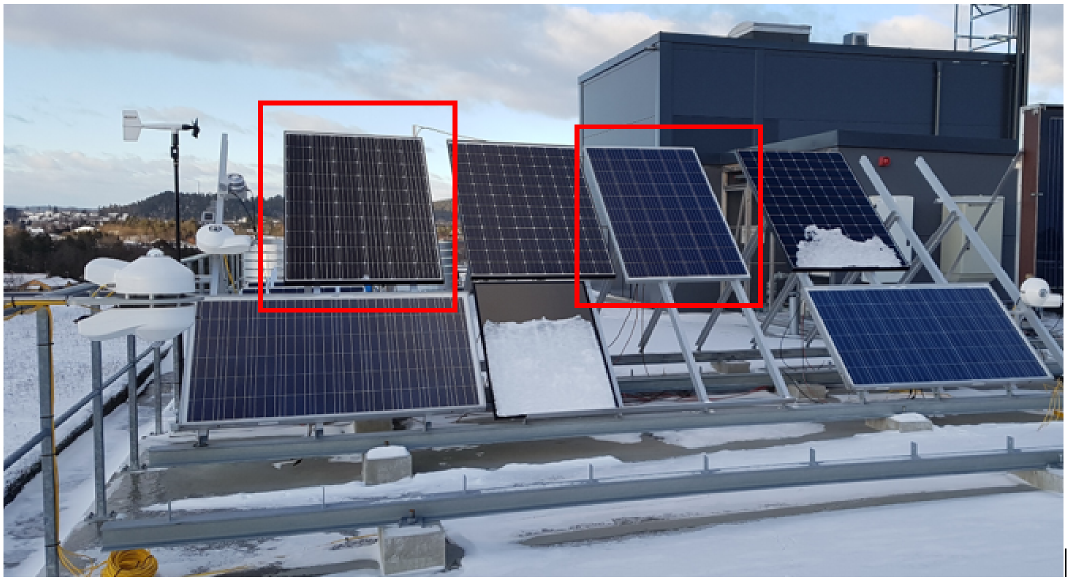


Figure 4.4: The two modules used from the 45° tilt rack: mono c-Si (left), poly c-Si (right).

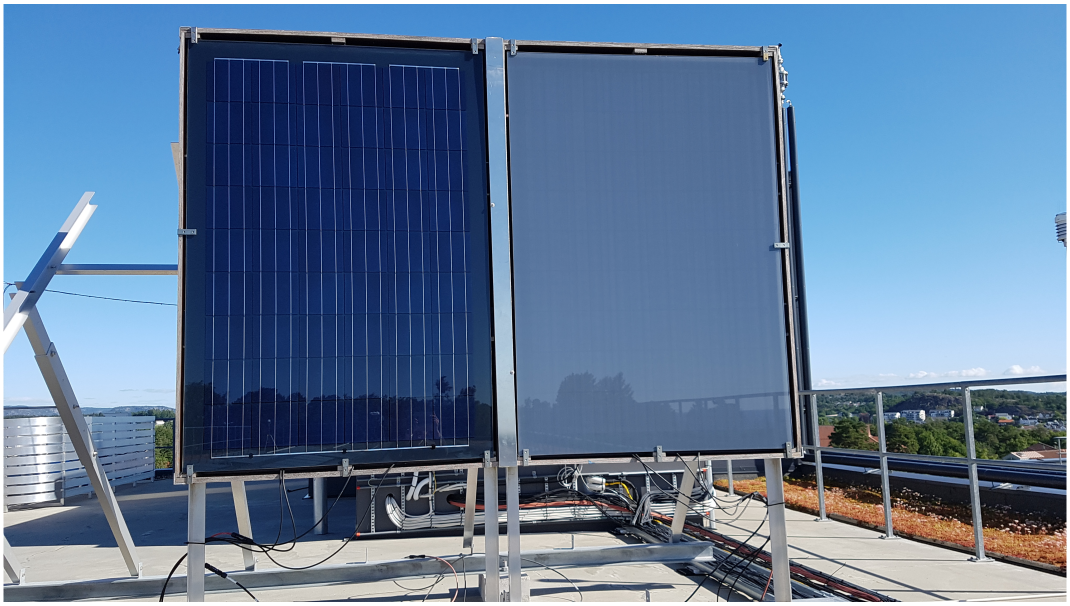


Figure 4.5: Picture of the normal BIPV (left) and the grey BIPV (right) modules mounted at a 90° tilt.

Table 4.1 shows an overview of installation details for the 8 modules used in the analysis. Table 4.2 shows an overview of some technical details extracted from the datasheet of the PV technologies. The IBC mono- and poly-Si modules have a traditional polymer backsheet, whereas the BIPV modules are of a glass-glass configuration. The BIPV normal data is based on initial indoor laboratory characterization, given in Appendix C, by the Austrian Institute of Technology (AIT) as part of the IEA PVPS task 15 Round Robin. The BIPV grey module is identical to the BIPV normal module, except for the added color that has been printed on the backside of the front glass. This module was custom-made for the University of Agder and its initial performance data was not provided. Hence, this data has been estimated from the BIPV normal module data in combination with outdoor measurements of both BIPV modules. It has been assumed that the main influence of the color-coating is to reduce the photo-generated current of the module due to transmission loss.

Table 4.1: Overview of the PV modules at different orientations used in this report.

Tilt angle	**Technology**	**Module type**	**Module area [m^2]**	**Orientation**
10°	Mono c-Si	IBC MonoSol 315 VL5	1.627	-96.8°
10°	Poly c-Si	IBC PolySol 270 CS4	1.637	83.2°
45°	Mono c-Si	MonoSol 315 VL5	1.627	-6.8°
45°	Poly c-Si	PolySol 270 CS4	1.637	-6.8°
90°	Poly S-ci	ERTEX VSG-L	1.760	-6.8°
90° Grey	Poly S-ci	ERTEX VSG-L	1.760	-6.8°

Table 4.2: Data from the datasheets and the initial characterization of each PV technology.

Parameter	**IBC Mono**	**IBC Poly**	**Ertex BIPV normal**	**Ertex BIPV Grey**
P$_{mpp}$ [W]	315	270	218.1	193
$I_{sc(STC)}$ [A]	10.02	6.48	8.26	7.31
η [%]	19.4	16.5	12.4	11
Tcoeff$_{Pmpp}$ [%/°C]	-0.38	-0.411	-0.405	-0.405
NOCT [°C]	44	44.1	N/A	N/A

4.1.2 Instrumentation

As the PV monitoring system is quite comprehensive, the full monitoring system will not be described here but only the relevant sensors and data sources used in this analysis:

- **Irradiation sensors POA:** 2 x CMP11 pyranometers with CVF4 ventilators (from Kipp and Zonen) for 45° og 90° rack, and 2 x mono-Si reference cells (from Ingenieurbüro Mencke Tegtmeyer) for 10° East and 10° West.

- **Global horizontal irradiance:** 1 x Sollys II Sun tracker measures the global horizontal (GHI), with a CMP11 pyranometer with CVF4 ventilator (from Kipp and Zonen), and a similar instrument with shading ball assembly to measure diffuse horizontal irradiance (DHI).

- **PV temperature sensors:** Campbell Scientific 110 PV Thermistor, mounted at the back of the representative modules for the different PV technologies.

- **Ambient temperature:** Vaisala WXT535 Weather Transmitter measures ambient temperature.

- **Wind speed:** 1 x R.M. Young 5103 wind monitor from Campbell Scientific measures wind speed and direction.
- **Optimizers:** The 10° tilt PV systems has TIGO TS4-R optimizers at the back of each module monitoring PV module performance, and logging DC production in 1 min resolution.

4.2 Data Filtering and Availability

Before conducting any analysis, a subject to consider is filtering the data to remove any obvious outliers or false data, such as negative irradiance values produced by pyranometers during nighttime. By this means removing any data before sunrise and after sunset, when the PV modules are not producing any output power. This can be one with several different methods. To show the effect of data filtering, performance calculations were visualized in graphs given in chapter 6.1 *Data Filtering.*

The data used in this thesis is collected in a period of 13 months, from March 2020 to March 2021. Because the systems were installed in early 2019, this was found to be the time period with overlapping data for all PV systems. Table 4.3 shows an overview of data availability for each month in percentage. The availability is calculated only accounting for data between sunrise/sunset.

When calculating data availability all data sets were first filtered to only contain data between sunrise and sunset. This was done using the *suntime* module in python. The script is attached in Appendix D and calculates the sunrise/sunset times for each day based on the latitude and longitude of the location. It then creates a new file containing data points between sunrise/sunset. For data points missing in the original file, it creates empty data points containing "NaN" values. The availability was found by comparing the filtered data to the original data. The data is given in one-minute resolution. To obtain the number of missing data points in each data set, another python script was written, given in Appendix D. Here the total amount of data points for each month was extracted, then the *isna.sum()* function in python was used to count the total number of missing data point. Eq. (4.1) was used to find the percentage missing data for each month.

$$Data\ availability = \frac{n}{N} \cdot 100\% \tag{4.1}$$

where n is the number of missing data points, or rows containing "NaN" values, and N is the total number of data points for each month between sunrise/sunset.

Table 4.3 shows an overview of data availability for all PV modules in percentage. This is important to know when analyzing and comparing calculations. As seen from the table there is a substantial difference in data availability between the 10° and 45°/BIPV systems. This is due to the use of different data collecting systems, which were not affected by the same downtime periods. When calculating the PR for each system the difference in availability is less important, because the PR equation uses the ratio of final yield to reference yield. However, performing specific yield calculations for different PV modules that have dissimilar data availability due to technical issues not related to the module technology itself will not give a fair comparison. In this case, as an attempt to provide a more common basis for comparison of performances, missing data points can be estimated based on regression analysis, as described in section 4.6.

Table 4.3: Data availability for each month in %.

Availability [%]	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**	**Jan 21**	**Feb 21**	**Mar 21**
Mono and poly 45°	79.28	93.37	89.48	83.46	81.13	91.66	56.18	68.00	78.57	93.62	91.63	88.87	75.96
BIPV normal and grey	79.28	93.37	89.48	83.46	81.13	91.66	56.18	68.00	78.57	93.62	91.63	88.87	75.96
Mono (10°) East	96.45	98.00	98.69	96.46	97.38	97.76	96.92	94.53	89.90	63.90	84.16	78.64	96.94
Mono (10°) West	95.87	98.02	98.15	96.28	97.19	97.39	96.58	93.94	90.51	72.72	85.84	76.01	96.51
Poly (10°) East	95.64	98.21	98.65	96.75	97.52	97.86	97.11	94.84	91.19	71.18	85.32	73.85	96.89
Poly (10°) West	95.71	97.20	97.70	95.45	96.41	96.72	96.01	93.59	89.64	70.37	80.93	72.66	95.61

4.3 Yield Calculations

4.3.1 Generator Yield

The generator yield was calculated using eq. (2.11). Here the DC output power (P_{DC}) for each PV module was used. In the data sets containing measurements with one-minute intervals, the output power was multiplied with (1/60) h to get the energy in Wh. The nominal power of each PV module, P_{STC}, is given in table 4.1. The output energy was then summed for each month and divided by the nominal peak power for each PV modules.

$$Y_A = \frac{\sum(P_{DC} \cdot (1/60))}{\sum(P_{STC})} \tag{4.2}$$

4.3.2 Reference Yield

When calculating the reference yield, the plane-of-array irradiance (E_{POA}) was divided by the irradiance at STC (E_{STC}), as expressed in eq. (2.9). The irradiance used was measured in was measured by either pyranometers or silicon reference cells mounted in the same planes as the PV modules evaluated. The irradiance [W/m^2] was measured with one-minute resolution and multiplied by (1/60) h to convert the data to Wh/m^2]. The data was then sum for each month and devided by the STC irradiance (1000 W/m^2), as shown in eq. (4.3).

$$Y_R = \frac{\sum(E_{POA} \cdot (1/60))}{E_{STC}} \tag{4.3}$$

4.4 Performance Ratio

The PR is dependent on meteorological variations and therefore varies with both location and seasons throughout the year. To investigate this variation both a normal PR and a temperature corrected PR was calculated using DC output power (PR_{DC}). The normal PR was calculated using eq. (2.12), by dividing the generator yield by the reference yield for each PV module.

The temperature corrected performance ratio (PR_{corr}) was calculated using eq. (2.14) and (2.15). For the 45° mono c-Si, 45° poly c-Si, and both BIPV systems there are no temperature measurements available at the start of the period, i.e. March and the first half of April 2020, therefore these calculations span from April 15 2020 to March 30 2021. Firstly the temperature corrected output power was calculated (P_{Tcorr}), using eq. (2.14). The temperature used was measured at the back of each PV module. For the BIPV systems, the temperature is measured at the top and the bottom of the modules, the temperature used to correct the PR was a mean value between these measurements. The temperature coefficient for each system is given in table 4.2. The corrected power was then used in eq. (2.15) to calculate the PR_{corr} for each month.

4.5 Investigation of Power Correction

To further investigate the effect of tilt angle and orientation on the PV performance, further analysis of power correction was conducted. By correcting the power back to STC conditions for irradiance and temperature, the performance of similar PV modules mounted at different orientations can easily be compared, and a module not experiencing shadow can be compared to a module experiencing shadow. These corrections were conducted for 19 April 2020, which was a clear day with good sun-conditions, for all modules. The python code to calculate the power correction is given in Appendix D.

Firstly, the power output and module temperature were plotted. Then the power was corrected for temperature using eq. (2.14). The temperature corrected power was then corrected for irradiance in the plane-of-array ($P_{corr\,T+G}$). This was done by eq. (4.4), as shown by eq. (4.4). Results are given and discussed in chapter 6.4.

$$P_{corr\,T+G} = P_{Tcorr} \cdot \left(\frac{E_{STC}}{E_{POA}} \right) \tag{4.4}$$

where P_{Tcorr} is the temperature corrected power [W], E_{STC} is the irradiance at STC [1000 W/m^2], and E_{POA} is the irradiance in the plane-of-array [W/m^2].

Lastly the $P_{corr\,T+G}$ was corrected for spectral factor, resulting in $P_{corr\,T+G+SF}$. The spectral factor was calculated using eq. (2.20) and resulting in (4.5).

$$P_{corr\,T+G+SF} = \frac{P_{corr\,T+G}}{SF} \tag{4.5}$$

4.6 Estimation of Missing Data by Linear Regression

Due to the large deviation in data availability between the different PV configurations the specific yield cannot be compared directly. To further investigate the performance of the modules at 45° rack and BIPV modules, it is desirable to predict output power for the missing data points, based on the irradiance and measured output power. The output power estimation is done using linear regression on a monthly basis. Firstly, the temperature corrected power (P_{Tcorr}) is calculated for each month using eq. (2.14). Figure 4.6 show the irradiance and temperature corrected power plotted in a scatter plot for the month of May 2020. The *sklearn.linear_ model.LinearRegression* function in python was used to compute the linear regression (python code may be found in Appendix D). Based on this a function for the slope was found. There is no missing data in the irradiance data, thus the irradiance together with the slope of the linear regression is used to estimate the missing output power data. Figure 4.7 show the temperature corrected power for the month of May 2020. The red line is the measured output power, while the black line is where the estimated values are filling in the missing data points.

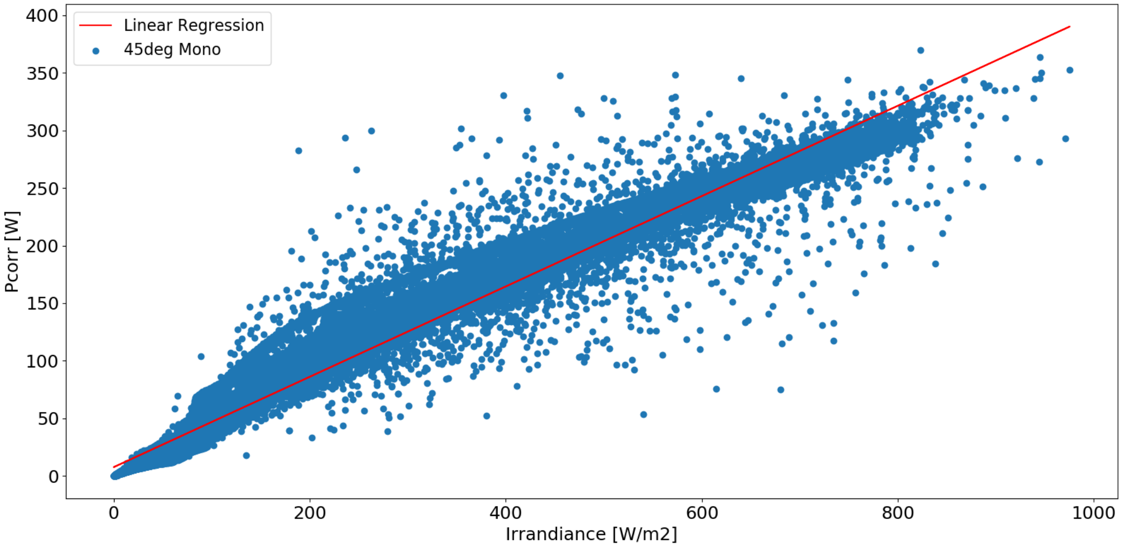


Figure 4.6: The linear regression for May 2020. Scatter plot of the irradiance [W/m^2] on the x-axis and temperature corrected power [W] on the y-axis.

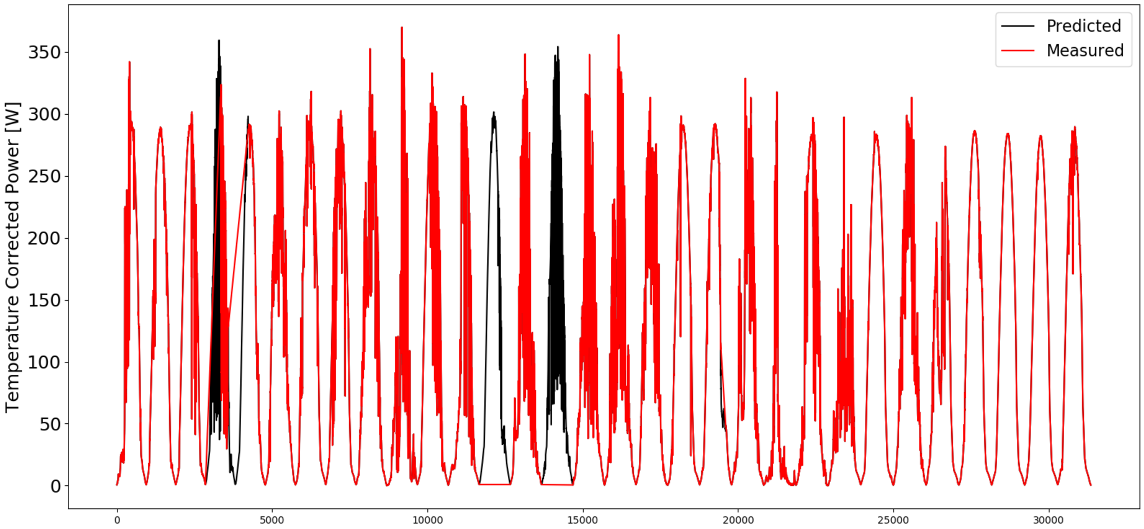


Figure 4.7: Graph showing how the predicted values for output power fill in the gap in the measured output power values, May 2020.

After the linear regression, the P_{Tcorr} had to be corrected back to a non-temperature corrected power. This was done by modifying eq. (2.14) to eq. (4.6), as seen below.

$$P = P_{Tcorr} \cdot (1 + TC(T_m - 25°C)) \tag{4.6}$$

where P_{Tcorr} is the temperature corrected power [W], TC is the temperature coefficient for the PV module [$\%/°C$], and T_m is the module temperature [$°C$].

For the entire month of March 2020 and from the 1st - 5th of April 2020, there is no measured module temperature for any of the PV modules. These values therefore had to be estimated. The estimation is done using the Sandia temperature model, by eq. (2.16). This model is chosen based on the results found in the author's previous Research project [8], conducted in the fall of 2020 at UiA. Here several different temperature models were tested and compared, and it was concluded that the Sandia model resulted in the best module temperature estimation. The values for the convective heat transfer coefficients are selected from table 2.1. The mono and poly c-Si modules are glass/cell/polymer sheet modules and have an open rack configuration. The BIPV normal and grey modules are glass/cell/glass configurations with a closed mounting, however, it has an open top and bottom to allow airflow in the back. With trial and error, the heat transfer coefficients have been adjusted for the best fit. The coefficients used are given for the two 45° tilt and the two BIPV modules in table 4.4.

Table 4.4: Empirical convective heat transfer coefficients for the Sandia temperature model.

PV module	a	b	ΔT_{cnd}
45° mono c-Si	-3.60	-0.0592	3
45° poly c-Si	-3.60	-0.0592	3
BIPV normal	-3.80	0.0471	1
BIPV grey	-3.80	0.0471	1

The final estimated output power is used to calculate PR and final yield for all PV modules, described in chapter 6.2.1 and 6.3.

5 Simulation Software: PVsyst

This chapter contains a description of the simulations carried out for performance predictions for PV modules mounted at the UiA. The measured data and calculated results from the real modules at UiA were compared with the simulated results. The software tool used to do the simulations was PVsyst 7.1. However, there are several other commercial softwares or open-source codes that could have been used to simulate PV module performance, for instance HOMER, PVSol, and PVlib etc. The main resource for this chapter and the different parameters used by the simulation is the PVsyst website [73].

5.1 System Simulated

The simulation in PVsyst was conducted on five mono c-Si PV modules mounted at 10° tilt in the East-facing orientation. The modules are connected in one string to one inverter. These modules are chosen to compare real values from UiA to a simulation. By doing this the performance of the system at UiA may be evaluated, and potential issues in the system may be investigated. It also allows for investigating of degradation of the system, if the simulated values differ considerably form the measured values. Figure 5.1 show the location of the modules on the roof at UiA. A string diagram of the connection is given in chapter 5.4. Performance analysis for the five mono c-Si modules is included in chapter 6.5 together with the simulated results, to compare the performance of the PVsyst simulation to the actual performance of the five mono c-Si modules.

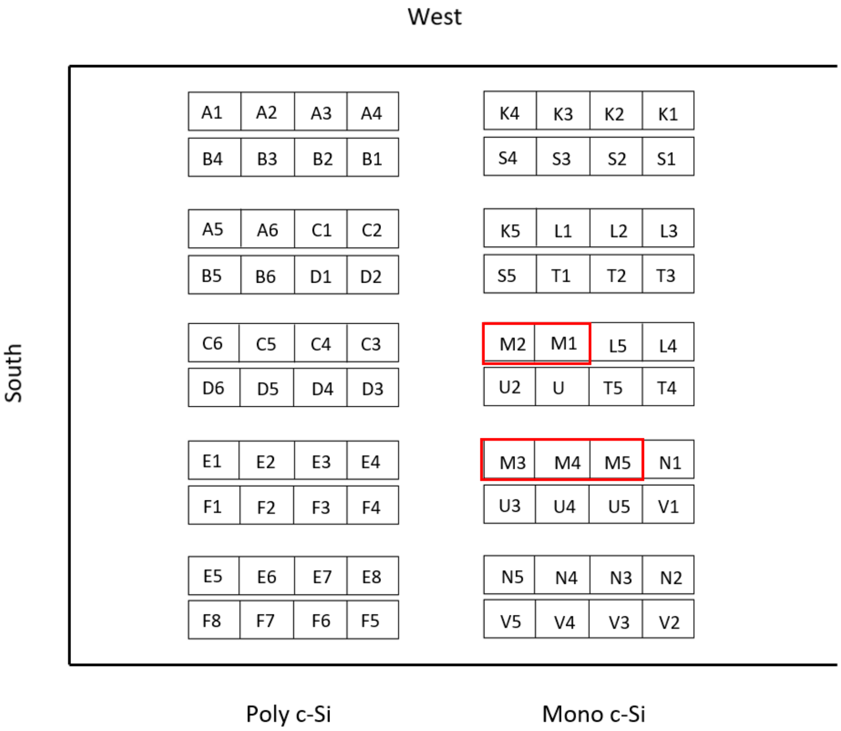
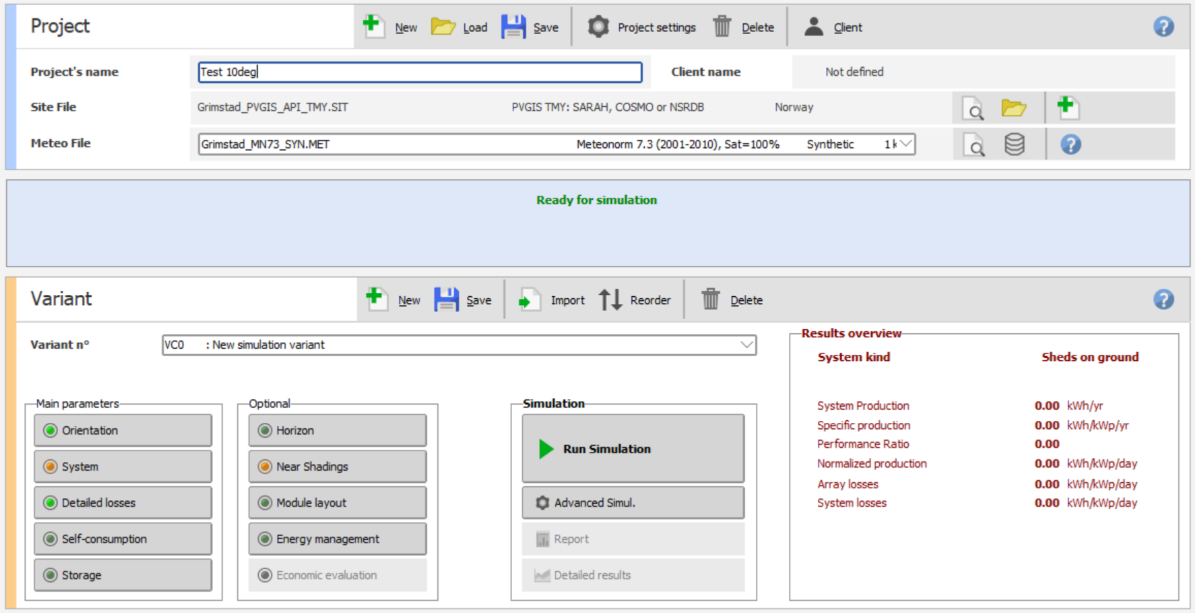


Figure 5.1: Location of the five mono c-Si modules simulated in PVsyst, East-facing, marked in red.

5.2 Project Design

The *Project design* section of PVsyst is the main part of the software. This part aims to perform a thorough design and performance analysis using hourly steps over a full year. Figure 5.2 shows a screen dialog of the project design window. Through different simulation variants, optimization and parameter analysis of the system may be performed. For this thesis, the grid-connected system is selected. The output of the simulations from project design is a thorough performance rapport, given in Appendix B.



Project's name: Test 10deg

Client name: Not defined

Site File: Grimstad_PVGIS_API_TMY.SIT PVGIS TMY: SARAH, COSMO or NSRDB Norway

Meteo File: Grimstad_MN73_SYN.MET Meteonorm 7.3 (2001-2010), Sat=100% Synthetic 1 h

Ready for simulation

Variant n°: VC0 : New simulation variant

Results overview

System kind	Sheds on ground
System Production	0.00 kWh/yr
Specific production	0.00 kWh/kWp/yr
Performance Ratio	0.00
Normalized production	0.00 kWh/kWp/day
Array losses	0.00 kWh/kWp/day
System losses	0.00 kWh/kWp/day

Figure 5.2: Project design overview window in PVsyst.

5.3 Project Specification

The *Project specification* is the definition of site-specific parameters in PVsyst. Here the geographical location, meteorological data and albedo values are specified.

Project site

The project site can either be chosen from a database, or a new site can be created by site coordinates in latitude and longitude. The site coordinates are used to calculate the Sun's position. Coordinates for Grimstad are latitude 58.334630° N and longitude 8.575214° E.

Meteorological data

The meteorological data in PVsyst can either be chosen from a database in PVsyst, which are PVGIS or Meteonorm 7.3, or a custom file based on measured data can be imported. For this project, a custom file using measured values was imported. This makes comparing the simulated results to the calculated analysis more accurate. The required parameters consist of global horizontal irradiance (GHI) in $[W/m^2]$ and ambient temperature in $[°C]$. In addition, diffuse horizontal irradiance (DHI) in $[W/m^2]$ and wind speed (WS) in [m/s] may be included to assure a more accurate result. The conversion in PVsyst requires a *.csv file, and converts it to a PVsyst compatible file. When the data is imported it is important to check data quality to achieve relevant simulations. This can be done PVsyst under *Check data quality*.

The simulation in PVsyst operate at hourly intervals, hence the measure data was imported at 5 min intervals and are converted by the PVsyst conversion tool to hourly values.

Project settings

In the tab *Project settings* some additional parameters related to the project may be defined. This includes albedo, design conditions and other limitations.

The *albedo settings* are defined on a monthly basis. The standard values are set to 0.2 in PVsyst. For this project, the albedo is set at 0.30 to better represent the concrete roof where the PV systems are placed. The value of concrete according to the albedo table in PVsyst is between 0.25-0.35.

In the *design conditions* tab design temperatures use to match the PV array to the inverter are specified. These temperatures are only used for sizing and are not involved in the simulations. These values are set to default values from PVsyst.

Lastly the *other limitations* tab defines the limits with shading representation. These are limits in the 3D constructions and are left to the default values.

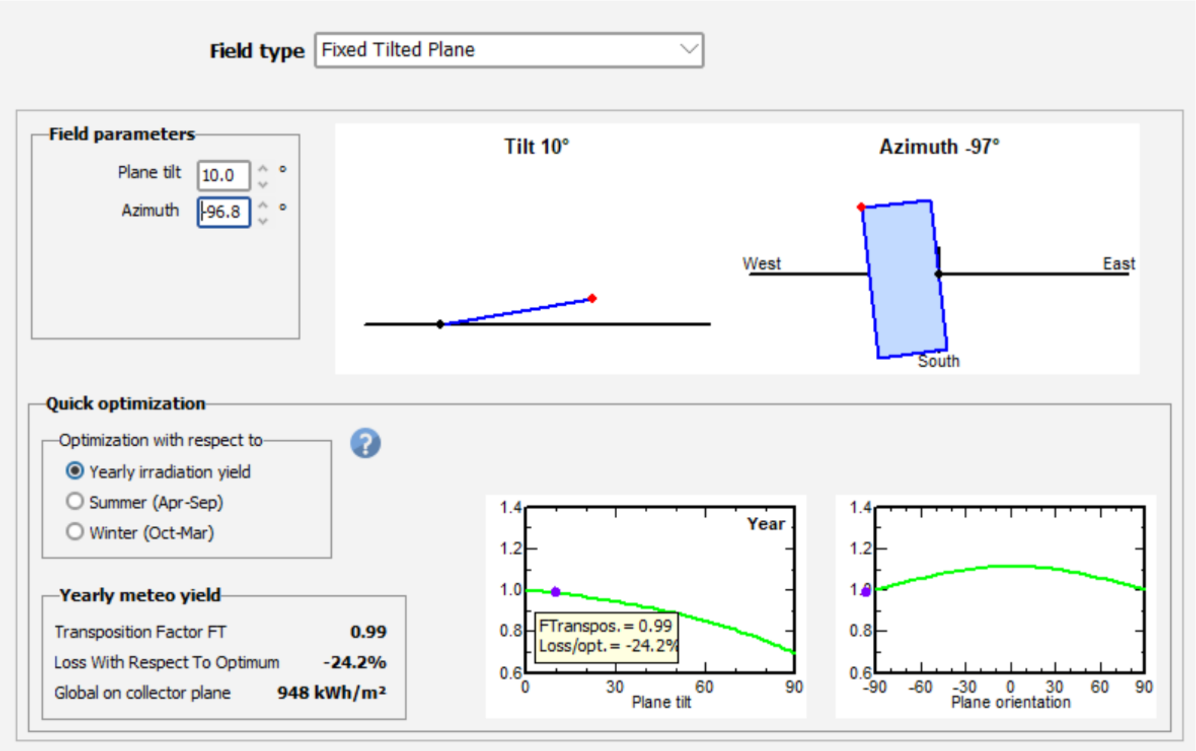
5.4 System Variant Manager

In the *system variant manager* the detailed PV system is defined by a set of input parameters, and the simulation is conducted. The input parameters are divided into mandatory and optional parameters.

Mandatory Parameters

In *orientation*, tilt angle and azimuth angle for the system are defined. Azimuth angle is defined with 0° due South. Positive angles are to the West and negative angles are to the East. For the simulated system the tilt angle is 10° and the azimuth angle is -98.6°. PVsyst also proposes optimized angles based on the geographical site. The optimization may be done using yearly yield, summer, or winter yield.

Figure 5.3 shows an overview of the orientation tab in PVsyst, where the orientation origin is seen as the red dot on the PV module. The quick optimization shows the transposition factor, losses with respect to the optimum orientation, and the available irradiation on this tilt angle. The quick optimization is also given in two graphs. The left graph shows transposition factor and loss due to optimum orientation. The purple dot indicates the tilt chosen for the project. The right graph show irradiation in regards to orientation.



Field type Fixed Tilted Plane

Field parameters

Plane tilt 10.0 °

Azimuth -96.8 °

Quick optimization

Optimization with respect to

● Yearly irradiation yield

○ Summer (Apr-Sep)

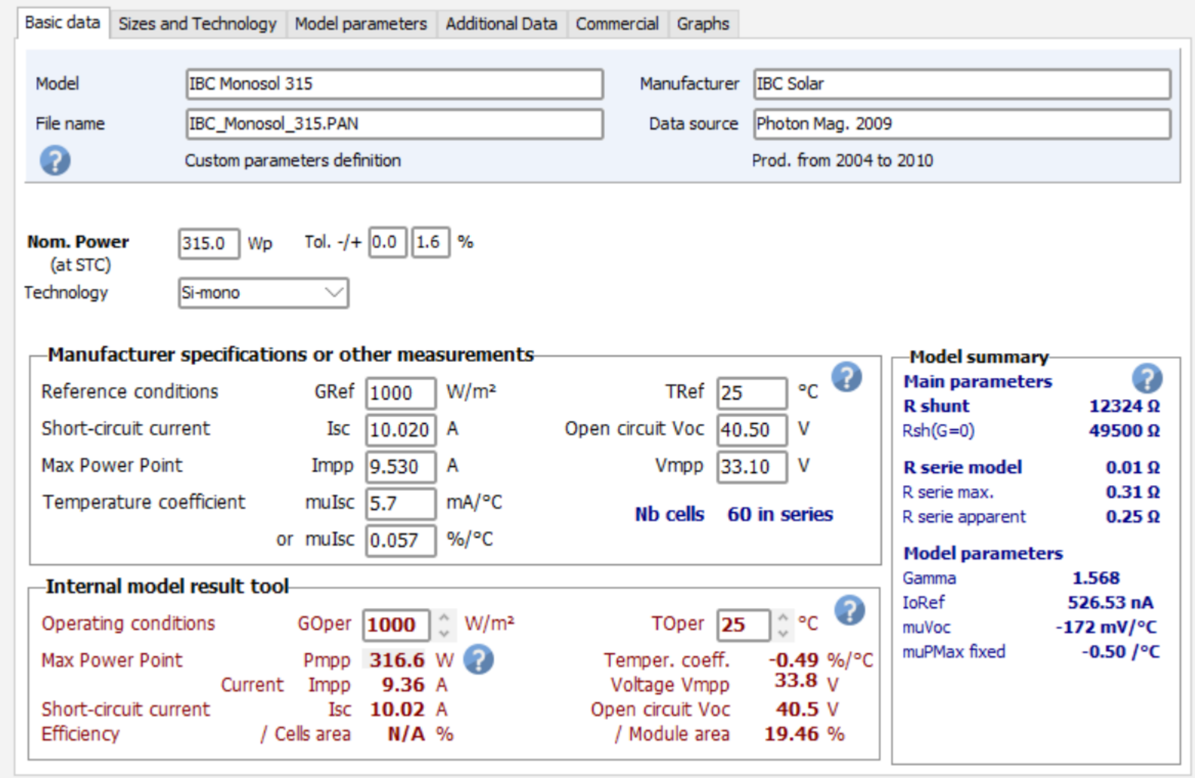
○ Winter (Oct-Mar)

Yearly meteo yield

Transposition Factor FT	0.99
Loss With Respect To Optimum	-24.2%
Global on collector plane	948 kWh/m²

Figure 5.3: Tilt angle and azimuth defined in PVsyst.

In the *System* tab the PV module and inverter type are selected, and the number of sub-systems and strings as well as system design is defined. For this thesis, the module had to be defined manually as the exact module type was not in the library. This was done by changing the values defined in the module selection, as seen in figure 5.4. The module type is IBC Monosol 315. The manufacture specifications are taken from the data sheet for the PV module, given in Appendix C. The inverter type is a Solarxpower 3.0 kW. Figure 5.5 show a screen dialogue of the system window in PVsyst, and the parameters that can be changed to achieve the most accurate simulation. In addition to the inverter, a TIGO TS4-O optimizer is chosen. Number of optimizer in a string is set to five. In the bottom right corner a short summary of the system is given.



Model: IBC Monosol 315
Manufacturer: IBC Solar
File name: IBC_Monosol_315.PAN
Data source: Photon Mag. 2009
Custom parameters definition
Prod. from 2004 to 2010

Nom. Power (at STC): 315.0 Wp Tol. -/+ 0.0 1.6 %
Technology: Si-mono

Manufacturer specifications or other measurements

Reference conditions	GRef	1000 W/m²		TRef	25 °C
Short-circuit current	Isc	10.020 A	Open circuit Voc		40.50 V
Max Power Point	Impp	9.530 A		Vmpp	33.10 V
Temperature coefficient	muIsc	5.7 mA/°C	Nb cells		60 in series
	or muIsc	0.057 %/°C			

Internal model result tool

Operating conditions	GOper	1000 W/m²	TOper	25 °C
Max Power Point	Pmpp	316.6 W	Temper. coeff.	-0.49 %/°C
Current	Impp	9.36 A	Voltage Vmpp	33.8 V
Short-circuit current	Isc	10.02 A	Open circuit Voc	40.5 V
Efficiency	/ Cells area	N/A %	/ Module area	19.46 %

Model summary

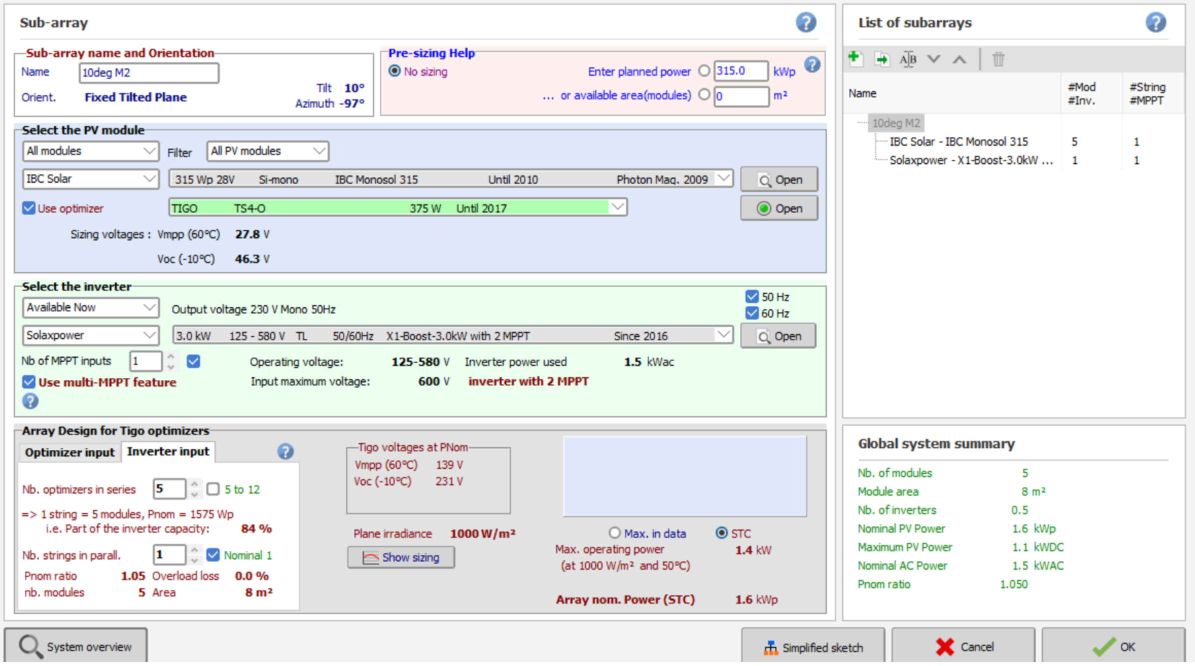
Main parameters

R shunt	12324 Ω
Rsh(G=0)	49500 Ω
R serie model	0.01 Ω
R serie max.	0.31 Ω
R serie apparent	0.25 Ω

Model parameters

Gamma	1.568
IoRef	526.53 nA
muVoc	-172 mV/°C
muPMax fixed	-0.50 /°C

Figure 5.4: Screen dialog for defining the PV module in PVsyst.



Sub-array

Sub-array name and Orientation
Name: 10deg M2
Orient.: Fixed Tilted Plane Tilt 10° Azimuth -97°

Pre-sizing Help
● No sizing
Enter planned power ○ 315.0 kWp
... or available area(modules) ○ 0 m²

Select the PV module
All modules Filter: All PV modules
IBC Solar 315 Wp 28V Si-mono IBC Monosol 315 Until 2010 Photon Mag. 2009
☑ Use optimizer TIGO TS4-O 375 W Until 2017
Sizing voltages : Vmpp (60°C) **27.8** V
Voc (-10°C) **46.3** V

Select the inverter
Available Now Output voltage 230 V Mono 50Hz ☑ 50 Hz ☑ 60 Hz
Solaxpower 3.0 kW 125 - 580 V TL 50/60Hz X1-Boost-3.0kW with 2 MPPT Since 2016
Nb of MPPT inputs: 1 ☑ Operating voltage: **125-580** V Inverter power used **1.5** kWac
☑ **Use multi-MPPT feature** Input maximum voltage: **600** V **inverter with 2 MPPT**

Array Design for Tigo optimizers
Optimizer input | **Inverter input**
Nb. optimizers in series: 5 □ 5 to 12
=> 1 string = 5 modules, Pnom = 1575 Wp
i.e. Part of the inverter capacity: **84 %**
Nb. strings in parall.: 1 ☑ Nominal 1
Pnom ratio **1.05** Overload loss **0.0 %**
nb. modules **5** Area **8 m²**

Tigo voltages at PNom
Vmpp (60°C) 139 V
Voc (-10°C) 231 V

Plane irradiance **1000 W/m²**
○ Max. in data ● STC
Max. operating power (at 1000 W/m² and 50°C) **1.4** kW
Array nom. Power (STC) **1.6** kWp

List of subarrays

Name	#Mod #Inv.	#String #MPPT
10deg M2		
IBC Solar - IBC Monosol 315	5	1
Solaxpower - X1-Boost-3.0kW ...	1	1

Global system summary

Nb. of modules	5
Module area	8 m²
Nb. of inverters	0.5
Nominal PV Power	1.6 kWp
Maximum PV Power	1.1 kWDC
Nominal AC Power	1.5 kWAC
Pnom ratio	1.050

Figure 5.5: Screen dialog of the system window in PVsyst. Here the sub-array is set up.

Figure 5.6 is a string diagram for the 10° mono c-Si and poly c-Si modules. The figure presents an overview of the modules stringed together and the inverters connected to the modules. The five mono c-Si modules simulated in PVsyst are grey and marked by the red square. They are connected to inverter INV8-1.

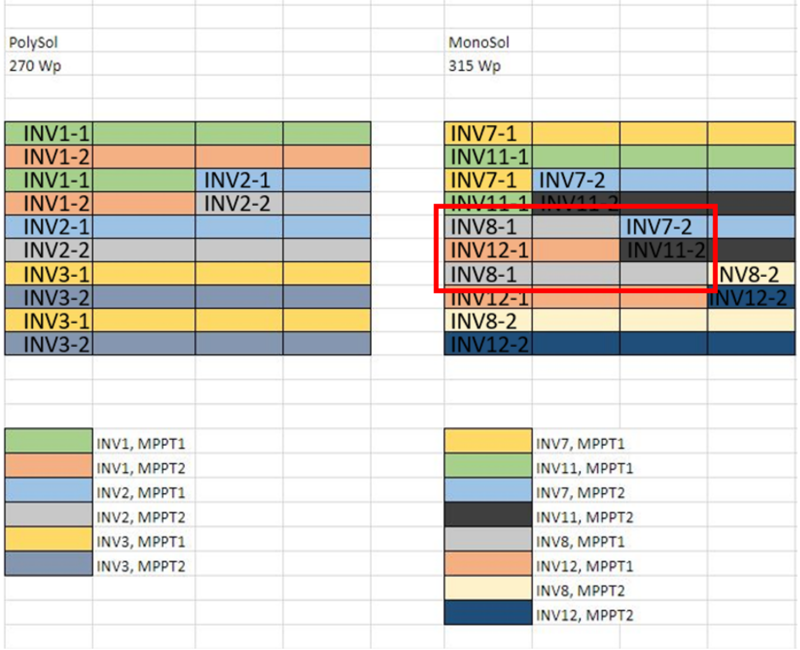


Figure 5.6: The grey modules circled by the red square are the five mono c-Si PV modules simulated in PVsyst. They are connected to inverter INV8-1.

Optional Parameters

The optional parameters that can be specified in the system window, help make the simulations more accurate.

The *Horizon* tab describe far shadings located at a sufficient distance from the PV system. This is referring objects not directly casting a shadow on the modules, but they block some of the irradiance from the Sun, such as mountains in the distance. The horizons curve may be defined manually based on measured angles or imported from e.g., Meteonorm. The roof at UiA is the same height as all surrounding buildings and treetops, and without any surrounding mountain areas, and the impact from the horizon shading is therefore negligible.

The *Near shadings* tab define the objects close to the PV system, causing shading on the modules. These are implemented through a 3D model of the PV system and its surroundings. The object has to be manually drawn or altered from sketches in PVsyst. Figure 5.7 show the 3D model of the PV system at UiA. Two losses must be considered from the near shading model: irradiance losses and electrical losses.

Irradiance losses take the deficit of irradiance on the PV module into account. The losses are caused by nearby objects causing direct shadow, reducing the direct and diffuse irradiance on the modules.

Electrical losses is caused by the mismatch of electrical response of the modules in series and strings in parallel.

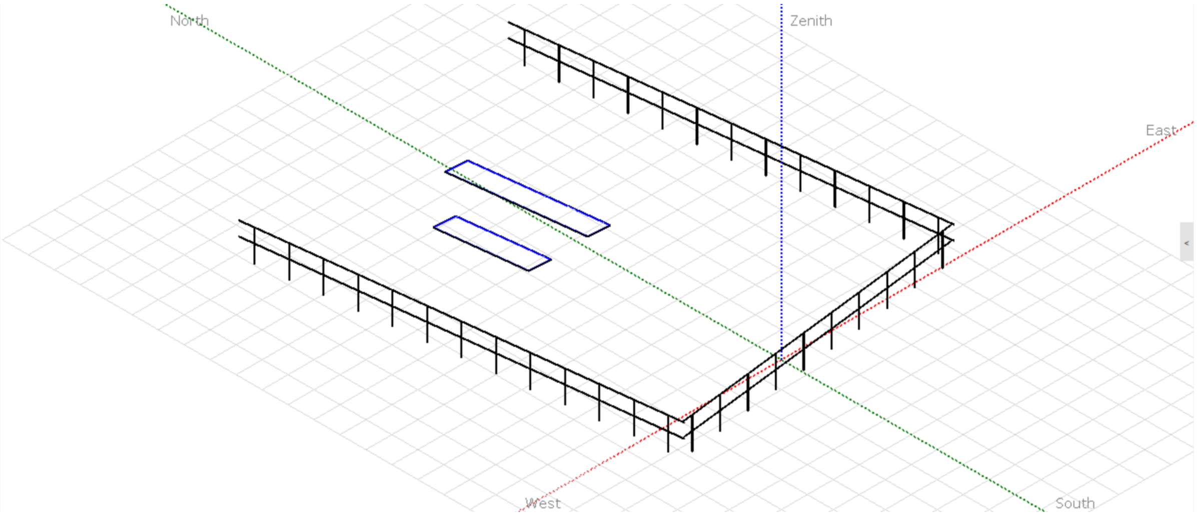


Figure 5.7: 3D model of the near shadings experienced by the five mono c-Si modules at UiA.

In the *Module layout* tab the geometrical arrangement of the PV modules and their interconnections as strings may be specified. By defining these specifications, a more accurate calculation of the electrical shadings mismatch loss may be obtained. The module layout is defined based on the 3D shading scene.

5.5 Detailed Losses

Under *Detailed losses* several factors impacting the losses in the PV system may be specified, or the default values provided by PVsyst may be used. Most values were set to default in this simulation.

Thermal Parameter

To estimate the module temperature, PVsyst uses eq. (5.1). Constant loss factor (U_c) is by default set to 20.0 W/m^2K, and the wind loss factor (U_v) is set to 0.0 $W/m^2K\ m/s$. The thermal loss factors may also be chosen to default values according to mounting, where the options are: "Free" mounted modules with air circulation, semi-integrated with air duct behind, and integration with fully insulated back.

$$U = U_c + U_v * WS \tag{5.1}$$

where U_c is the constant loss factor [W/m^2K], U_v is the wind loss factor [$W/m^2K\ m/s$], and WS is the wind speed [m/s].

For the simulation at UiA the factors are set to:

- $U_c = 25\ W/m^2K$
- $U_v = 1.2\ W/m^2K\ m/s$

These values are based on the research done by the author in ENE503 Energy Research Project [8]. These results was based on a recommendation from PVsyst [74].

Ohmic Losses

The ohmic losses refer to wiring circuit losses between the power available from the PV modules and the power at the terminals of the sub-array.

$$E_{loss} = R_w \cdot I^2 \tag{5.2}$$

where R_w is the equivalent resistance of the wire [Ω], and I is the current [A]. The ohmic losses are only accounted for if the wire length and cross section is defined. If else, the power loss ratio is based on a loss fraction of STC, at a default value of 1.5 %.

Soiling Losses

Soiling losses are losses caused by dirt accumulating on the front of the PV modules. The effect of soiling on the PV system strongly depends on the environment on the system, such as rain conditions etc. Soiling losses can be set to default, yearly loss %, or monthly values. In this simulation they are set to zero.

IAM Losses

IAM or *Incident Angel Modifier* corresponds to the amount of irradiance actually reaching the PV cell's surface due to reflection loss and transmission loss through the module transparent cover. The decrease is mostly affected by the type of glass used in the module, and can be imprved by the use of antireflection coatings or structured surfaces on the glass and cell. The IAM in the simulation is set to default value by PVsyst, according to specifications of module characteristics.

System Unavailability

Under system unavailability, downtime due to system failure, maintenance, or downtime of the grid may be defined. This loss can be defined as fraction of time, number of days or set to random. This loss is not accounted for in this simulation to compare the recorded system to a system at 100 % data availability.

6 Results and Discussion

6.1 Data Filtering

Figure 6.1 shows PR calculations for mono c-Si and poly c-Si at a 10° tilt, both East- and West-facing, without data filtering. The figure shows a stable PR_{DC} for the months March to November 2020. The calculations, without data filtering consist of data collected throughout both the day and night, and then summed together. Figure 6.2 shows the PR calculations, after data filtering, for mono c-Si and poly c-Si at a 10° tilt, both East- and West-facing.

The filtered PR_{DC} calculations show a slight equalization of the PR_{DC} values. The poly c-Si module facing West shows the overall lowest PR_{DC} values. For both calculations the PR is stable from March 2020 to October 2020 for all modules, with a slight drop in the summer, June, and July 2020. The PR_{DC} after data filtering still experiences a substantial drop in the winter months. This is due to several factors. The first reason is the lover data availability for these months. Another reason is the lower sun angle in the winter, which increases the amount of reflection on the top glass of the PV module. Additionally, the low sun angle also increases the shadow on the PV modules from nearby objects. Soiling is a third explanation of low performance. Dust and other objects may accumulate on the PV modules, preventing the irradiance from reaching the modules. For the winter months, snow is one of the main reasons for low performance [55]. Figure 6.1 also include data availability for all four modules. This shows that there is a clear correlation between the percentage data availability and the unfiltered PR_{DC}.

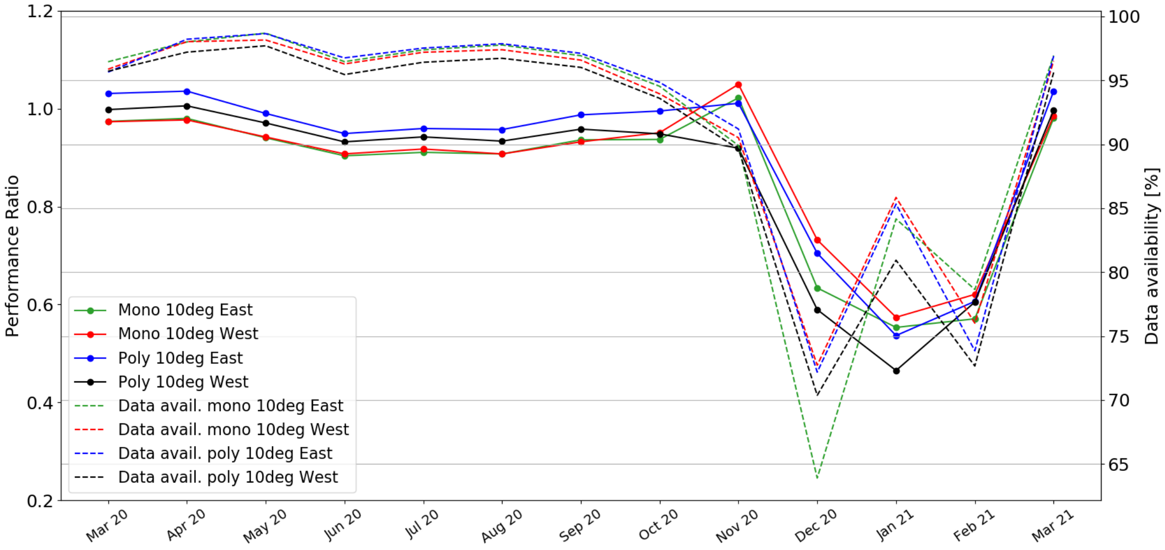


Figure 6.1: Performance ratio (PR_{DC}) for mono c-Si and poly c-Si at 10° tilt, both East and West without data filtering. Solid lines show the PR_{DC}, and the stippled lines show the data availability (left axis).

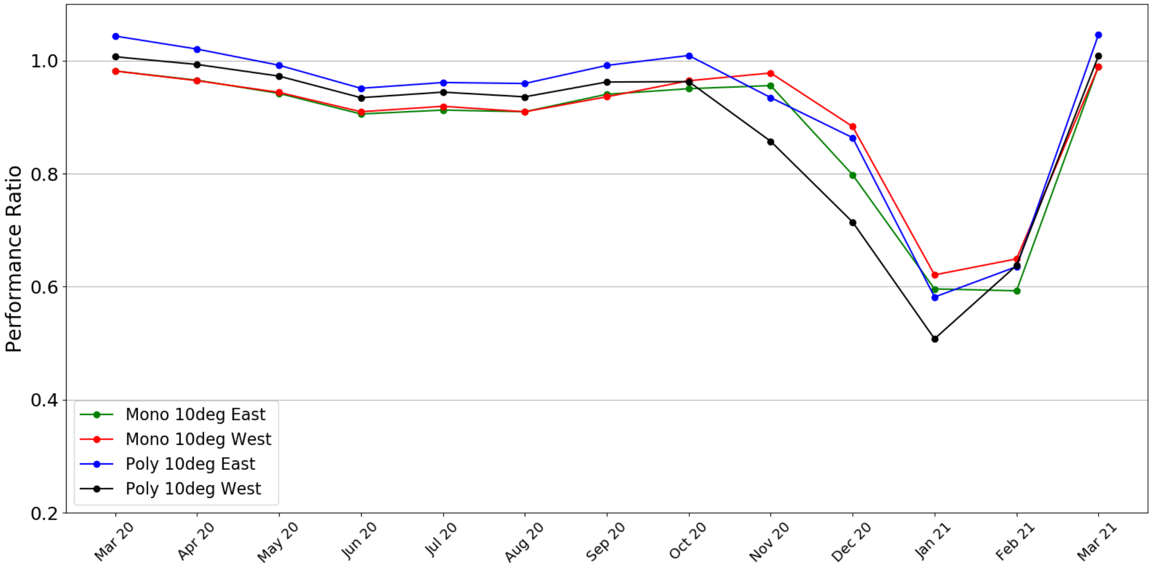


Figure 6.2: Performance ratio (PR_{DC}) for mono c-Si and poly c-Si at 10° tilt, both East and West, calculated with data filtering.

Figure 6.4 shows PR_{DC} without data filtering for mono c-Si and poly c-Si at a 45° tilt. Here the values vary more than for the 10° tilt. This is due to a larger variation in data availability as seen in figure 6.4. The large drop in September can be correlated to very low data availability for this month.

Figure 6.5 shows the PR_{DC} for 45° tilt, after data filtering. As expected, the PR_{DC} is more stable throughout the year. It experiences the same drop in the summer months, June, and July, as the modules at 10° tilt. In December, the poly c-Si module reaches the value of PR_{DC}= 1, which is very high. The average PR value is 0.859 for the mono c-Si module, and 0.909 for the poly c-Si module. The poly c-Si PV module shows an overall better PR value for the 45° tilt, with a mean deviance of 0.05. In contrast to the modules mounted at a 10° tilt, the modules at 45° tilt do not experience a drop in performance during the winter months. At a higher tilt angle, soiling losses due to snow are lower, see pictures in Appendix A. Large amounts of the snow on the PV modules slide off due to the high tilt angle. Additionally, the high tilt angle decreases the loss due to a lower sun angle in the winter.

It can be seen for all calculations conducted on the mono c-Si and poly c-Si modules mounted at 45° tilt angle that the poly c-Si module obtain a higher PR_{DC}. Usually, it is expected that the mono c-Si module would show higher specific yield due to a higher module efficiency than poly c-Si, and the PR values are expected to be similar under the same installation conditions. When this was discovered, the question of why presented itself. After some investigation, it was discovered that the measurement instruments for irradiance and wind speed mounted next to the module cast a shadow on the mono c-Si module late in the evening, as seen in figure 6.3. The mono c-Si module is mounted at the end of the rack, while the poly c-Si is mounted in the middle of the rack and is therefore not affected by the shadowing. This may explain some of the lack in performance for this exact module.

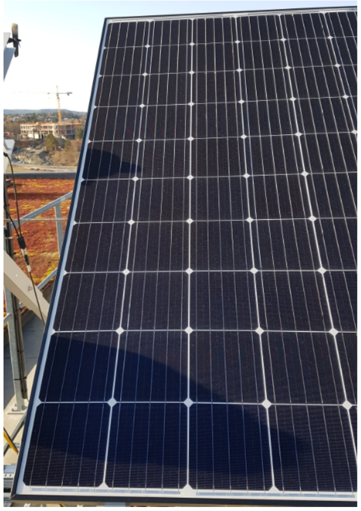


Figure 6.3: Picture of the shadow on the 45° tilt mono c-Si module at 5pm.

The temperature coefficient for the two module types, mono c-Si and poly c-Si is almost identical and will not explain the difference in performance. For both the 10° tilt and 45° tilt modules assessed, the poly c-Si module obtains a better performance. One possible explanation is that the rated power of the mono c-Si is slightly lower, and/or the rated power of the poly c-Si is slightly higher, than the stated value from the manufacturer. The tolerance of the rated power is -0/+5 [Wp] for both PV modules. Another possible explanation is that the mono-Si modules have degraded more than the poly-Si modules, but this would not be expected. This could be tested by individual I-V curve measurements of the modules.

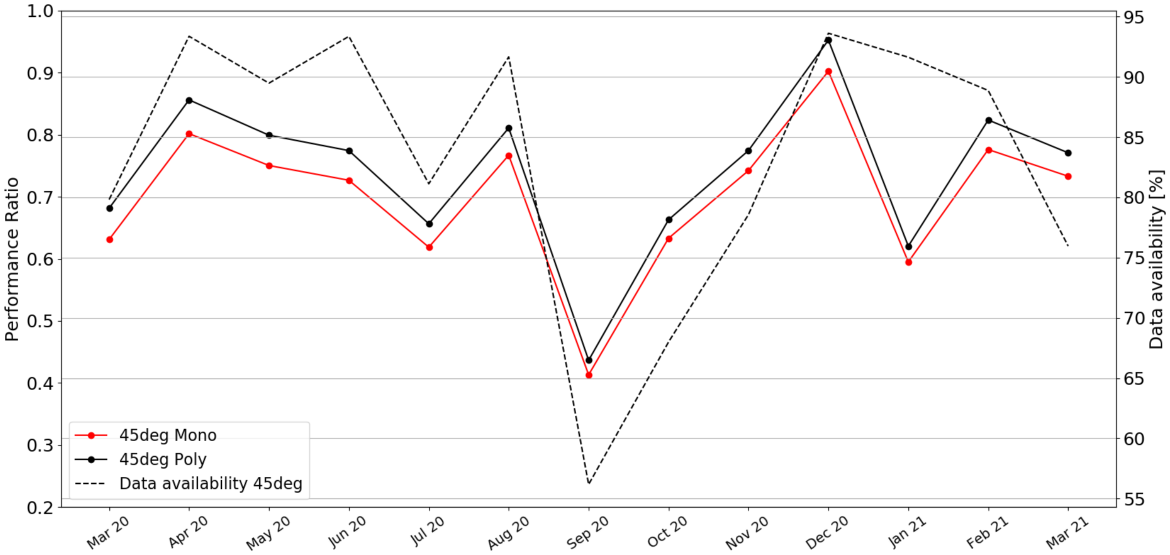


Figure 6.4: Performance ratio (PR_{DC}) for 45° tilt mono c-Si and poly c-Si without data filtering. Solid lines show the PR_{DC} (left axis), while the stippled line show the data availability (right axis).

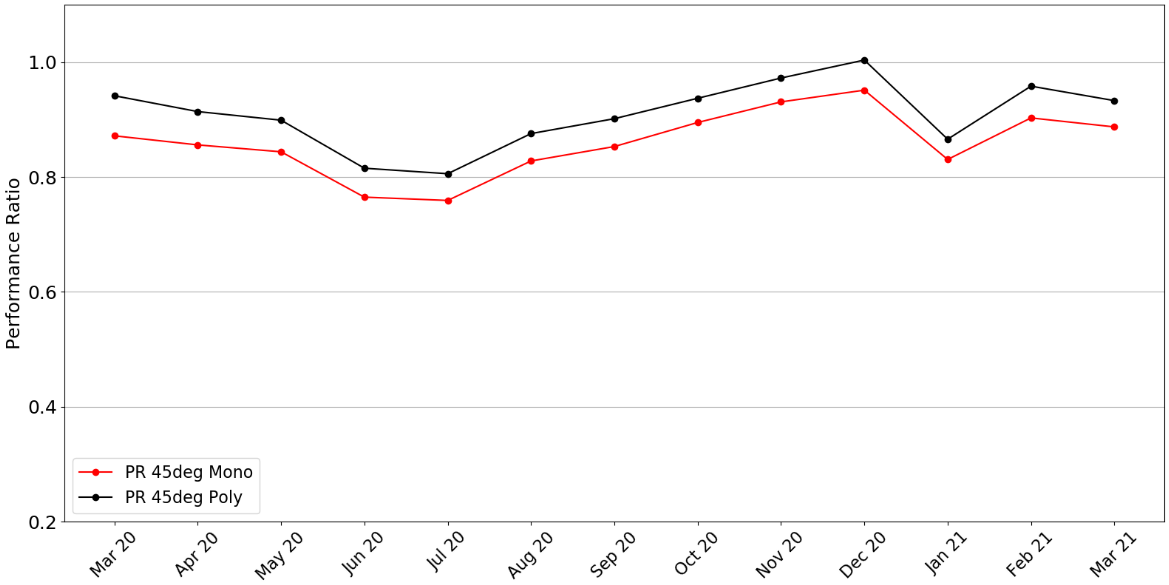


Figure 6.5: Performance ratio (PR_{DC}) for 45° tilt mono c-Si and poly c-Si, with data filtering.

Figure 6.6 and 6.7 shows PR_{DC} calculations for both the normal and the grey-tinted BIPV modules mounted at a 90° tilt. Both modules follow each other closely both with and without data filtering. Without data filtering the PR_{DC} values vary a lot throughout the year, with the same drop in September 2020 as for the 45° tilt modules, due to low data availability. The stippled data availability line shows the correlation between data availability and the PR_{DC}, or the performance of the module.

After the data is filtered the PR show a smoother graph with an average PR_{DC} of 0.866 for the normal BIPV module, and 0.871 for the grey BIPV module. The values decrease slightly in the summer and increase slightly in the winter. This is due to the module temperature affecting the performance of the PV modules. In addition, the higher sun angle in the summer results in less irradiance hitting the PV module mounted at 90° tilt, and vice versa in the winter, as discussed in chapter 3 *Literature review*. As for the 45° tilt modules, the BIPV modules do not experience a problem with snow in the winter months, due to the high tilt angle.

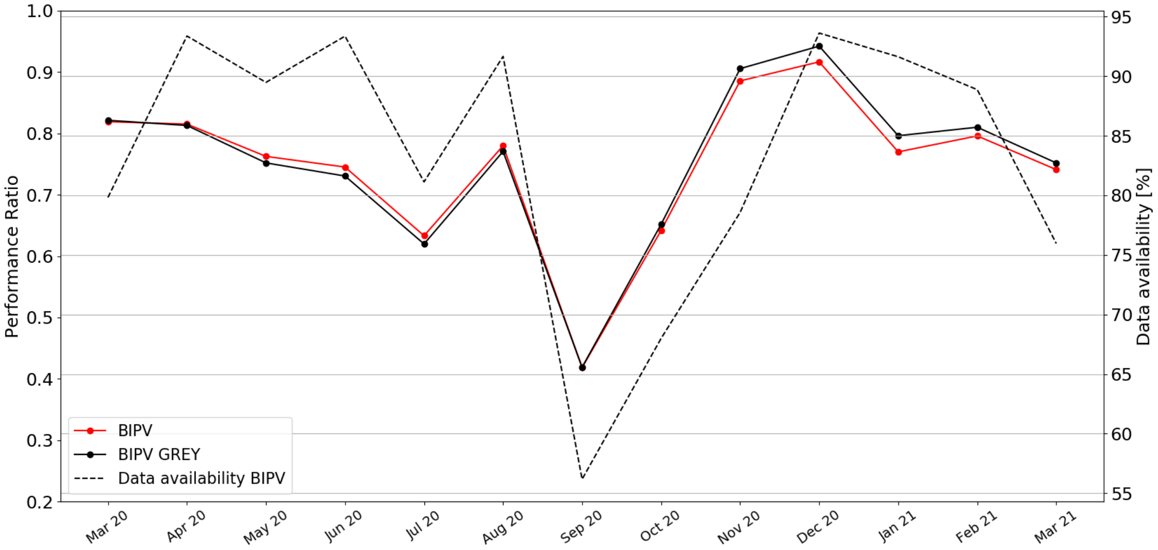


Figure 6.6: Performance ratio (PR_{DC}) for normal BIPV and grey BIPV without data filtering. Solid lines show the PR_{DC} (left axis), while stippled line show the data availability (right axis).

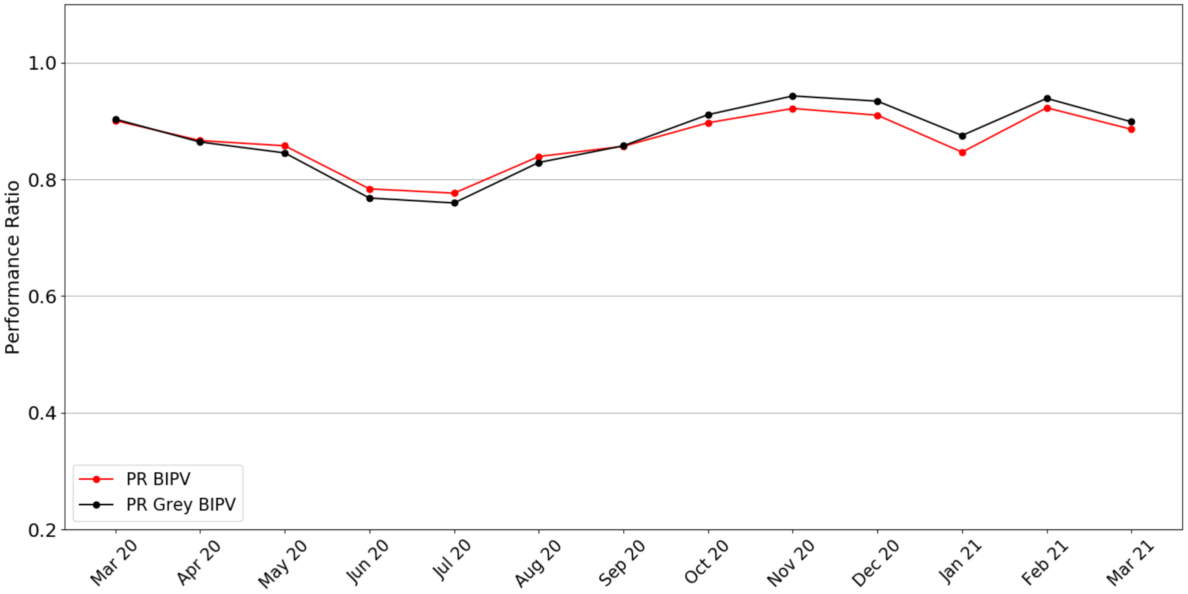


Figure 6.7: Performance ratio (PR_{DC}) for normal BIPV and grey BIPV, with data filtering.

All data sets show a significantly improved PR_{DC} with data filtering than without, obtaining a relatively stable value throughout the year. They all show the same trends, with a slight decrease in the summer. The average PR_{DC} values are further discussed in chapter 6.3.

6.2 Specific Yield

Figure 6.8 shows the specific yield using filtered data for all eight PV modules. The four modules mounted at a 10° tilt, East- and West-facing, show a better specific yield than the four modules mounted at a 45° tilt and 90° tilt for the months from March 2020 through September 2020. For the winter months, from October 2020 to January 2021 the modules mounted due South show a better specific yield. This can be explained with the higher tilt angle and, thus, a better performance with low sun angles. However, since there is a large variance in data availability between the 10° tilt and 45° tilt/BIPV modules, comparing the modules in a plot like this is not a fair representation. The low data availability for the 45° tilt and BIPV modules results in a lower specific yield, and a conclusion for what installation angle performs the best cannot be obtained by these results.

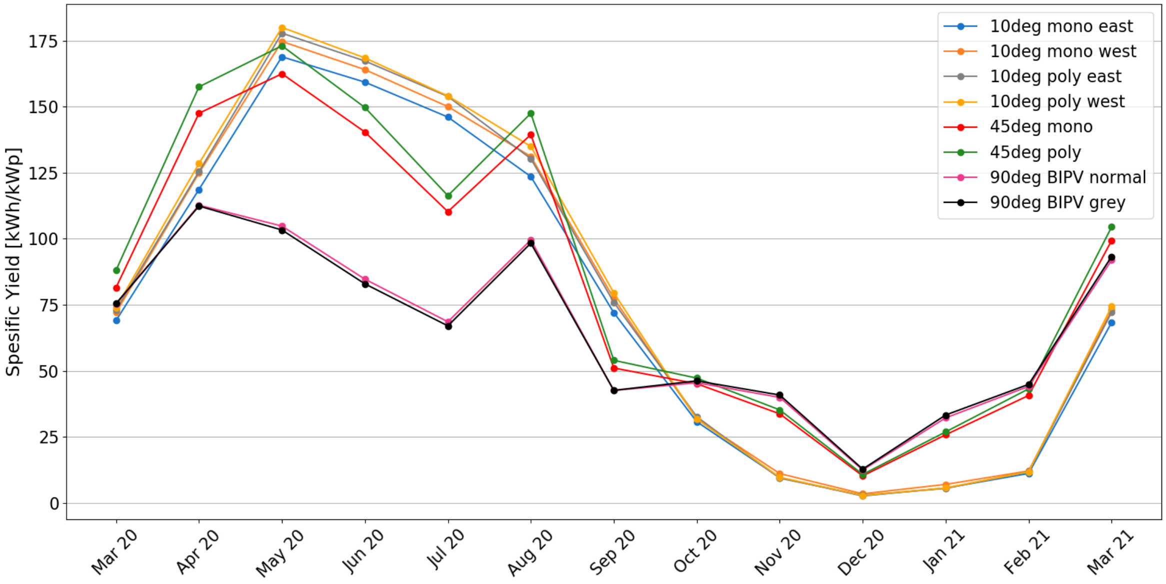


Figure 6.8: Monthly specific yield for all eight PV modules for 13 months, from March 2020 to March 2021. Calculations are conducted using filtered data but with differences in data availability for the 10° versus the 45° and BIPV datasets.

6.2.1 Estimated Specific Yield

As described in chapter 4.6, estimated output power has been calculated for the 45° tilt modules and the BIPV modules, based on the irradiance measurements, to fill in the gaps in data availability. Thus, a specific yield based on the estimated output power has been calculated, see figure 6.9.

This figure shows a better correlation between the modules mounted at a 10° tilt angle and the 45° tilt/BIPV modules. Compared to figure 6.8, the specific yield for 45° and both BIPV now follow the 10° tilt curve more closely. There is especially a large improvement in specific yield for the months from March 2020 to October 2020. The best performing module is now the poly c-Si module at 45° tilt. The BIPV modules still produce a lower specific yield than the other PV modules due to the higher tilt angle, however it has a more even production profile throughout the year and performs better in winter.

Figure 6.9 shows a lower specific yield for both 45° and BIPV modules in June and July 2020 than anticipated, based on the results for the 10° modules. It is not clear why these two months deviate from the expected result. When estimating output power, the same python code has been used for all months. In addition to replacing missing values, a code has been written to delete measured power (P_{DC}) values below 0.05 W in order to replace power values around zero that were proven to be false values by comparison with irradiance data. When investigating the June data, three days of values below 0.05 W were detected, however, removing these values and estimating power based on irradiance did not improve the result, as anticipated. As these corrections did not solve the problem completely, some unknown factor is influencing the results and will require further investigation.

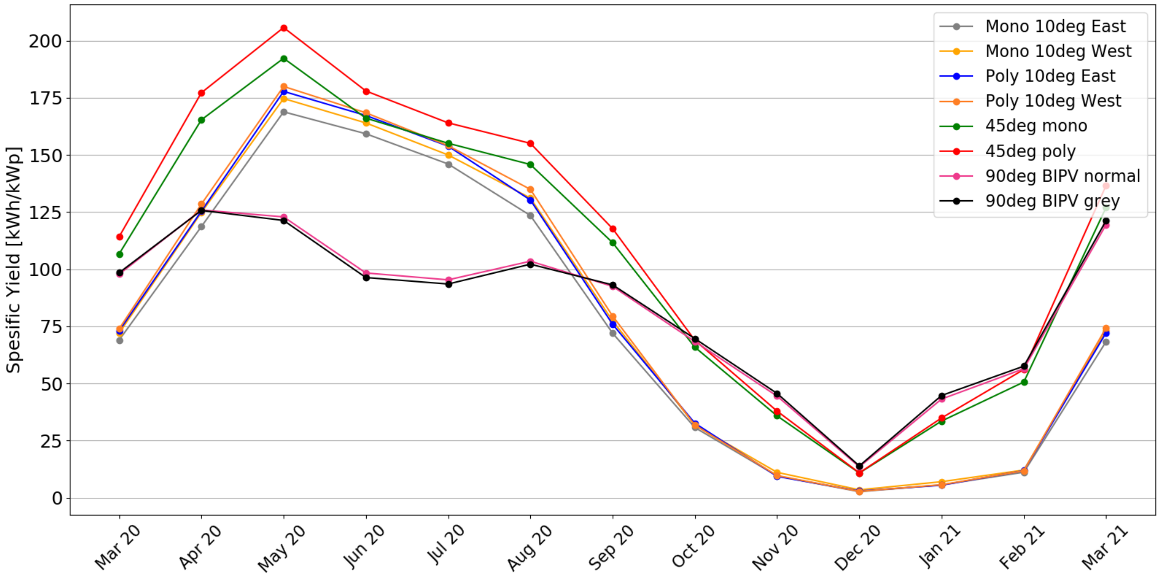


Figure 6.9: Specific yield for all eight PV modules using filtered data and with estimated power output for the missing data points for the 45° tilt and BIPV modules.

6.2.2 Annual Specific Yield

Figure 6.10 shows the annual specific yield for all eight PV modules for 12 months, from March 2020 to February 2021, using filtered data. Results are given both without and with the power estimation of missing data points. As expected, all four 10° tilt and 45° tilt modules gives very similar values for specific yield, while the normal and grey BIPV show a lower annual specific yield. Compared to other PV systems operating in Norway the system at UiA show a relatively good performance. The AC specific yield (or final yield) for south-facing BIPV facades was, as discussed in chapter 3 *Literature review*, found to be between 500 and 770 kWh/kWp in similar locations in Norway. The two BIPV modules at UiA obtain a DC specific yield (or generator yield) of approximately 760 kWh/kWp. This is within the expected performance range, even with the unexplained lower yield seen during the summer months of 2020. Without data estimation the 10° tilt modules and 45° modules all obtain a specific yield between 920 - 1050 kWh/kWp. These results are similar to other PV installations in Norway, as discussed in chapter 3, when taking into account that the latter refer to AC specific yield (with system losses), whereas the results in this thesis are DC specific yield. When calculating the specific yield with data estimation for missing data points, the specific yield results in 1171 kWh/kWp (45° mono c-Si), 1244 kWh/kWp (45° poly c-Si), 926 kWh/kWp (BIPV normal), and 925 kWh/kWp (BIPV grey).

The best annual specific yield is obtained by the poly c-Si mounted at a 45° tilt. As discussed previously, it was discovered that the mono c-Si at 45° tilt experienced a shadow during parts of the day, thus, reducing the performance but also other factors could contribute to this deviation, as discussed above. The result is not unexpected, as 45° south is the estimated optimum tilt angle for maximum annual energy yield at this location.

Figure 6.10 also shows the annual specific yield using estimate output power for the 45° tilt and BIPV modules. Here it is clear that the performance with full data availability is substantially better than with the actual data availability obtained for this time period. For the 10° tilt modules, the data availability is above 90 % from March 2020 to October 2020. There has therefore not been estimated any output power for missing data points for these four modules. Based on results from the 45° tilt and BIPV modules, it is expected that the annual specific yield would increase if the data availability for the 10° tilt was 100 %.

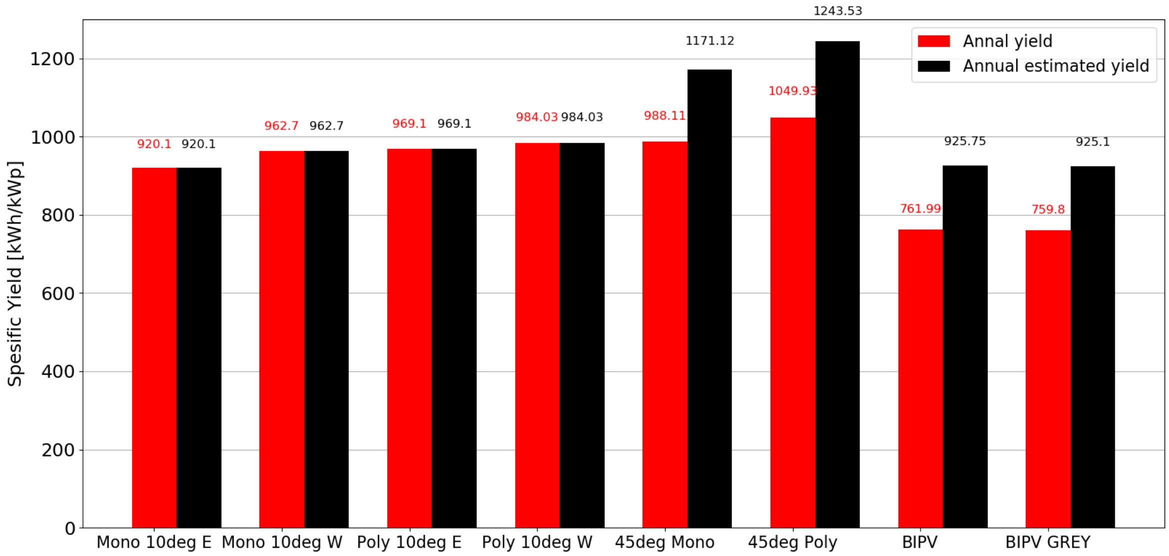


Figure 6.10: Annual specific yield for all eight PV modules in the period from March 2020 to February 2021. The red bars are the annual specific yield with measured data, and the black bars are the annual specific yield with the estimated output power.

6.3 Performance Ratio

A plot displaying all performance ratios compared is given in figure 6.11. The PR values are calculated based on the filtered data, without correction for data availability. From March 2020 to September 2020 all eight modules follow the same trend, with only a small difference in value. However, from October 2020 to January 2021 the values differ depending on tilt angle and orientation. The modules mounted at a 45° tilt and the BIPV modules at a 90° tilt follow the same trend, while the modules mounted at a 10° tilt experience a decrease in December 2020 and January 2021. PV modules mounted at a higher tilt angle experience better performance in the winter. The low installation angels result in a larger reflection in the top glass cover of the 10° tilt modules. Also, as explained previously, soiling losses is a factor affecting the performance of the PV modules. The best PR_{DC} is obtained by the poly c-Si PV modules mounted at a 10° angle.

6.3.1 Temperature Corrected Performance Ratio

Due to a lack of module temperature measurements for the 45° tilt modules and the BIPV modules, there is no PR_{DCcorr} for March and April 2020. Figure 6.12 shows the PR_{DCcorr} for all eight modules compared. From the figure, it is clear that modules with the same tilt angle follow the same path. The modules at 45° tilt and the BIPV modules experience a decrease in the summer months June and July 2020, which corresponds with the decrease in specific yield as discussed previously. The PV modules mounted at a lower tilt angle obtain a lower PR_{DCcorr} during the winter. For the 10° tilt, all modules show the same general trend from March to October 2020. The 10° tilt poly c-Si module show a higher PR_{DCcorr} for all months from March to October and December 2020. In January 2021 there is a large decrease for all PV modules. This is most likely due to snow covering the modules during this month, in addition to a lower data availability. For the 45° tilt, the poly c-Si obtain a better PR by about 0.05 for all months. The best PR_{DCcorr} was obtained in December 2020 for the mono c-Si at 45° tilt with a value of 1.018. A degree of uncertainty is linked to the PR_{DCcorr} calculation in the winter months, especially the very high value for December 2020. Very low irradiance or frost/snow covering the irradiance sensors may result in overestimated values.

By correcting the performance ratio for temperature, the seasonal-variation in the PR is removed, allowing the assessment of other factors affecting the PV module. When the temperature dependency is removed from the measurements, a smoother PR_{corr} graph is expected. Especially in the summer, with no snow covering the modules. This is seen in figure 6.12 for the modules mounted at 10° tilt, where all modules obtain a PR_{DCcorr} between 0.9 - 0.95 from March to October 2020. However, for the 45° tilt and BIPV modules this is not the case, indicating that other factors are affecting the performance of the PV modules.

6.3.2 Estimated Performance Ratio

Figure 6.13 shows a comparison of the normal PR_{DC} for the 10° tilt modules, and PR_{DCest} calculated using estimated output power values for the 45° tilt and BIPV modules. Temperature has been estimated for March and April 2020, and PR_{DCest} has been calculated. The PR_{DCest} in figure 6.13 is calculated as normal PR, not temperature corrected.

The PR_{DCest} shows a more even graph throughout the year, than the previous calculated PR_{DC} and PR_{DCcorr}. For June and July 2020, the drop seen in figure 6.11 is improved. This indicates that the low performance was mainly because of low data availability rather than high PV module temperatures. Also, the drop in January 2021 is reduced. Even though the 45° tilt and BIPV modules are now at presented with 100 % data availability, the 10° tilt still achieves a higher performance ratio from March to October 2020.

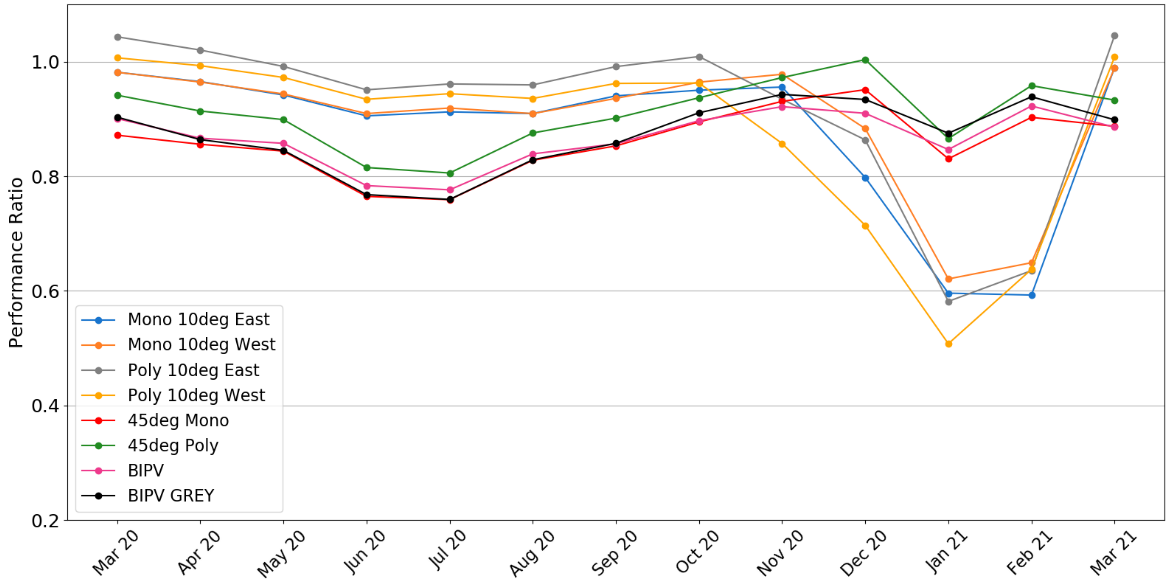


Figure 6.11: Performance ratio (PR_{DC}) for all eight PV modules. Calculated with filtered data, without temperature correction or correction for data availability.

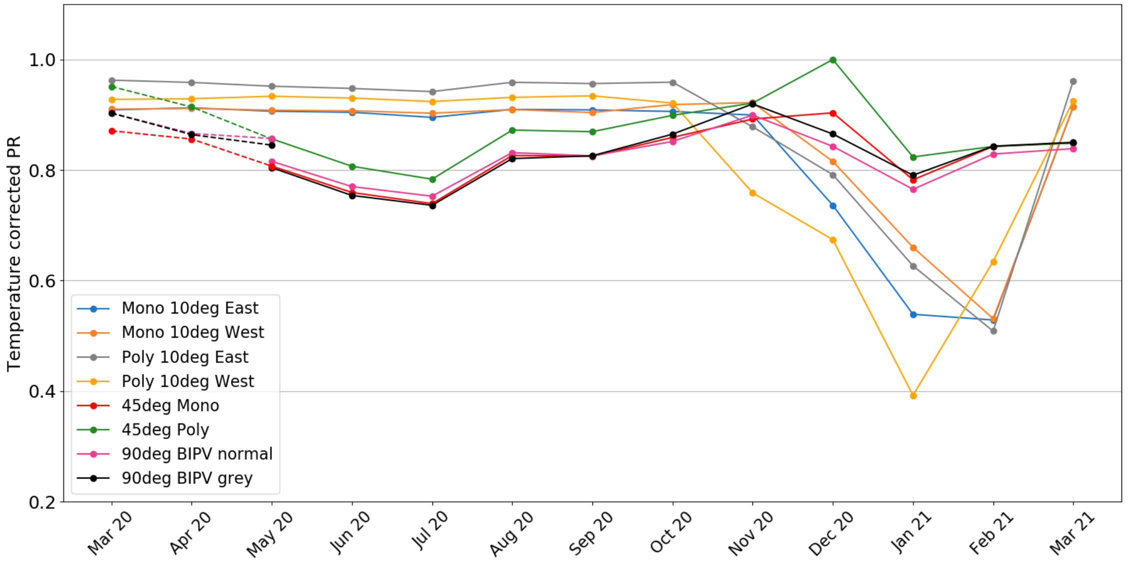


Figure 6.12: Temperature corrected performance ratio (PR_{DCcorr}) for all eight modules. Calculated with filtered data. The two months of March and April 2020 (stippled lines) are uncorrected for the 45° and BIPV modules due to missing PV module temperature data.

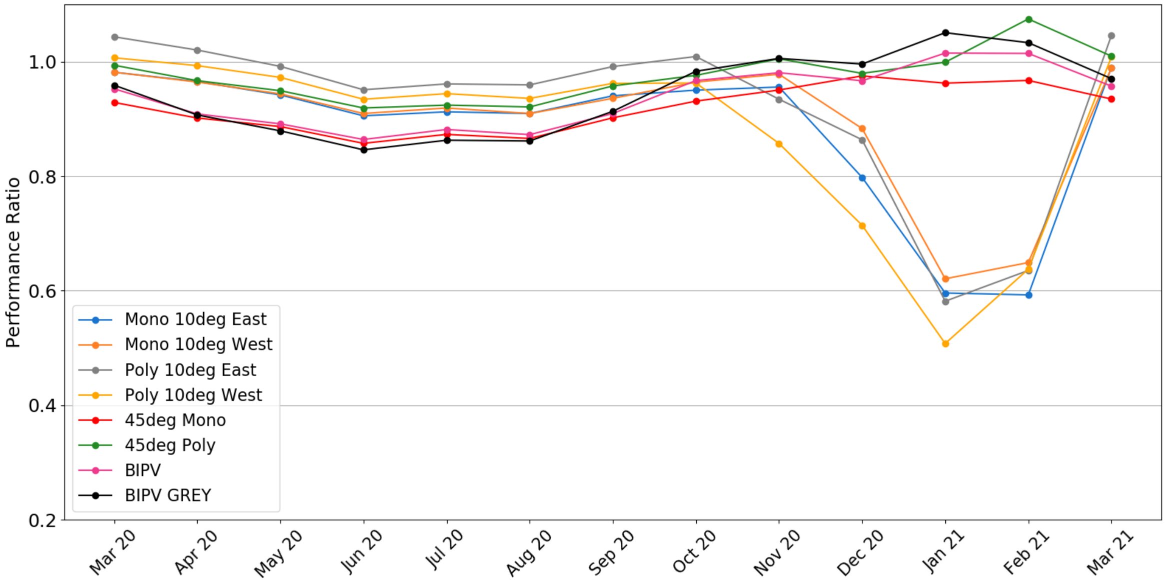


Figure 6.13: Performance ratio (PR_{DCest}) calculated with estimated power output values for the 45° tilt and BIPV modules. Without temperature correction.

Table 6.1 shows an overview of the mean values for all eight PV modules, comparing normal PR_{DC}, PR_{DCcorr}, and PR_{DCest} for the 45° tilt and BIPV modules. All calculations are conducted with data filtering and DC values. Here it is clear that there is not much deviance between the modules in normal uncorrected PR_{DC}, with regards to tilt angle or orientation. From table 6.1 it is clear that the PV module achieving the best PR_{DC} is 45° tilt poly c-Si. For PR_{DCcorr} all systems experience a drop in mean PR value, which indicates that the PV systems typically operate at temperature conditions lower than STC. However, the overall assessment of the PV modules is better, and there is less deviance between the different tilt angles and orientations. The overall best PR for 45° tilt and BIPV is found when estimating output power for missing data, which is also the most correct indication of the module performances in the various installation angles. From these results, it is clear that the mean PR_{DCest} (March 2020 - March 2021) for the 10° tilt modules are in the range 0.88 - 0.90 (mono c-Si E/W) and 0.88-0.92 (poly c-Si W/E), while for the 45° tilt modules the values are 0.92 (mono c-Si) and 0.95 (poly c-Si), and for both the normal and grey BIPV modules at 90° tilt the value is 0.94. Hence, the performance ratio of the higher tilted modules is on average slightly better than for the near-horizontal installation angle, which is mainly due to the large difference in PR during winter.

PV Module	$\mathbf{PR}_{DC}$	$\mathbf{PR}_{DCcorr}$	$\mathbf{PR}_{DCest}$
Mono 10° E	0.880	0.836	0.880
Mono 10° W	0.896	0.851	0.896
Poly 10° E	0.896	0.873	0.896
Poly 10° W	0.880	0.833	0.880
45° Mono	0.859	0.826	0.918
45° Poly	0.909	0.865	0.948
90° BIPV Normal	0.866	0.820	0.937
90° BIPV grey	0.871	0.825	0.943

Table 6.1: Comparison of mean PR_{DC}, PR_{corr} and PR_{DCest} for all eight PV modules.

6.4 Power Loss Analysis

To thoroughly investigate how orientation and tilt angle, as well as environmental factors, affect the different PV modules and their loss factors, the DC output power curve from a clear sunny day was compared in more detail for the different PV modules. Because of the fence surrounding the perimeter of the roof, it was interesting to compare one module from the outer most PV row to one from the middle of the array. This gives a good indication of losses that are to be expected throughout the day based on shading loss due to location of the PV modules. All calculations are conducted using measured data from April 19th 2020, which was a clear sunny day with stable irradiance. Figure 4.2 shows an overview of where the modules are mounted relative to each other.

The location in the PV system of the modules assessed in the power loss analysis is given in figure 4.2. Figure 6.14 shows two mono c-Si modules U2 and V5 mounted at a 10° tilt, both West-facing. Module V5 is mounted at the perimeter of the grid-connected PV system and experiences a shadow from the railing from sunrise until 12 pm, when the sun is the highest in the sky. Apart from the shadow, both PV modules behave similarly. Both obtain the same temperature, and similar disturbances during sunrise/sunset. The same is observed for the M2 and K2 mono c-Si modules at a 10° tilt, East-facing, however the K2 module experiences shadow in the evening until sunset. Also, the M2 module, figure 6.15, show a disturbance in the morning. This may indicate that the module is exposed to shadow from the fence or other PV modules.

In all figures for power loss analysis, the blue graph in the plots is the measured DC output power for April 19 2020, and the red graph is the measured module temperature. In the green graph the temperature-dependency for the output power is removed by correcting the power for temperature by eq. (2.14). The black graph is power corrected for both temperature and irradiance at STC. This removes the impact of the irradiance, and should result in a flat, smooth graph. The last line in the graph is the yellow line for spectral factor correction. This correct for air mass outside STC AM 1.5, and should have the largest impact in the mornings and evenings. In addition to these graphs, each plot has a stippled grey line indicating the STC P_{mpp} for each module.

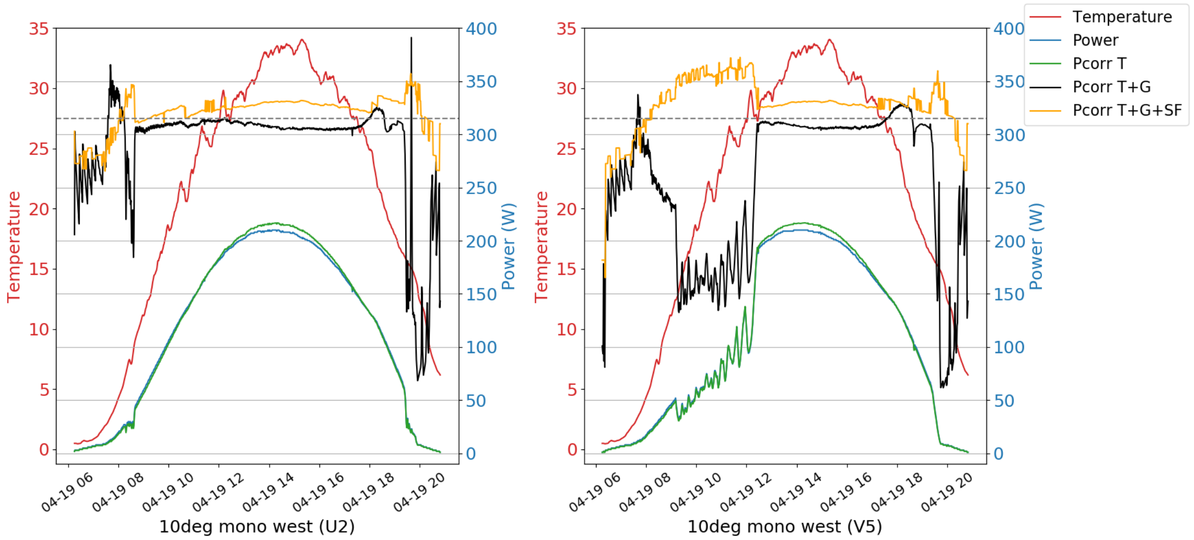


Figure 6.14: Two West-facing mono c-Si modules at 10° tilt, April 19. 2020. The V5 module (right) is exposed to shadow in the first part of the day. The blue and green lines show power and temperature corrected power, respectively. The red line show module temperature, the black line show power corrected for temperature and irradiance, and the yellow line show power corrected for temperature, irradiance, and spectral factor. They grey-stippled line is the STC-P_{mpp} for the modules.

In figure 6.14 and 6.15 the yellow line ($P_{corr\,T+G+SF}$) is smooth in the middle of the day, but has a lot of disturbance at sunrise and sunset. The disturbance in $P_{corr\,T+G+SF}$ increases when the module is exposed to shadow, see right side of figure 6.14. This is due to the DC/DC optimizer installed on the PV module, which tries to adjust the output power.

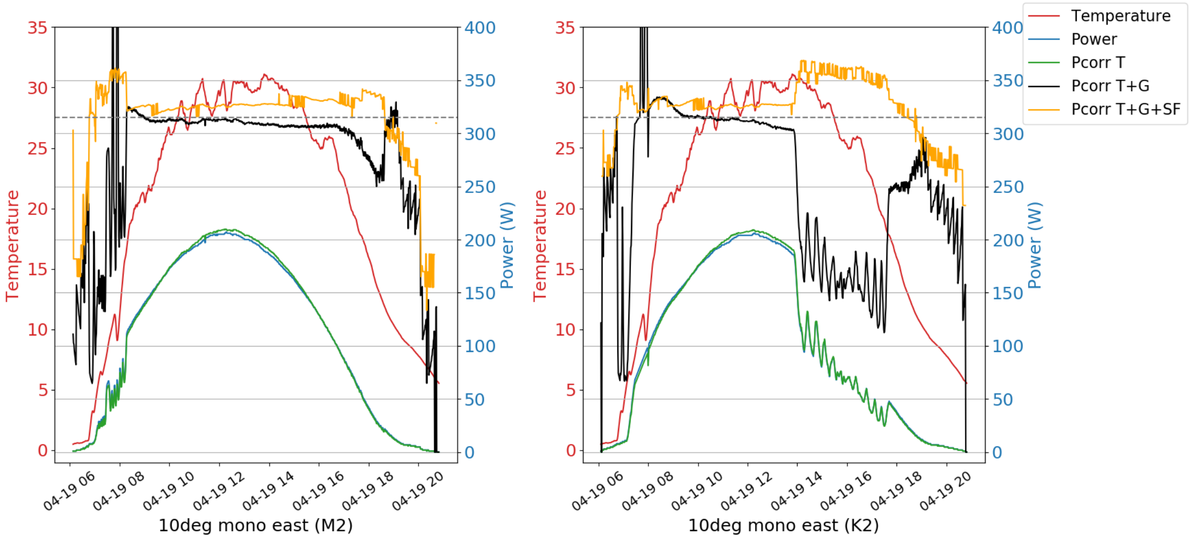


Figure 6.15: Two East-facing mono c-Si modules at 10° tilt, April 19. 2020. The K2 module (right) experience shadow in the afternoon. The blue and green lines show power and temperature corrected power, respectively. The red line show module temperature, the black line show power corrected for temperature and irradiance, and the yellow line show power corrected for temperature, irradiance, and spectral factor. They grey-stippled line is the STC-P_{mpp} for the modules.

For all modules included in the power loss analysis the temperature correction $P_{corr\ T}$ has little effect on the performance of the modules. This date in April still has pretty cold ambient temperatures, and the modules operate at module temperatures not far from STC, however, there is a small improvement in the middle of the day when the temperature is the highest. The next correction, for STC irradiance at 1000 W/m^2, has the largest impact on the performance. The irradiance in Grimstad rarely reach this high irradiance value. The last correction, spectral factor, has the largest impact in the morning and evening, as expected. As seen from figure 6.14, the spectral factor experiences the most uncertainties at sunrise and sunset, when sun angle is low. As a result of the $P_{corr\ T+G+SF}$ correction the power corresponds well with the STC P_{mpp} values for each module. For the modules at 45° tilt and BIPV modules there is a dip in the yellow line ($P_{corr\ T+G+SF}$). Ideally this line should be completely flat. Most likely due to cable losses, amongst other things, this line has a bend in the middle of the day, when the production is the highest. To obtain the best results the power should be corrected for this loss, however, due to time constraint this has not been done.

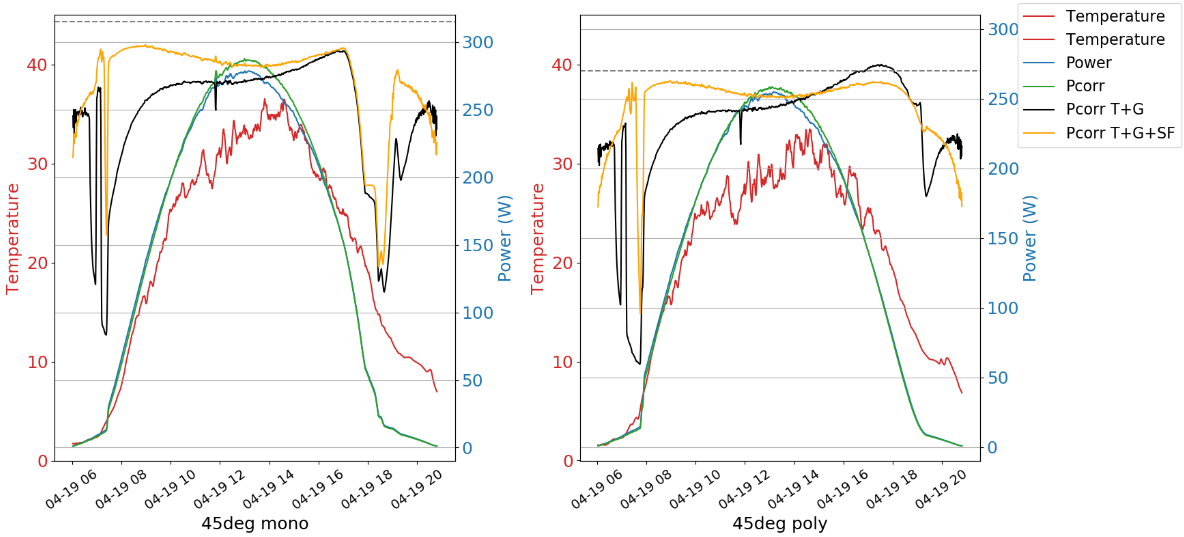


Figure 6.16: Power correction for both 45° PV modules, April 19 2020. Mono c-Si on the left and poly c-Si on the right. The blue and green lines show power and temperature corrected power, respectively. The red line show module temperature. The black line show power corrected for temperature and irradiance, and the yellow line show power corrected for temperature, irradiance, and spectral factor. They grey-stippled line is the STC-P_{mpp} for the modules.

As expected, the normal BIPV module produce a higher DC output curve than the grey BIPV module, while the grey module obtains a higher module temperature throughout the day. Regardless, both modules obtain a nice, even, output power graph. There is little deviance between the normal and the temperature corrected power curve. The power corrected for both temperate and irradiance ($P_{corr\,T+G}$) and spectral factor ($P_{corr\,T+G+SF}$) both stabilizes nicely in large parts of the day, after the initial sunrise and before sunset. The yellow line ($PR_{corr\,T+G+SF}$) is almost stable throughout the day, which is the desired output.

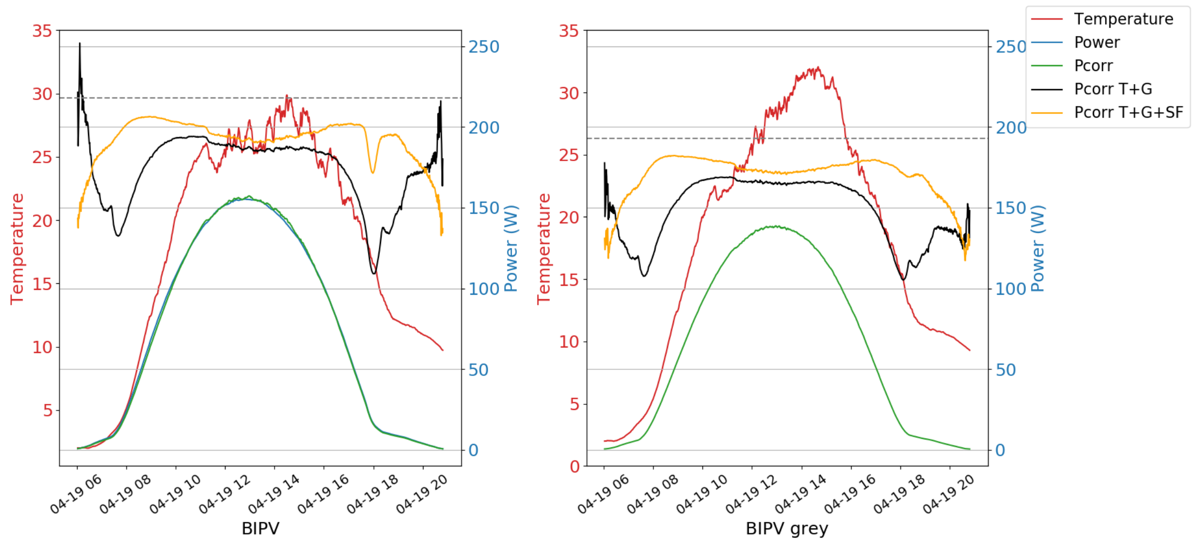


Figure 6.17: Power correction for both BIPV modules, April 19. 2020. Normal BIPV on the left and grey BIPV on the right. The blue and green lines show power and temperature corrected power, respectively. The red line show module temperature. The black line show power corrected for temperature and irradiance, and the yellow line show power corrected for temperature, irradiance, and spectral factor. They grey-stippled line is the STC-P_{mpp} for the modules.

6.5 Result of PVsyst Simulation

PVsyst produces a seven-page report of the simulation which may be found in Appendix B.

6.5.1 PV Performance

Figure 6.18 shows a screen dialog of the simulation of the five 10° tilted mono c-Si modules in PVsyst. The total simulated system production is 1436 kWh/year, and the specific production is 912 kWh/kWp/year. For the actual system, the recorded total system production was 1451.5 kWh/year and the specific production 910.2 kWh/kWp/year. Hence, the annual simulation results (AC) are very similar to the measured annual results (DC).

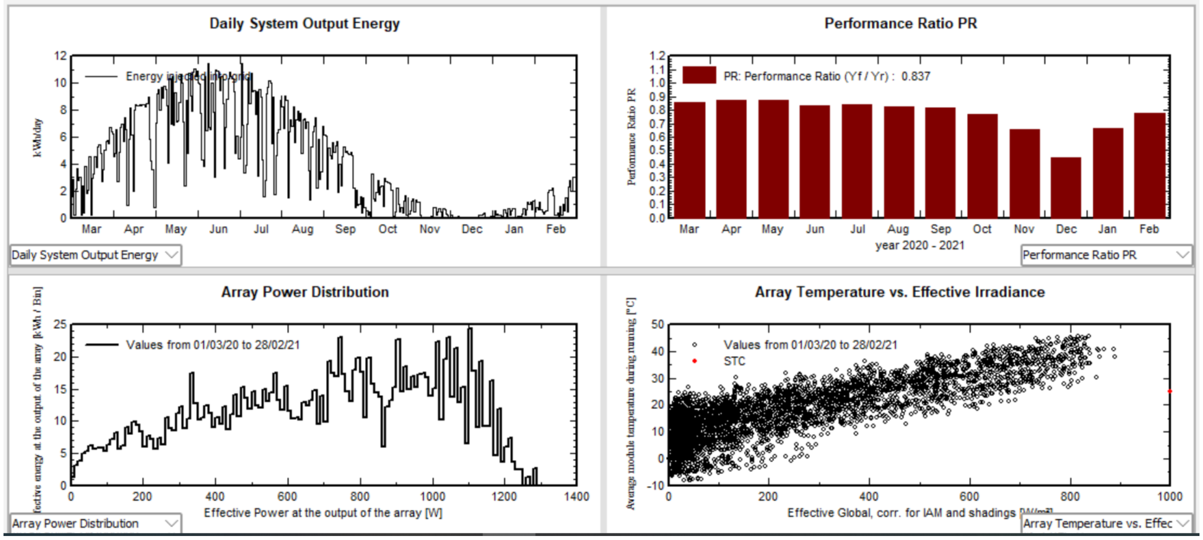

Daily System Output Energy
Energy injected into grid
kWh/day
0
2
4
6
8
10
12
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Daily System Output Energy
Performance Ratio PR
PR: Performance Ratio (Yf / Yr) : 0.837
Performance Ratio PR
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
year 2020 - 2021
Performance Ratio PR
Array Power Distribution
Values from 01/03/20 to 28/02/21
fective energy at the output of the array [kWh / Bin]
0
5
10
15
20
25
0
200
400
600
800
1000
1200
1400
Effective Power at the output of the array [W]
Array Power Distribution
Array Temperature vs. Effective Irradiance
Values from 01/03/20 to 28/02/21
STC
Average module temperature during running [°C]
-10
0
10
20
30
40
50
0
200
400
600
800
1000
Effective Global, corr. for IAM and shadings [W/m²]
Array Temperature vs. Effec

Figure 6.18: Screen dialog of result of simulation of the five 10° tilted mono c-Si modules from the PVsyst simulation for 10° mono c-Si.

As mentioned, the albedo value was set to 0.30 (concrete) in PVsyst, but soiling were set to zero. It not been considered that the system may experience snow, which has been the case in January and February. Therefore, the PVsyst result in a much higher yield value for these two months, than the calculations obtained with real, measured, data. Figure 6.19 shows a comparison of the simulated total energy production from PVsyst (AC values) and the actual energy production by the five PV modules (DC values). The results show almost similar values for all months, except January and February 2021 where the simulated system perform better than the real PV modules. As mentioned, the simulated system shows the total AC energy output as seen from the inverter, while the real PV modules show the DC energy output as seen from the PV modules. Hence, the real measured, energy output does not account for conversion losses from DC to AC, further discussed in chapter 6.5.2. The PVsyst simulation uses default values for loss calculations. The real system may therefore experience a different loss.

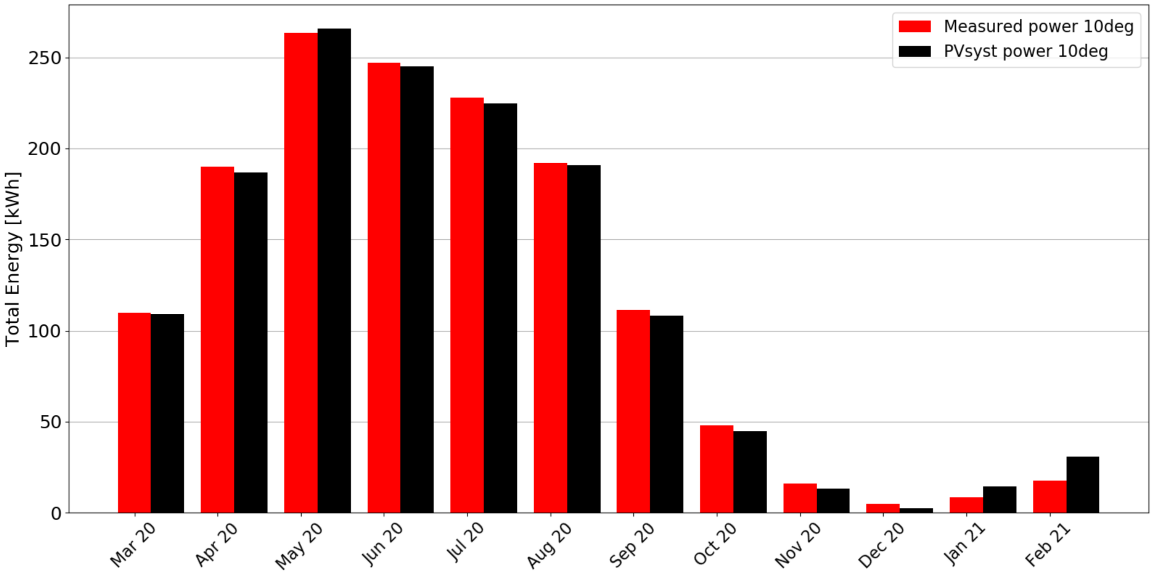


Figure 6.19: Comparison of total energy production from the five 10° tilted mono c-Si modules. The actual production by the system (DC values), and the simulated PVsyst energy output (AC values).

Table 6.2 shows monthly values for a set of performance parameters for the simulated 10° mono c-Si system. The Glob. hor. (global horizontal irradiance) values are based on measured values imported into PVsyst and is the sum of hourly values for each month. The yearly sum is the sum of all global horizontal irradiances throughout the year. The Tot. energy (total output energy) is the AC simulated output energy from the inverter. The final yield is also based on AC output values, in contrast to the calculated generator yield, which is based on DC values. PR is the PR_{AC} from the simulation, with an annual value of 0.837. Lastly T_{mod} is the simulated module temperature in $°C$. The simulated results from PVsyst in table 6.2 are compared with the measured data in figures (6.20 and 6.21) below.

Table 6.2: Monthly result for performance parameters from the PVsyst simulations of five mono c-Si PV modules at 10° tilt.

	Glob. hor. [kWh/m^2]	**Tot. energy [kWh]**	**Final yield [kWh/kWp]**	PR_{AC}	**T**$_{mod}$ **[**$°C$**]**
Mar 2020	80.7	109.1	60.44	0.859	10.76
Apr 2020	139.1	186.7	118.5	0.871	16.58
May 2020	197.1	265.7	168.64	0.871	20.90
Jun 2020	189.1	245.1	155.7	0.835	27.66
Jul 2020	171.2	224.9	142.91	0.843	23.92
Aug 2020	150.5	190.7	121.21	0.827	26.26
Sep 2020	87.0	108.0	68.7	0.818	20.46
Oct 2020	37.0	44.6	28.21	0.765	13.64
Nov 2020	13.4	13.2	8.4	0.656	10.24
Dec 2020	4.1	2.4	1.55	0.444	5.47
Jan 2020	14.8	14.4	8.99	0.663	0.46
Feb 2020	26.5	30.9	19.6	0.774	2.58
Year	**1110.6**	**1436**	**912.5**	**0.837**	**17.98**

Figure 6.20 shows a comparison of the calculated specific yield for all five PV modules and the simulated specific yield from PVsyst. The two systems show a near identical specific yield for the months March 2020 to December 2020, however, the simulated specific yield is based on AC values, while the calculated specific yield is based on DC values. For January and February 2021, the PVsyst simulated system show significantly better values. This may be because the simulated system does not take snow into account, while the real PV modules were affected by snow in parts of this period.

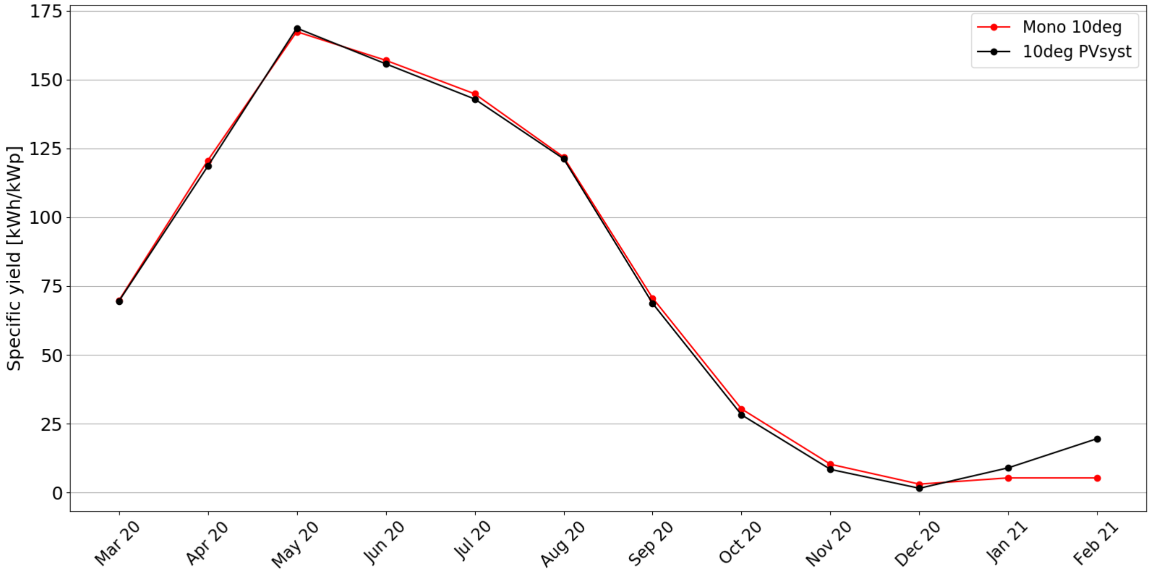


Figure 6.20: Specific yield for five mono c-Si modules compared to the PVsyst simulated specific yield. From March 2020 to February 2021.

Figure 6.21 shows a comparison of simulated module temperature for the 10° mono c-Si system, and measured module temperature from the back of the PV modules. The module temperatures are the average values for each month. The simulated PVsyst module temperature shows higher values for almost all months. However, both graphs follow each other closely and show the same trends. The drop in July 2020 may be due to lower ambient temperatures for this month, thus resulting in a lower module temperature. For January and February, the deviance between simulated and measured values increases. The constant loss factor (U_c) and wind loss factor (U_v) was set to 25 W/m^2K and 1.2 $W/m^2K\ m/s$, respectively, in the thermal loss factor calculations in PVsyst. These values are based on previous research done by the author in [8]. The 10° tilt modules at UiA are mounter somewhere between close-roof mount and open-rack mount. An algorithm could be written to calculate accurate values for U_c and U_v. This may close the gap between measured module temperature and simulated modules temperature in figure 6.21.

PR calculated in PVsyst is based on the AC energy output (PR_{AC}), while the PR calculate in the PV performance for this report is based on the DC energy output (PR_{DC}). Therefore, PR values are deviant between the simulated system and the calculated. The PVsyst simulated PR_{AC} has a yearly value of 0.837. It decreases in the winter months from October 2020 to December 2020, simultaneously as the loss due to irradiance and the incident angle loss increase. In January and

February 2021, the PVsyst simulated PR_{AC} increase above the calculated PR_{DC}. This is due to the constant soiling value in the PVsyst simulation being set to zero, while the actual soiling value in January and February was higher due to large periods of snow. To further evaluated the PR, a PR_{AC} has been calculated by accounting for the conversion losses from DC to AC, see chapter 6.5.2. The calculation was simplified by assuming that the annual conversion loss given by the PVsyst simulation was constant for all months. This calculated PR_{AC} for the real PV modules is given as the blue graph in figure 6.21.

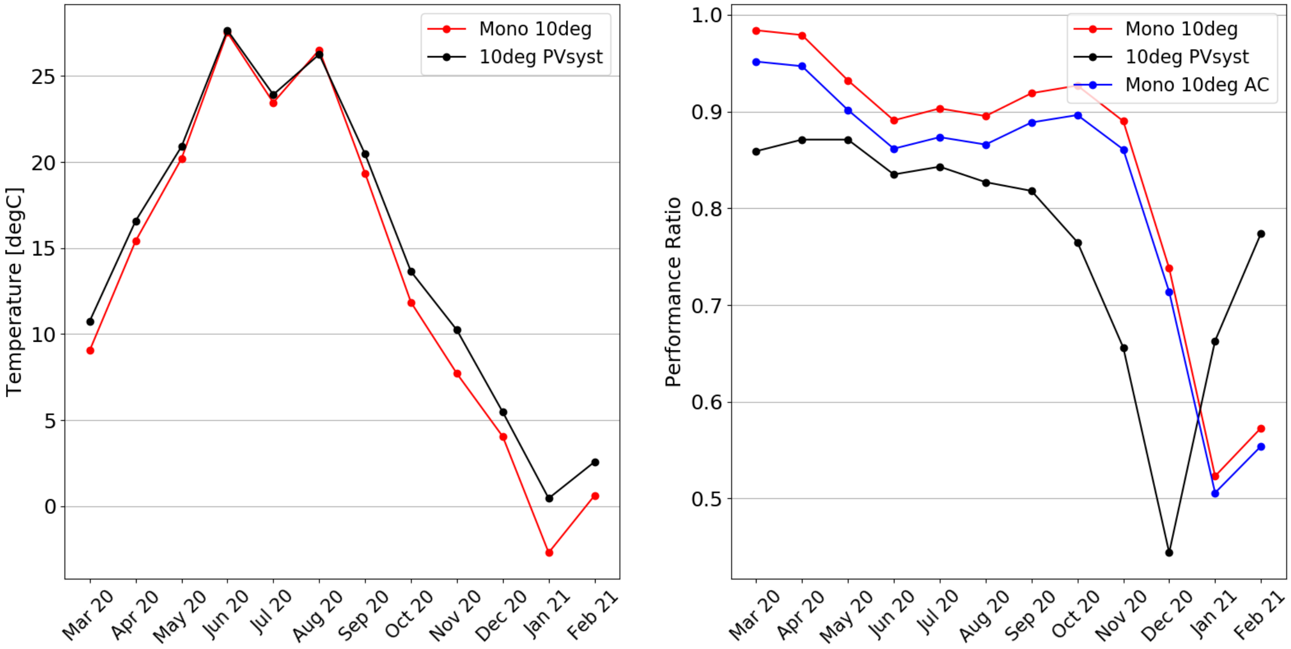


Figure 6.21: (Left) Comparison of measured module temp for the five mono c-Si modules and the simulated PVsyst module temperature. (Right) Comparison of PR for the five mono c-Si modules (PR_{DC}), PVsyst PR (PR_{AC}) and the calculated PR_{AC} for the actual system.

6.5.2 Losses

When performing a simulation in PVsyst, a loss diagram is calculated for the system, see figure 6.23. The upper part of the diagram is the optical losses, the middle part is the array losses, and the lower part is the system losses. Figure 6.23 show the yearly report losses for the whole year, however, monthly losses are also available.

The Global horizontal irradiance for the simulation was 1111 kWh/m^2. The largest optical loss is -5.2 %, this is due to incident angle. The smallest optical loss is due to near shading and is -0.3 %.

The largest loss in the array losses is due to irradiance level, with a value of -6.6 %. The efficiency of the PV system decreases with the decrease in irradiance, this results in the losses due to low irradiance levels, with respect to the STC irradiance at 1000 W/m^2. The irradiance loss can be explained with the one-diode model, see chapter 2.3, and the efficiency at low irradiances depends on two parameters: shunt resistance (R_{sh}) and series resistance (R_s). The R_{sh} has an exponential behavior when the irradiance decreases the R_{sh} increases exponentially. The R_s corresponds the square of the current (I^2) and increases with power. If the R_s is high, the losses are higher at STC. Figure 6.22 shows a graph of the efficiency in correlation with the irradiance for the mono c-Si module type used in the simulation and installed at UiA. Earlier research done by Imenes et al. (2017) [75], investigated the irradiance and temperature distribution in several locations in Norway. The result show that the irradiance for Kristiansand (south of Norway) is represented by relatively high frequencies of irradiance in the lower regions (200 - 800 W/m^2). The diffuse irradiance has a peak fraction at 100 - 200 W/m^2. This result correlates well with the high irradiance loss for this PVsyst simulation. The total GHI reported in [75] is 1054 kWh/m^2 for Kristiansand, which correlates quite well with the measured GHI for UiA at 1111 kWh/m^2, taking into account the natural inter-annual variations.

The last part of the loss diagram, figure 6.23 is the system losses. The annual loss for DC to AC conversion is -3.4 % in total. -3.3% for which is inverter loss during operation, and -0.1 % is inverter loss due to power threshold.

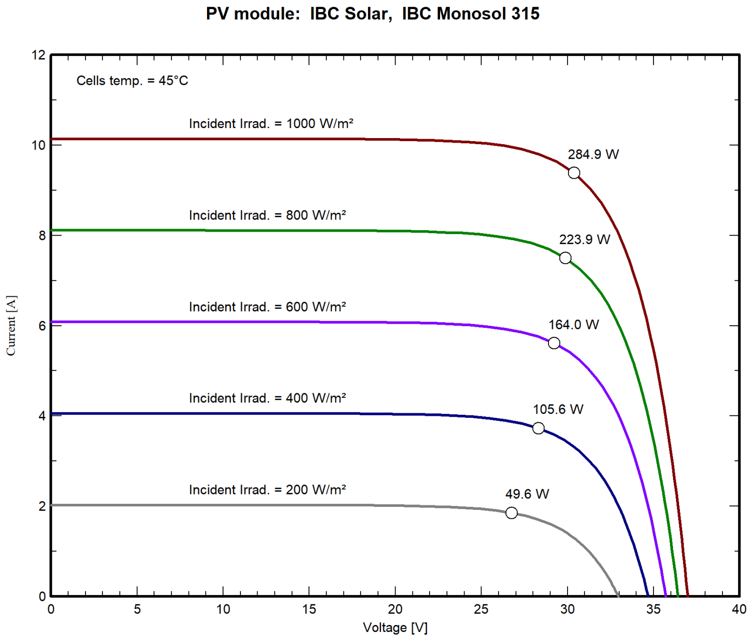


Figure 6.22: Efficiency vs. irradiance for the mono c-Si module used in the simulation and at UiA.

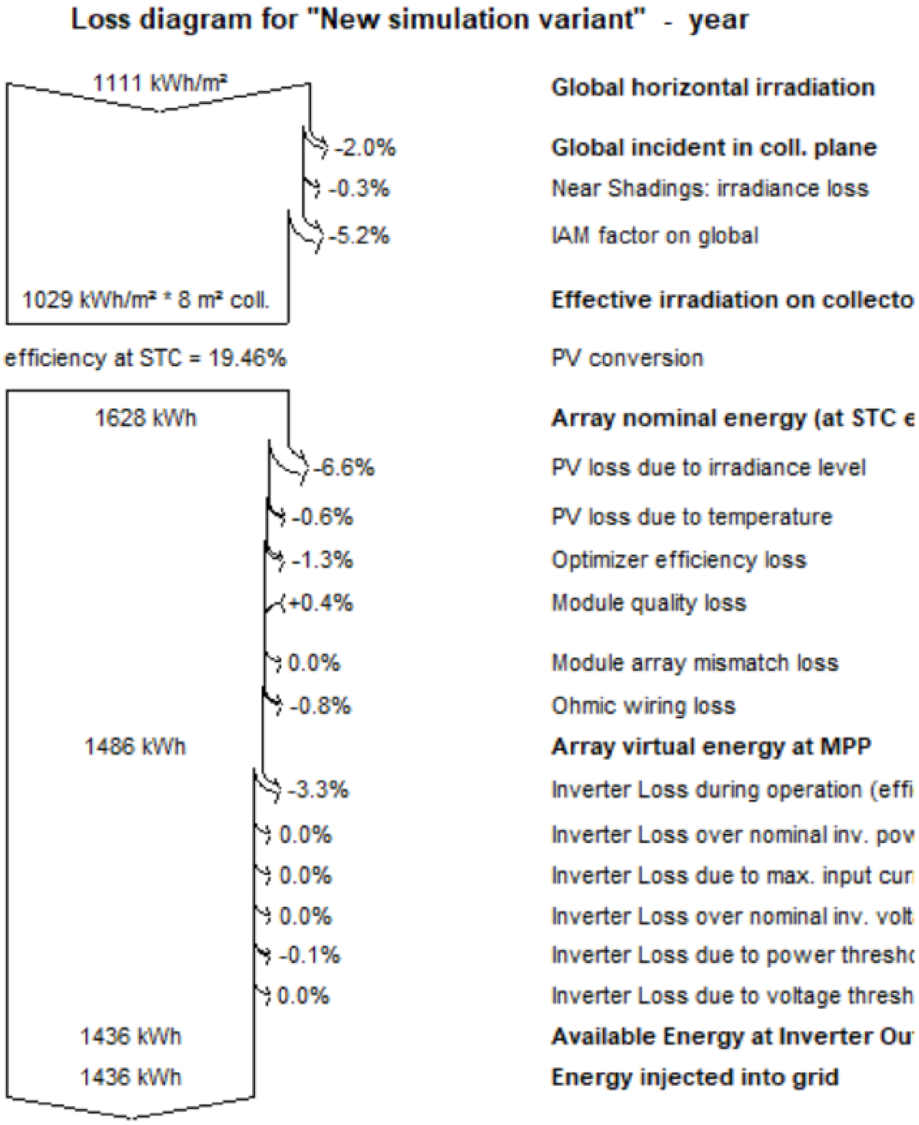

Loss diagram for "New simulation variant" - year
1111 kWh/m²
Global horizontal irradiation
-2.0%
Global incident in coll. plane
-0.3%
Near Shadings: irradiance loss
-5.2%
IAM factor on global
1029 kWh/m² * 8 m² coll.
Effective irradiation on collecto
efficiency at STC = 19.46%
PV conversion
1628 kWh
Array nominal energy (at STC e
-6.6%
PV loss due to irradiance level
-0.6%
PV loss due to temperature
-1.3%
Optimizer efficiency loss
+0.4%
Module quality loss
0.0%
Module array mismatch loss
-0.8%
Ohmic wiring loss
1486 kWh
Array virtual energy at MPP
-3.3%
Inverter Loss during operation (effi
0.0%
Inverter Loss over nominal inv. pov
0.0%
Inverter Loss due to max. input cur
0.0%
Inverter Loss over nominal inv. volt
-0.1%
Inverter Loss due to power threshc
0.0%
Inverter Loss due to voltage thresh
1436 kWh
Available Energy at Inverter Ou
1436 kWh
Energy injected into grid

Figure 6.23: Loss diagram calculated by the PVsyst simulation.

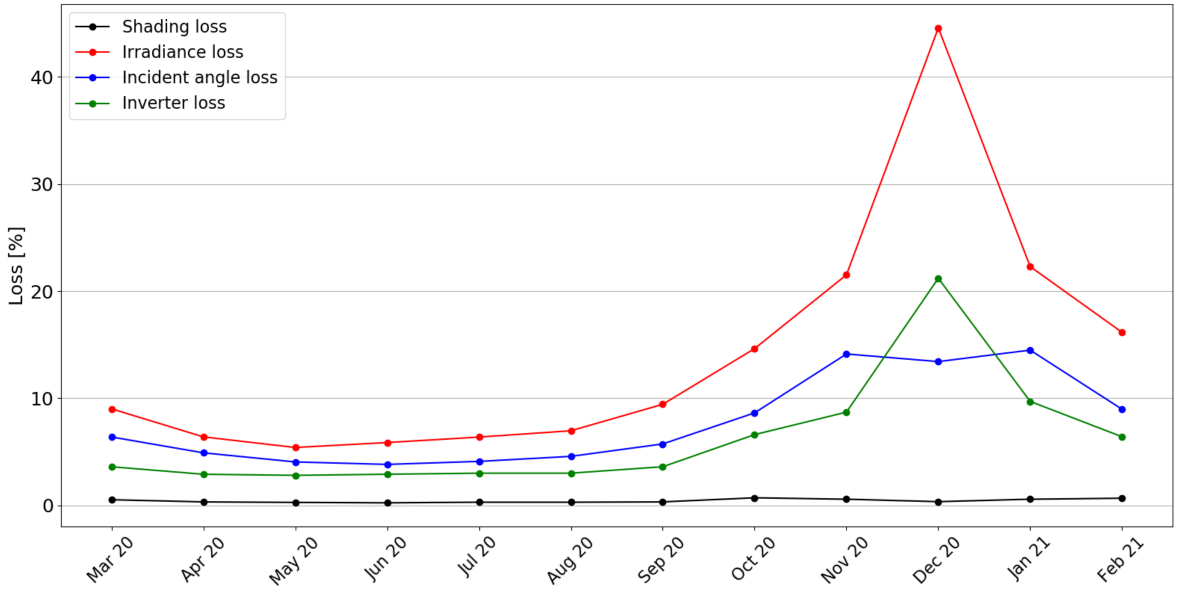


Figure 6.24: Monthly losses in % for shading loss (black), irradiance loss (red) and incident angle loss (blue).

Figure 6.24 shows an overview of monthly loss values for shading loss, irradiance loss and incident angle loss for the PVsyst simulation. The losses are given in percentage [%] . The shading losses are close to zero throughout the whole year. The incident angle losses dip slightly in the summer and increase in the winter. This loss corresponds to the amount of irradiance effectively reaching the PV cell. With the lower sun angle in the winter, a larger portion of the irradiance is reflected in the top glass of the module, and less irradiance will reach the cell. The irradiance loss is close to the incident angle loss from March to September 2020 but increases rapidly during the winter months. In December 2020 it reaches a value of 44.58 %. This corresponds with the low GHI measured in December. Lastly, the inverter loss during operation also experiences an increase in December 2020. This loss is given as the efficiency of the inverter in PVsyst, however, it has been recalculated to the inverter loss to be able to compare it to the other losses in figure 6.24.

6.5.3 Uncertainty

Figure 6.20 shows a comparison of the AC specific yield simulated in PVsyst and the DC specific yield calculated from measured data at UiA. These two graphs show near identical values. From this result it was expected that simulated PR_{AC} and calculated PR_{DC} should have a good correlation, however, these graphs deviate a lot. Due to the close comparison of specific yield the difference has to be in the calculation of reference yield, i.e., the irradiation measurements. The DC measurements uses E_{POA} measured by a reference cell close to the ground, while the irradiance data inserted in PVsyst was GHI and DHI measurement, which is then converted to POA data in PVsyst. This conversion is affected by an uncertainty. There will also be a difference in soiling for the two irradiance instruments. The reference cell is mounted close to the ground, and is therefore affected by dust, rain-water, frost, and snow covering in the winter. The GHI and DHI is measured with a pyranometer mounted 1.5 m above the ground, and it is therefore not as exposed to soiling. The pyranometer also has a CVF4 ventilator to prevent soiling, frost and snow from covering it. The soiling on the reference cell makes the reference yield for POA lower than the reference yield for GHI. This will again affect the performance ratio calculation, resulting in a higher PR_{DC} than PR_{AC}.

I addition, there is not a 100 % data availability for the five mono c-Si modules simulated. The DC output values should consequently be a bit higher with full data availability, resulting in a slightly higher DC specific yield than AC specific yield. This is also affecting the PR calculations, making the PR_{DC} higher than the PR_{AC}. Finally, there is also uncertainty in all measurement sensors that can affect the degree of matching between simulated and measured results.

7 Conclusion

The outdoor performance of eight mono- and poly-crystalline PV modules in different installation angles at the University of Agder, Norway, has been investigated. The problem statement for this thesis was divided into three parts. The first part was to analyse the monitoring data and calculate specific yield and performance ratio for the PV technologies in the different installation configurations. The second part was to investigate the effect temperature have on the performance of PV technologies at different tilt angles and orientations. The last part was to use a software tool to predict performance of the PV modules and compare the results with measured data. From the analysis of the monitoring data for the different tilt angle and orientations for a time period from March 2020 to Mach 2021.

Four of the investigated modules were grid-connected and mounted in an East-West configuration at a 10° tilt angle. The analysis was based on power optimizer data from each individual module, and the four modules show similar results for all the analysis performed. The annual DC specific yield was in the range 920 - 984 kWh/kWp, with a slightly better performance for poly c-Si compared to mono c-Si modules. The annual average PR_{DC} for mono c-Si was 0.88 (East) and 0.90 (West), and for poly c-Si was 0.90 (East) and 0.88 (West), with high data availability for most of the year except during winter, indicating a well-functioning system.

The other four modules investigated were part of a research system monitoring I-V curves of individual PV modules mounted on a 45° free rack, and two vertically installed BIPV modules, all facing South. The two modules mounted on the 45° tilt rack and the BIPV modules all had lower data availability due to missing data points. The specific yield before estimating data for missing data points was 988 kWh/kWp (mono c-Si 45°), 1050 kWh/kWp (poly c-Si 45°), 762 kWh/kWp (BIPV normal), and 760 kWh/kWp (BIPV grey). When estimating output power for the missing data point, all modules experience an increase in specific yield performance. The specific yield after estimation was 1171 kWh/kWp (mono c-Si 45°), 1244 kWh/kWp poly c-Si 45°), 926 kWh/kWp (BIPV normal), and 925 kWh/kWp (BIPV grey). For the 45° tilt and BIPV modules the PR_{DC} before temperature correction was 0.86 (mono c-Si 45°), 0.91 (poly c-Si 45°), 0.90 (BIPV normal) and 0.87 (BIPV grey). When correcting the PR for temperature all PR_{DCcorr} values decreased to 0.83 (mono c-Si 45°), 0.87 (poly c-Si 45°), 0.83 (BIPV normal) and 0.98 (BIPV grey). The specific yield for the 45° tilt and BIPV modules varies a lot throughout the period and is highly affected by the low data availability.

For all PV modules the mean performance ratio decreased with a value between 0.02 - 0.05 when correcting it for temperature. The module with the highest PR_{DCcorr} was the poly c-Si at 45°. Overall, all modules obtained more even performance during the time period when correcting the PR for temperature. The module temperature for all modules follow each other closely. Typically the 45° modules obtain a slightly higher temperature throughout the year. For the 10° tilt all modules obtain approximately the same module temperature, not affected by orientation (East/West) or module type.

A simulation of five mono c-Si modules was conducted in PVsyst to compare results and investigate factors affecting the performance of the system. The simulated PVsyst result are similar to the recorded measurements from the system at UiA. The total simulated system production (AC) was 1436 kWh/year, and the specific production was 912 kWh/kWp/year. For the actual system, the recorded total system production (DC) was 1451 kWh/year and the specific production 910 kWh/kWp/year. The largest loss in the simulated system was the array losses resulting in -9 % loss in total. The array loss consists of PV loss due to irradiance level, temperature, optimizer efficiency, module quality loss, and ohmic wiring loss. The annual average PR from the PVsyst simulation was 0.84 (AC), compared to the measured PR (DC) of 0.85.

Comparing the two different PV technologies mounted at different tilt angles and orientations, there is not detected a substantial difference in performance between the East- and West-facing modules at 10° tilt. When comparing the mono c-Si module at a 10° tilt East/West versus 45° tilt South, the specific yield result in a difference of 251 kWh/kWp (45° vs. 10° E), and 209 kWh/kWp (45° vs. 10° W), using specific yield with estimated missing data. For the poly c-Si modules the difference in 10° tilt, 45° tilt and 90° BIPV is between 260 kWh/kWp - 484 kWh/kWp. In performance ratio all mono c-Si modules obtain a PR_{DC} between 0.86-0.88, PR_{DCcorr} between 0.83-0.85, and PR_{DCest} between 0.88-0.92. For the poly c-Si modules the value of PR_{DC} is between 0.87-0.91, PR_{DCcorr} 0.83-0.87 and PR_{DCest} between 0.88-0.95.

The overall best PR is the PR_{DCest} for the 45° poly c-Si module at 0.95. However, the PR for the higher tilted modules are on average slightly better than for the 10° tilt modules. The best indication of module performance in the various installation angles are when estimating output power for missing data. From these results the PR_{DCest} results for the 10° tilt modules are in the range 0.88 - 0.90 (mono c-Si E/W) and 0.88-0.92 (poly c-Si W/E), while for the 45° tilt modules the values are 0.92 (mono c-Si) and 0.95 (poly c-Si), and for both the normal and grey BIPV modules at 90° tilt the value is 0.94.

Lastly, the analysis in this thesis is conducted with data measured for a time period of 13 months. More data, collected over a longer time period, is recommended to give a more reliable result that also takes into account natural weather variations. Also, the results cannot be generalized, as the performance of different installations will differ with other locations. However, this work gives a contribution to the documentation of outdoor performance from PV installations in Norway, and may be used as a reference in comparison with other similar systems based on crystalline silicon module technology.

7.1 Further Work

The recommendation for further work is to investigate the PV performance for additional PV modules, and modules in other parts of the array. The analysis should also be repeated when a longer time period of data is available.

The output power for missing data point should be estimated for the 10° tilt modules, to compare all tilt angles and orientations at 100 % data availability. In addition, the unknown factor resulting in unexpectedly low results for BIPV modules in June and July should be investigated.

Due to limited time only five mono c-Si modules at 10° tilt was simulated in PVsyst. To further evaluate the different tilt angles and orientations it would be beneficial to simulate PV modules from the poly c-Si 10° tilt system, 45° tilt rack and the BIPV modules.

The PVsyst simulation should be used to simulate other parts of the PV system, and other installation angles and orientations. Both the ones used in this thesis and potential others.

Lastly, the ohmic loss in the cables should be included in the analysis.

References

[1] Statkraft. “Globale energitrender og norske muligheter: Statkrafts Lavutslippsscenario”. In: (2019).

[2] *Energy – United Nations Sustainable Development.* https://www.un.org/sustainabledevelopment/energy/. (Accessed on 12/14/2020).

[3] *Solkraft i Norge: Økte med 29 prosent på ett år – Solenergiklyngen.* https://solenergiklyngen.no/2019/03/06/solkraft-i-norge-okte-med-29-prosent-pa-ett-ar/. (Accessed on 04/18/2021).

[4] *Solenergi i Norge - Stort potensial for solenergi i Norge!* https://www.otovo.no/blog/solenergi/solenergi-og-solcellepaneler-norge/. (Accessed on 05/25/2021).

[5] Glunz et al. *Comprehensive Renewable Energy.* Elsevier, 2012, pp. 353–387.

[6] *Research Centre for Sustainable Solar Cell Technology - Prosjektbanken.* https://prosjektbanken.forskningsradet.no/project/FORISS/257639?Kilde=FORISS&distribution=Ar&chart=bar&calcType=projects&Sprak=no&sortBy=score&sortOrder=desc&resultCount=30&offset=0&Fritekst=SUSOLTECH. (Accessed on 05/26/2021).

[7] Solutech. *What is SUSOLTECH? | SuSolTec.* https://susoltech.no/about-susoltech/. (Accessed on 05/26/2021).

[8] Sigrid Langholm Larsen. “Investigation of temperature influence on performance of different PV technologies under nordic conditions.” Final report ENE503 Energy Research Project, University of Agder. (internal report, not published). Dec 2020.

[9] Bert Herteleer. “Outdoor thermal and electrical characterisation of photovoltaic modules and systems”. In: *PhD thesis* (Feb. 2016).

[10] *Best Research-Cell Efficiency Chart | Photovoltaic Research | NREL.* https://www.nrel.gov/pv/cell-efficiency.html. (Accessed on 11/13/2020).

[11] Antonio Bayod-Rújula. *Solar hydrogen production, Chapter 8 - Solar photovoltaics (PV).* Academic press, 2019, pp. 237–295.

[12] Andrew Sendy. *Types of Solar Panels: Which One Is the Best Choice?* https://www.solarreviews.com/blog/pros-and-cons-of-monocrystalline-vs-polycrystalline-solar-panels. (Accessed on 02/15/2021). Jan. 2021.

[13] *Crystalline Silicon Photovoltaics Research | Department of Energy.* https://www.energy.gov/eere/solar/crystalline-silicon-photovoltaics-research. (Accessed on 02/15/2021).

[14] Konrad Mertens. *Photovoltaics: Fundamentals, Technology, and Practice.* Second Edition. Wiley & Sons Ltd, 2019, Munich.

[15] *Monocrystalline vs. Polycrystalline Solar Panels | EnergySage.* https://www.energysage.com/solar/101/monocrystalline-vs-polycrystalline-solar-panels/. (Accessed on 02/15/2021). June 2020.

[16] *Commercial Solar Cells | Silicon Solar.* https://www.siliconsolar.com/commercial-solar-cells/. (Accessed on 10/31/2020).

[17] Zhijian Liu et al. “A comprehensive study of feasibility and applicability of building integrated photovoltaic (BIPV) systems in regions with high solar irradiance”. In: *Journal of Cleaner Production* 307 (May 2021), pp. 1–12. DOI: 10.1016/j.jclepro.2021.127240.

[18] Erika Saretta et al. “A calculation method for the BIPV potential of Swiss façades at LOD2.5 inurban areas: A case from Ticino region”. In: *Solar Energy* 195 (Jan. 2020), pp. 150–165. DOI: 10.1016/j.solener.2019.11.062.

[19] Emrah Biyik et al. “A key review of building integrated photovoltaic (BIPV) systems”. In: *Engineering Science and Technology,an International Journal* 20.3 (May 2017), pp. 833–858. DOI: 10.1016/j.jestch.2017.01.009.

[20] *Solar Panel Construction — Clean Energy Reviews.* https://www.cleanenergyreviews.info/blog/solar-panel-components-construction. (Accessed on 11/12/2020).

[21] *Solar Panel Wiring Basics: An Intro to How to String Solar Panels.* https://blog.aurorasolar.com/solar-panel-wiring-basics-an-intro-to-how-to-string-solar-panels. (Accessed on 11/07/2020).

[22] *How to wire solar panels in series vs. parallel.* https://www.solarreviews.com/blog/do-you-wire-solar-panels-series-or-parallel. (Accessed on 11/07/2020).

[23] *What Are DC Power Optimizers?* https://www.solarreviews.com/blog/complete-guide-to-power-optimizers. (Accessed on 05/26/2021).

[24] PVeducation. *Fill Factor | PVEducation.* `https://www.pveducation.org/pvcdrom/solar-cell-operation/fill-factor`. (Accessed on 09/11/2020).

[25] *Diode Equation | PVEducation.* `https://www.pveducation.org/pvcdrom/pn-junctions/diode-equation`. (Accessed on 12/09/2020).

[26] Stuart R. Wenham et al. *Applied Photovoltaics.* Third Edition. Earthscan, 2011.

[27] *Shading | PVEducation.* `https://www.pveducation.org/pvcdrom/modules-and-arrays/shading`. (Accessed on 12/09/2020).

[28] *Bypass Diodes | PVEducation.* `https://www.pveducation.org/pvcdrom/modules-and-arrays/bypass-diodes`. (Accessed on 12/09/2020).

[29] *What are Standard Test Conditions (STC) | Silicon Solar.* `https://www.siliconsolar.com/what-are-standard-test-conditions-stc/`. (Accessed on 10/06/2020).

[30] George Makrides et al. "ANNUAL ENERGY YIELD OF 13 PHOTOVOLTAIC TECHNOLOGIES IN GERMANY AND IN CYPRUS". In: (2007).

[31] "Photovoltaic system performance: Part 1 Monitoring". In: (2017).

[32] SMA Solar Technology AG. "Performance ratio - Quality factor for the PV plant". In: *Technical report* ().

[33] *Project design > Results > Performance Ratio PR.* `https://www.pvsyst.com/help/performance_ratio.htm`. (Accessed on 05/27/2021).

[34] Mr. Rakesh Bohra. "Performance Analysis of 1MW SPV Plant; Temperature Corrected PR". In: *energetica india* (2014).

[35] J.A. Kratochvill D.L. King W.E. Boyson. "Photovoltaic Array Performance Model". In: *SANDIA REPORT* SAND2004-3535 (2005).

[36] Timothy Dierauf a et al. "Weather-Corrected Performance Ratio". In: *Technical Report NREL/TP-5200-57991* (2013), pp. 1–16.

[37] *Standard Solar Spectra | PVEducation.* `https://www.pveducation.org/pvcdrom/appendices/standard-solar-spectra`. (Accessed on 09/25/2020).

[38] M. Alonso-Abella et al. "Analysis of spectral effects on the energy yield of different PV (photovoltaic) technologies: The case of four specific sites". In: *Energy* 67 (2014), pp. 435–443.

[39] *Spectral Response | PVEducation*. https://www.pveducation.org/pvcdrom/solar-cell-operation/spectral-response. (Accessed on 04/15/2021).

[40] Jesus Polo et al. "Worldwide analysis of spectral factors for seven photovoltaictechnologies". In: *Solar Energy* 142 (Jan. 2017), pp. 194–203. DOI: 10.1016/j.solener.2016.12.024.

[41] Jesus Polo et al. "Worldwide analysis of spectral factors for seven photovoltaictechnologie". In: *Solar Energy* 142 (2017), pp. 194–203. DOI: 10.1016/j.solener.2016.12.024.

[42] *Solar Panel Angle: how to calculate solar panel tilt angle?* https://sinovoltaics.com/learning-center/system-design/solar-panel-angle-tilt-calculation/. (Accessed on 02/23/2021).

[43] H. Y. Cheng et al. "Estimating Solar Irradiance on Tilted Surface with Arbitrary Orientations and Tilt Angles". In: *Energies* 12.8 (Apr. 2019), pp. 1–14. DOI: 10.3390/en12081427.

[44] Norwegian electrotechnical publication. "Photovoltaic system performance, Part 1: Monitoring". In: *NEC IEC 61724-1:2017* (2017).

[45] B. Marison et al. "Performance parameters for grid-connected PV systems". In: *Conference Record of the IEEE Photovoltaic Specialists Conference* (Feb. 2005), pp. 1601–1606. DOI: 10.1109/PVSC.2005.1488451.

[46] *System Loss Diagram – Aurora Solar Help Center*. https://help.aurorasolar.com/hc/en-us/articles/235994088-System-Loss-Diagram. (Accessed on 05/18/2021).

[47] W. van Sark et al. "REVIEW OF PV PERFORMANCE RATIO DEVELOPMENT". In: *Fraunhofer ISE Conference paper* (2012). ISSN: 978-1-622-76092-3.

[48] Mahmoud Dhimish. "Performance Ratio and Degradation Rate Analysis of 10-Year Field Exposed Residential Photovoltaic Installations in the UK and Ireland". In: *Clean Technologies* 2 (May 2020), pp. 170–183. DOI: 10.3390/cleantechnol2020012.

[49] *Energy use by sector - Energifakta Norge*. https://energifaktanorge.no/en/norsk-energibruk/energibruken-i-ulike-sektorer/. (Accessed on 04/05/2021).

[50] Multiconsult. *Fornybar-energi.pdf*. https://multiconsultgroup.com/assets/Fornybar-energi.pdf. (Accessed on 05/16/2021).

[51] Solenergiklyngen. *Fakta – Solenergiklyngen*. https://solenergiklyngen.no/fakta/. (Accessed on 05/16/2021).

[52] M. S. Adaramola et al. "Preliminary assessment of a small-scale rooftop PV-grid tied in Norwegian climatic conditions". In: *Energy Conversion and Management* 90 (Dec. 2014), pp. 458–465.

[53] Aksel Pettersen. "THe Performance of Household PV Systems in Southeastern Norway". In: *Master Thesis: Norwegian University of LiFE Sience* (2019), pp. 1–104.

[54] Anne Gerd Imenes. "Performance of BIPV and BAPV Installations in Norway". In: *2016 IEEE 43rd Photovoltaic Specialists Conference (PVSC)* (2016), pp. 3147–3152. DOI: 10.1109/PVSC.2016.7750246.

[55] A. G. Imenes et al. "Performance of grid-connected PV system in Southern Norway". In: *2015 IEEE 42nd Photovoltaic Specialist Conference (PVSC)*. 2015, pp. 1–6. DOI: 10.1109/PVSC.2015.7355823.

[56] Gabi Friesen et al. "Photovoltaic Module Energy Yield Measurements: Existing Approaches and Best Practice". In: *IEA PVPS Task 13, Subtask 3* (2018). ISSN: 978-3-906042-52-7.

[57] T. Nordmann et al. "Analysis of Long-Term Performance of PV Systems". In: *IEA PVPS Task 13, Subtask 1* (2014). ISSN: 978-3-906042-21-3.

[58] T. Takashima T. Ishii and K. Otani. "Long-term performance degradation of various kinds of photovoltaic modules under moderate climatic conditions". In: *Progress in photovoltaics: Research and Applications* 19.2 (Feb. 2011), pp. 170–179. DOI: doi.org/10.1002/pip.1005.

[59] Daniel Tudo Cotfas et al. "Study of temperature coefficients for parameters of photovoltaic cells". In: *International Journal of Photoenergy* 2018 (2018), pp. 1–12. DOI: 10.1155/2018/5945602.

[60] Matthew Peilliman et al. "Transient Weighted Moving-Average Model of Photovoltaic Module Back-Surface temperature". In: *IEEE Journal of Photovoltaics* 10.4 (June 2020), pp. 1053–1060.

[61] Swapnil Dubey et al. "Temperature Dependent Photovoltaic (PV) Efficiency and Its Effect on PV production in the World - A Review". In: *Energy Procedia* 33 (2013), pp. 311–321. DOI: 10.1016/j.egypro.2013.05.072.

[62] M.A. Green O. Duåre R. Vaillon. "Physics of the temperature coefficients of solar cells". In: *Solar Energy Materials and Solar Cells* 140 (Jan. 2015), pp. 92–100. DOI: 10.1016/j.solmat.2015.03.025.

[63] Charly Berthod et al. "On the variability of the temperature coefficient of mc-Si solar cells with irradiance". In: *Energy Procedia* 92 (2019), pp. 2–9. DOI: 10.1016/j.egypro.2016.07.002.

[64] P. K. Dash and N. C. Gupta. "Effect of Temperature on Power Output from Different Commercially available Photovoltaic Modules". In: *Journal of Engineering Research and Applications* 5.1 (Jan. 2015), pp. 1–4.

[65] Getu Hailu and Alan S. Fung. "Optimum tilt angle and orientation of photovoltaic thermal system for application in greater Toronto area, Canada". In: *Sustainability* 11.22 (Nov. 2019), pp. 1–21.

[66] S. Abbasoglu A. A. Babatunde and M. Senol. "Analysis of the impact of dust, tilt angle and orientation on performance of PV plants". In: *Renewable and Sustainable Energy Reviews* 90 (July 2018), pp. 1017–1026. DOI: 10.1016/j.rser.2018.03.102.

[67] Mondol et al. "The impact of array inclination and orientation onthe performance of a grid-connected photovoltaicsystem". In: *Renewable energy* 32 (2007), pp. 118–140.

[68] Joannes I. Laveyne et al. "Impact of Solar Panel Orientation on the Integration of Solar Energy in Low-Voltage Distribution Grids". In: *International Journal of Photoenergy* 2020 (Feb. 2020), pp. 1–13. DOI: 10.1155/2020/2412780.

[69] Ramez Abdallah et al. "Estimating the Optimum Tilt Angles for South-Facing Surfaces in Palestine". In: *Energies* 13 (Feb. 2020), pp. 1–29. DOI: 10.3390/en13030623.

[70] Riyad Mubarak et al. "Why PV Modules Should Preferably No Longer Be Oriented to the South in the Near Future". In: *Energies* 12 (Nov. 2019), pp. 1–16.

[71] Christian Hals Frivold. "Photovoltaic facade systems in Norway: An assessment of energy performance, buildingintegration, and costs". In: *Master Thesis at University of Agder* (June 2018), pp. 1–120.

[72] Johannes Eisenlohr. *Enabling Framework for the Development of BIPV - IEA-PVPS*. https://iea-pvps.org/research-tasks/enabling-framework-for-the-development-of-bipv/. (Accessed on 05/26/2021).

[73] *PVsyst 7 Help*. https://www.pvsyst.com/help/. (Accessed on 04/24/2021).

[74] *Project design > Array and system losses > Array Thermal losses*. https://www.pvsyst.com/help/thermal_loss.htm. (Accessed on 12/13/2020).

[75] A. G Imenes et al. "Irradiance and temperature distribution at high latitudes: Design implications for photovoltaic systems". In: *2017 IEEE 44th Photovoltaic Specialist Conference (PVSC)* (June 2017), pp. 0619–0625. DOI: 10.1109/PVSC.2017.8366376.

Appendices

A Appendix

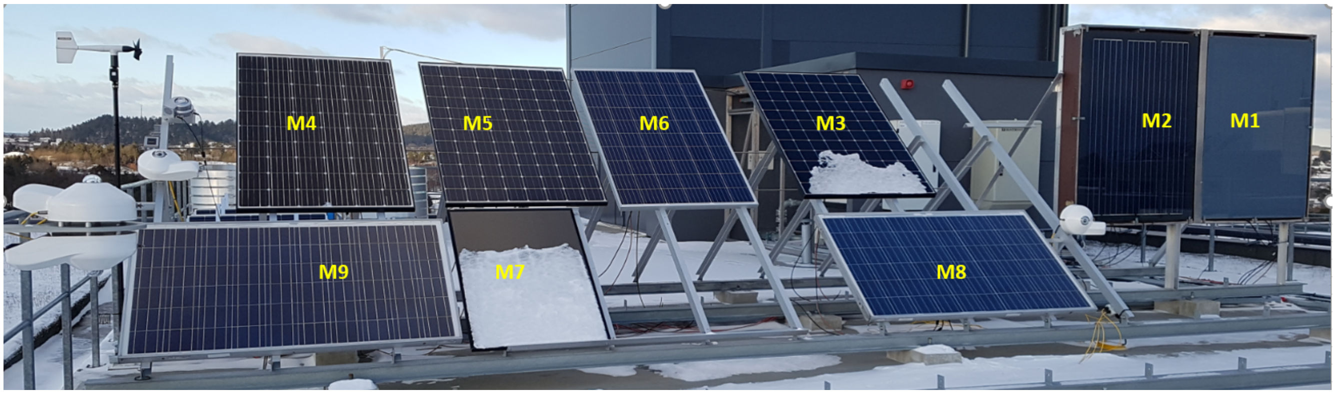


Figure A.1: Picture of all PV modules mounted on the 45° rack and the BIPV modules. The ones used in this report are M1, M2, M4 and M6.

Figure A.2: Picture of the 45° rack covered in snow in January 2021. The BIPV modules are not effected by the snow.

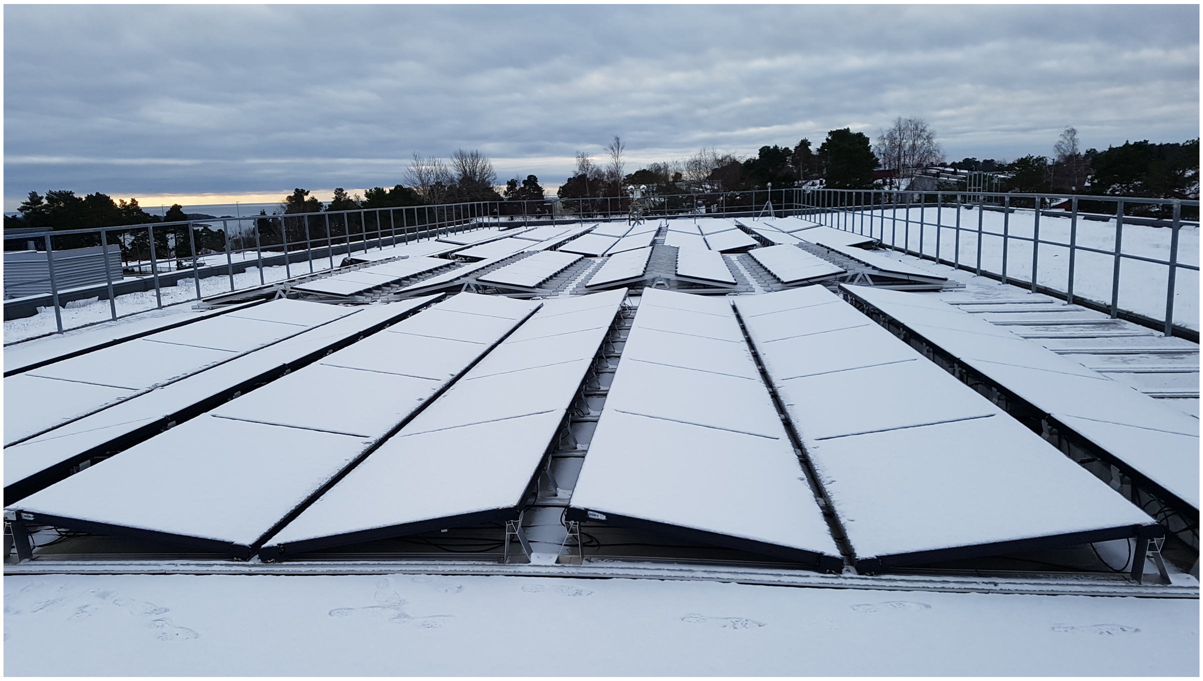


Figure A.3: Light cover of snow on the 10° modules at UiA. Picture taken in January 2021

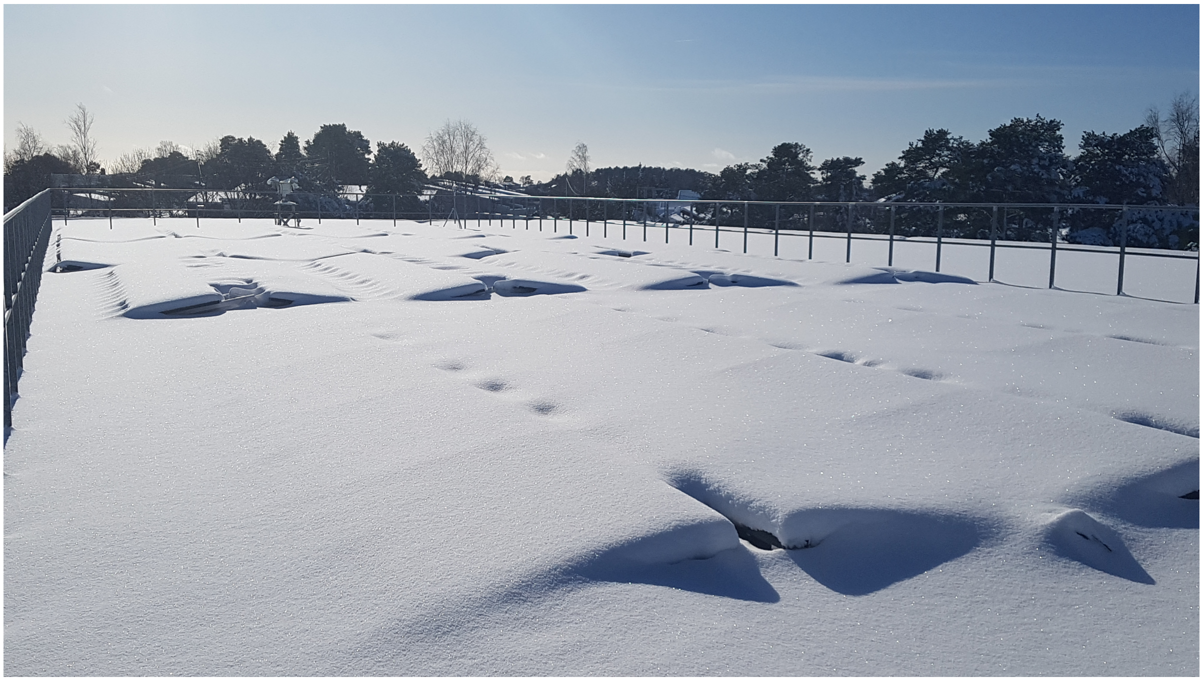


Figure A.4: Heavy cover of snow on the 10° modules in February 2020.

B Appendix

Version 7.1.8

PVsyst - Simulation report

Grid-Connected System

Project: UiA J5 building

Variant: New simulation variant

Sheds on ground

System power: 1575 Wp

Grimstad - Norway

Author
Sigrid Larsen (Norway)

Project summary

Geographical Site	**Situation**		**Project settings**	
Grimstad	Latitude	58.33 °N	Albedo	0.30
Norway	Longitude	8.58 °E		
	Altitude	12 m		
	Time zone	UTC+1		

Meteo data
Grimstad
Custom file - Imported

System summary

Grid-Connected System | **Sheds on ground**

PV Field Orientation		**Near Shadings**	**User's needs**
Fixed plane		Linear shadings	Unlimited load (grid)
Tilt/Azimuth	10 / -97 °		

System information

PV Array		**Inverters**	
Nb. of modules	5 units	Nb. of units	0.5 Unit
Pnom total	1575 Wp	Pnom total	1500 W
		Pnom ratio	1.050

Results summary

Produced Energy	1436 kWh/year	Specific production	912 kWh/kWp/year	Perf. Ratio PR	83.74 %

Table of contents

General parameters

Grid-Connected System

PV Field Orientation

Orientation

Fixed plane	
Tilt/Azimuth	10 / -97 °

Sheds on ground

Sheds configuration

Nb. of sheds	2 units
Identical arrays	
Sizes	
Sheds spacing	2.26 m
Collector width	1.01 m
Ground Cov. Ratio (GCR)	44.7 %
Top inactive band	0.01 m
Bottom inactive band	0.01 m
Shading limit angle	
Limit profile angle	8.1 °

Models used

Transposition	Perez
Diffuse	Imported
Circumsolar	separate

Horizon

Free Horizon

Near Shadings

Linear shadings

User's needs

Unlimited load (grid)

PV Array Characteristics

PV module

Manufacturer	Generic
Model	IBC Monosol 315
(Custom parameters definition)	
Unit Nom. Power	315 Wp
Number of PV modules	5 units
Nominal (STC)	1575 Wp
Optimizer Array	1 String x 5 In series
At operating cond. (50°C)	
Pmpp	1384 Wp
U mpp	148 V
I mpp	9.4 A

Tigo Optimizer

Model	TS4-O
Unit Nom. Power	375 W
Modules	1 String x 1 in series

Total PV power

Nominal (STC)	2 kWp
Total	5 modules
Module area	8.1 m²

Inverter

Manufacturer	Generic
Model	X1-Boost-3.0kW with 2 MPPT
(Custom parameters definition)	
Unit Nom. Power	3.00 kWac
Number of inverters	1 * MPPT 50% 0.5 units
Total power	1.5 kWac
Operating voltage	125-580 V
Pnom ratio (DC:AC)	1.05

Total inverter power

Total power	1.5 kWac
Nb. of inverters	1 Unit
	0.5 unused
Pnom ratio	1.05

Array losses

Thermal Loss factor

Module temperature according to irradiance

Uc (const)	25.0 W/m²K
Uv (wind)	1.2 W/m²K/m/s

Module mismatch losses

Loss Fraction	0.0 % at MPP

DC wiring losses

Global array res.	271 mΩ
Loss Fraction	1.5 % at STC

IAM loss factor

ASHRAE Param: IAM = 1 - bo(1/cosi -1)

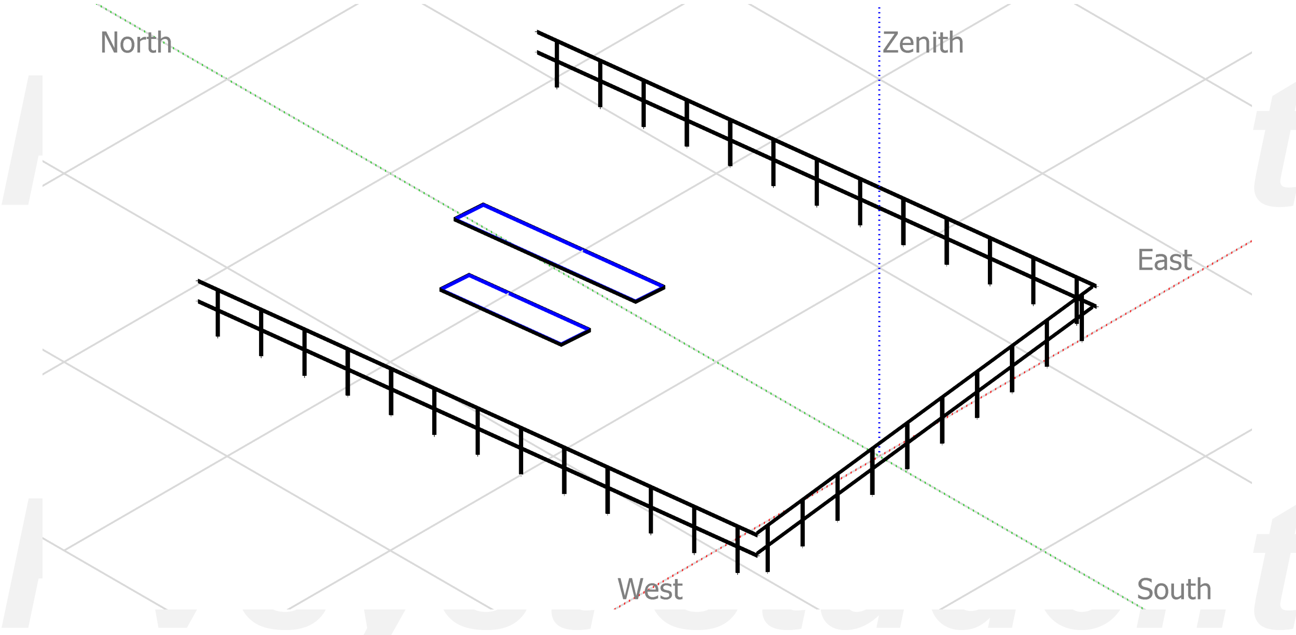
bo Param.	0.05

Module Quality Loss

Loss Fraction	-0.4 %

Near shadings parameter

Perspective of the PV-field and surrounding shading scene



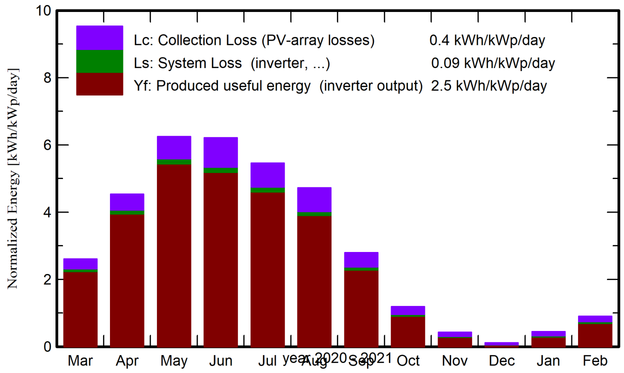
Iso-shadings diagram

Main results

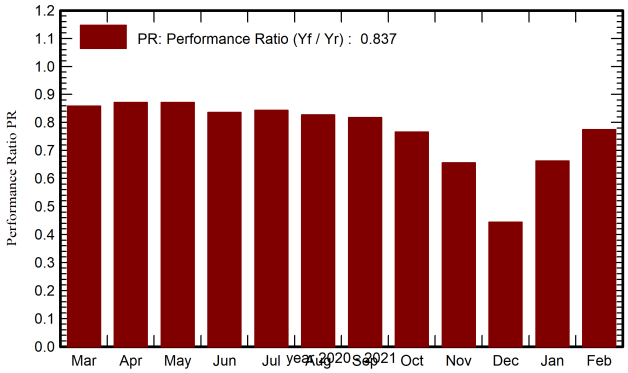
System Production

Produced Energy	1436 kWh/year	Specific production	912 kWh/kWp/year
		Performance Ratio PR	83.74 %

Normalized productions (per installed kWp)



Performance Ratio PR



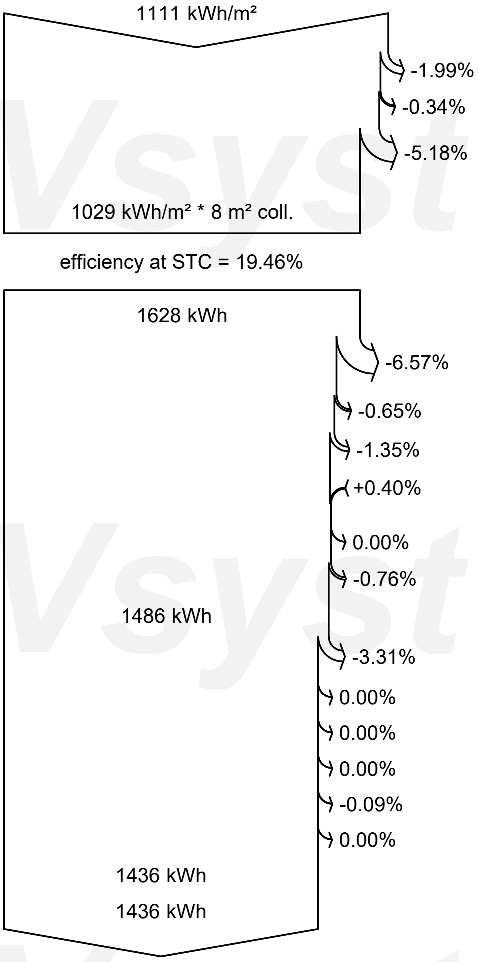
Balances and main results

	GlobHor	DiffHor	T_Amb	GlobInc	GlobEff	EArray	E_Grid	PR
	kWh/m²	kWh/m²	°C	kWh/m²	kWh/m²	kWh	kWh	ratio
Mar. 20	80.7	35.17	2.59	80.7	75.1	113.2	109.1	0.859
Apr. 20	139.1	39.93	5.11	136.1	128.8	192.3	186.7	0.871
May 20	197.1	66.27	8.13	193.7	185.2	273.3	265.7	0.871
June 20	189.1	70.86	13.85	186.3	178.6	252.3	245.1	0.835
July 20	171.2	70.35	11.81	169.3	161.8	231.8	224.9	0.843
Aug. 20	150.5	54.51	11.86	146.4	139.1	196.7	190.7	0.827
Sep. 20	87.0	38.25	8.17	83.9	78.6	112.1	108.0	0.818
Oct. 20	37.0	19.01	4.71	37.0	33.5	47.2	44.6	0.765
Nov. 20	13.4	6.50	2.38	12.8	10.8	14.4	13.2	0.656
Dec. 20	4.1	2.75	1.35	3.4	2.9	3.1	2.4	0.444
Jan. 21	14.8	7.39	-0.40	13.8	11.5	15.9	14.4	0.663
Feb. 21	26.5	14.25	-0.02	25.3	22.7	33.0	30.9	0.774
Year	1110.6	425.25	5.82	1088.5	1028.7	1485.3	1435.6	0.837

Legends

GlobHor	Global horizontal irradiation	EArray	Effective energy at the output of the array
DiffHor	Horizontal diffuse irradiation	E_Grid	Energy injected into grid
T_Amb	Ambient Temperature	PR	Performance Ratio
GlobInc	Global incident in coll. plane		
GlobEff	Effective Global, corr. for IAM and shadings		

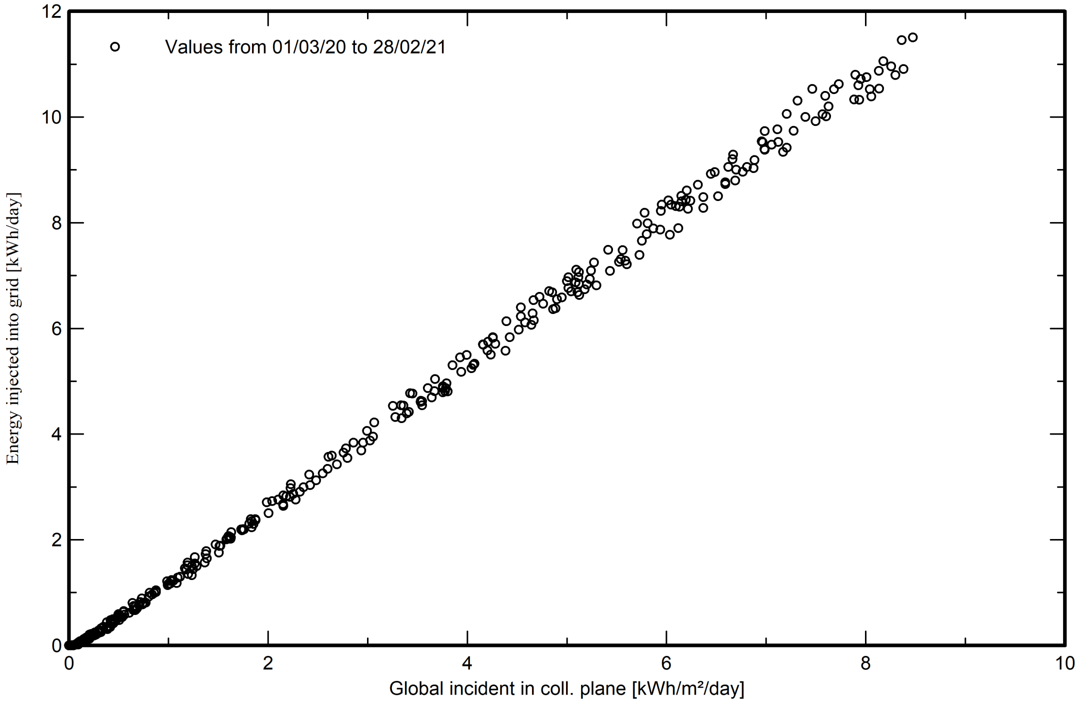
Loss diagram



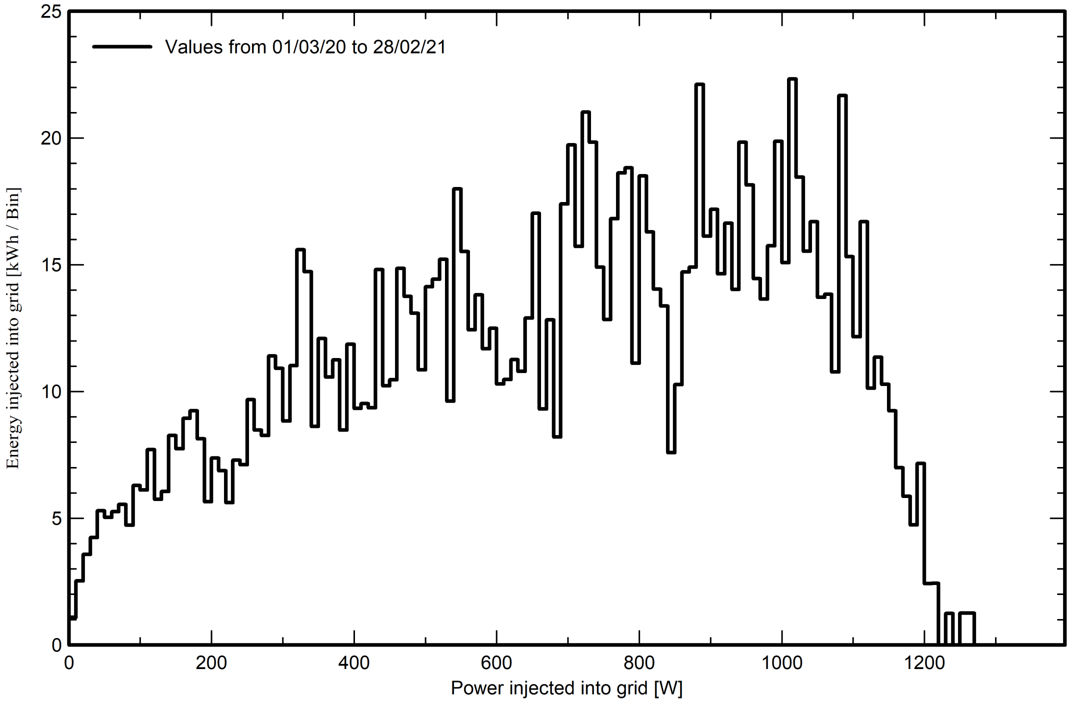
1111 kWh/m²	**Global horizontal irradiation**
-1.99%	**Global incident in coll. plane**
-0.34%	Near Shadings: irradiance loss
-5.18%	IAM factor on global
1029 kWh/m² * 8 m² coll.	**Effective irradiation on collectors**
efficiency at STC = 19.46%	PV conversion
1628 kWh	**Array nominal energy (at STC effic.)**
-6.57%	PV loss due to irradiance level
-0.65%	PV loss due to temperature
-1.35%	Optimizer efficiency loss
+0.40%	Module quality loss
0.00%	Module array mismatch loss
-0.76%	Ohmic wiring loss
1486 kWh	**Array virtual energy at MPP**
-3.31%	Inverter Loss during operation (efficiency)
0.00%	Inverter Loss over nominal inv. power
0.00%	Inverter Loss due to max. input current
0.00%	Inverter Loss over nominal inv. voltage
-0.09%	Inverter Loss due to power threshold
0.00%	Inverter Loss due to voltage threshold
1436 kWh	**Available Energy at Inverter Output**
1436 kWh	**Energy injected into grid**

Special graphs

Daily Input/Output diagram



System Output Power Distribution



C Appendix

Smart Systems
for Solar Power

IBC SOLAR

The Value-Added Modules of the IBC SOLAR Line.

IBC MonoSol 305 VL5, 310 VL5, 315 VL5

First-class solar modules made of monocrystalline silicon (PERC cell concept)

PLUS-WARRANTY
10+5 YEARS
NO STRINGS ATTACHED

- 25 year linear power and 15 year product warranty[1]
- Positive power tolerance (–0/+5 Wp)
- Increased mechanical stability (5400 Pa)[2]
- German warrantor
- 100% tested quality
- Maximum transparent ARC glass

IBC SOLAR – your partner for energy solutions

IBC SOLAR AG has had a successful presence in the photovoltaic market for **more than 35 years** and is one of the leading international energy companies providing high-performance system solutions in every size and for every application with intelligent photovoltaic systems. The **economic strength and financial independence** is confirmed by globally recognised rating agencies.

Smart Systems for Solar Power thanks to perfectly matched components. **More than 1,000 highly qualified partners** around the world, as well as **more than 3,000 megawatts of installed power**, which supply **around 2 million people with solar power**, underline the high level of expertise of IBC SOLAR.

IBC SOLAR – leading PV system integrator from Germany since 1982!

TÜVRheinland ZERTIFIZIERT
OHSAS 18001:2007
ISO 9001:2008
ISO 14001:2004
www.tuv.com
ID 9105069440

TÜVRheinland CERTIFIED
IEC 61215
IEC 61730
www.tuv.com
ID 0000042521

CE

Engineered in GERMANY

The ideal solution for: Home Business

TECHNICAL DATA

IBC MonoSol	305 VL5	310 VL5	315 VL5
Article number	2004200014	2004200015	2004200016

Electrical data (STC):			
STC Power Pmax (Wp)	305	310	315
STC Nominal Voltage Umpp (V)	32.6	33.0	33.1
STC Nominal Current Impp (A)	9.36	9.41	9.53
STC Open Circuit Voltage Uoc (V)	40.2	40.4	40.5
STC Short Circuit Current Isc (A)	9.87	10.01	10.02
Module Efficiency (%)	18.7	19.1	19.4
Power Tolerance (Wp)	−0/+5	−0/+5	−0/+5

Electrical data (NOCT):			
800 W/m² NOCT AM 1.5 Power Pmax (Wp)	224.1	227.8	231.4
800 W/m² NOCT AM 1.5 Nominal Voltage Umpp (V)	29.66	29.86	29.95
800 W/m² NOCT AM 1.5 Open Circuit Voltage Uoc (V)	36.23	36.47	36.56
800 W/m² NOCT AM 1.5 Short Circuit Current Isc (A)	7.97	8.02	8.03
Relative Efficiency Reduction at 200 W/m² (%)	3.5	3.5	3.5

Temperature coefficient:			
NOCT (°C)	44	44	44
Tempcoeff Isc (%/°C)	+0.06	+0.06	+0.06
Tempcoeff Voc (mV/°C)	−112.56	−113.12	−113,40
Tempcoeff Pmpp (%/°C)	−0.38	−0.38	−0.38

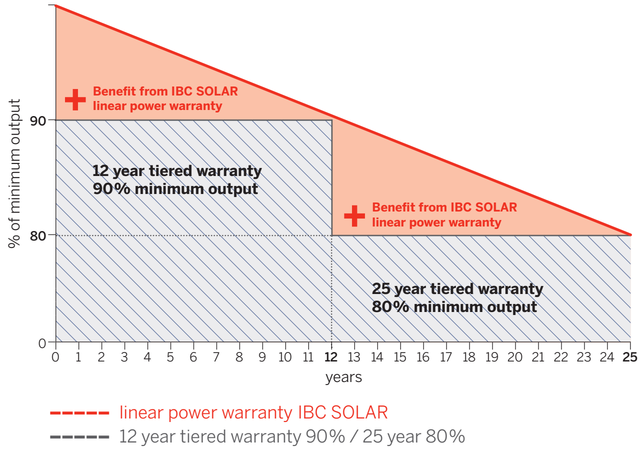
Operating conditions:	
Max. System Voltage (V)	1000
Application Class	A
Reverse Current Ir (A)	20
Current value string fuse (A)	15
Fuse protection from parallel strings	3

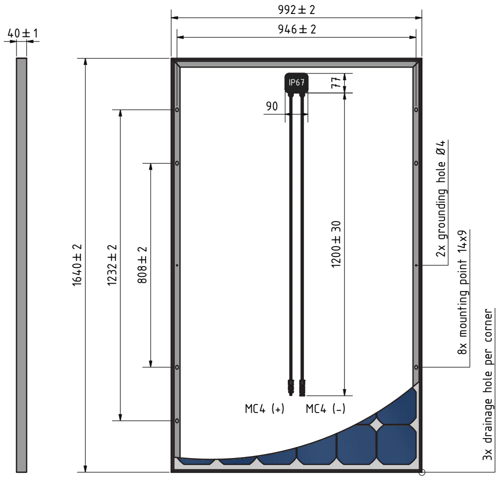
Mechanical properties:	
Dimensions (L × W × H in mm)	1640 × 992 × 40
Weight (kg)	18.5
Load capacity (Pa)[2]	5400
Front sheet (mm)	3.2 (low-iron photovoltaic glass and anti-reflective coating)
Frame	anodized aluminium, sturdy hollow-chamber frame
Cells	6 × 10 monocrystalline silicon cells
Connection type	MC4 (IP65)

Warranties and certification:	
Product warranty	15 years[1]
Power warranty	25 years, linear
Certification	IEC 61215, IEC 61730-1/-2, ISO 9001, ISO 14001, OHSAS 18001

Packaging information:	
Number of modules per pallet	26
Number of pallets per 40' container	28
Number of pallets per lorry	30
Dimensions incl. pallet (L × W × H in mm)	1680 × 1135 × 1145
Gross weight incl. pallet (kg)	520
Stackability per pallet	3-fold

25 year linear power warranty by IBC SOLAR





Presented by:

[1] The 15 year product warranty is only valid for installations within Europe and Japan. The warranty requires installation according to the valid installation instructions. Standard test conditions: 1000 W/m² irradiation with a spectral distribution of AM 1.5 and a cell temperature of 25°C. 800 W/m², NOCT. Information according to EN 60904-3 (STC). All values according to DIN EN 50380. Errors and changes reserved.
The precise conditions and content can be taken from the respectively valid version of the product and power warranty, which you can obtain from your IBC Premium Partner.

[2] Tested according IEC 61215 for snow loads up to 5,400 Pa (5.4 kN/m²).

As of: 2017-11-21
9001500067

Smart Systems
for Solar Power

IBC
SOLAR

The Value-Added Modules of the IBC SOLAR Line.

IBC PolySol 265 CS5, 270 CS5, 275 CS5

First-class solar modules made of polycrystalline silicon

PLUS-WARRANTY
10+5 YEARS
NO STRINGS ATTACHED

- 25 year linear power and 15 year product warranty[1]
- Positive power tolerance (–0/+5 Wp)
- Increased mechanical stability (5400 Pa)[2]
- German warrantor
- 100% tested quality
- Maximum transparent ARC glass

IBC SOLAR – your partner for energy solutions

IBC SOLAR AG has had a successful presence in the photovoltaic market for **more than 35 years** and is one of the leading international energy companies providing high-performance system solutions in every size and for every application with intelligent photovoltaic systems. The **economic strength and financial independence** is confirmed by globally recognised rating agencies.

Smart Systems for Solar Power thanks to perfectly matched components. **More than 1,000 highly qualified partners** around the world, as well as **more than 3,000 megawatts of installed power**, which supply **around 2 million people with solar power**, underline the high level of expertise of IBC SOLAR.

IBC SOLAR – leading PV system integrator from Germany since 1982!

TÜVRheinland ZERTIFIZIERT
OHSAS 18001:2007
ISO 9001:2008
ISO 14001:2004
www.tuv.com
ID 9105069440

TÜVRheinland CERTIFIED
IEC 61215
IEC 61730
www.tuv.com
ID 1419040636

CE

Engineered in GERMANY

The ideal solution for: Home Business Power Plant

TECHNICAL DATA

IBC PolySol	265 CS5	270 CS5	275 CS5
Article number	2203800026	2203800027	2203800028

Electrical data (STC):			
STC Power Pmax (Wp)	265	270	275
STC Nominal Voltage Umpp (V)	31.4	31.7	31.8
STC Nominal Current Impp (A)	8.44	8.50	8.67
STC Open Circuit Voltage Uoc (V)	38.6	38.9	39.0
STC Short Circuit Current Isc (A)	9.03	9.08	9.23
Module Efficiency (%)	16.2	16.5	16.8
Power Tolerance (Wp)	−0/+5	−0/+5	−0/+5

Electrical data (NOCT):			
800 W/m² NOCT AM 1.5 Power Pmax (Wp)	198.6	202.3	206.1
800 W/m² NOCT AM 1.5 Nominal Voltage Umpp (V)	29.1	29.2	29.5
800 W/m² NOCT AM 1.5 Open Circuit Voltage Uoc (V)	34.91	35.10	35.35
800 W/m² NOCT AM 1.5 Short Circuit Current Isc (A)	7.61	7.68	7.77
Relative Efficiency Reduction at 200 W/m² (%)	1.5	1.5	1.5

Temperature coefficient:			
NOCT (°C)	44.1	44.1	44.1
Tempcoeff Isc (%/°C)	+0.041	+0.041	+0.041
Tempcoeff Voc (mV/°C)	−120	−121	−122
Tempcoeff Pmpp (%/°C)	−0.411	−0.411	−0.411

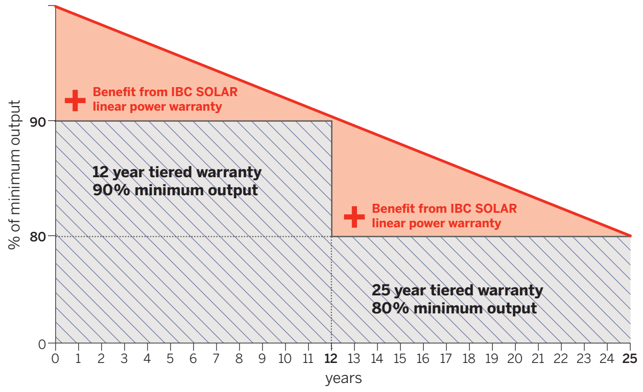
Operating conditions:	
Max. System Voltage (V)	1000
Application Class	A
Reverse Current Ir (A)	20
Current value string fuse (A)	15
Fuse protection from parallel strings	4

Mechanical properties:	
Dimensions (L × W × H in mm)	1650 × 992 × 40
Weight (kg)	19.5
Load capacity (Pa)[2]	5400
Front sheet (mm)	3.2 (low-iron photovoltaic glass and anti-reflective coating)
Frame	anodized aluminium, sturdy hollow-chamber frame
Cells	6 × 10 polycrystalline silicon cells
Connection type	MC4 (IP65)

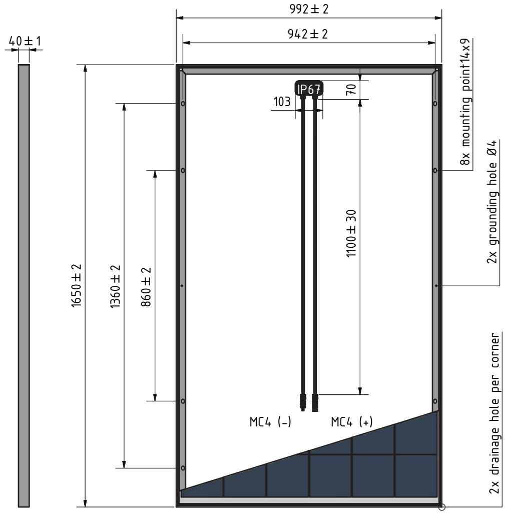
Warranties and certification:	
Product warranty	15 years[1]
Power warranty	25 years, linear
Certification	IEC 61215, IEC 61730-1/-2, ISO 9001, ISO 14001, OHSAS 18001

Packaging information:	
Number of modules per pallet	26
Number of pallets per 40' container	28
Number of pallets per lorry	30
Dimensions incl. pallet (L × W × H in mm)	1695 × 1130 × 1150
Gross weight incl. pallet (kg)	520
Stackability per pallet	3-fold

25 year linear power warranty by IBC SOLAR



----- linear power warranty IBC SOLAR
----- 12 year tiered warranty 90% / 25 year 80%



Presented by:

[1] The 15 year product warranty is only valid for installations within Europe and Japan. The warranty requires installation according to the valid installation instructions. Standard test conditions: 1000 W/m² irradiation with a spectral distribution of AM 1.5 and a cell temperature of 25 °C. 800 W/m², NOCT. Information according to EN 60904-3 (STC). All values according to DIN EN 50380. Errors and changes reserved.
The precise conditions and content can be taken from the respectively valid version of the product and power warranty, which you can obtain from your IBC Premium Partner.

[2] Tested according IEC 61215 for snow loads up to 5,400 Pa (5.4 kN/m^2).

As of: 2017-11-28
9001500065

Results of initial characterization of ERTEX VSG-L modules:

ERTEX SOLARTECHNIK GmbH
www.ertex-solar.at
Type: VSG-L 227,26Wp
Max. Sys.Spannung: 1000V
IP 54; Schutzklasse II
Auftrags.Nr.FA17-0107
Typ 1 Serien Nr.M17-02284

Type: ERTEX VSG-L, *Dimension:* 1.100 mm x 1.600 mm x 11,52 mm
54 Poly TT Cells (156 x 156 mm),

Cell spacing: 5 mm by 5 mm;
Edge distances: On top 78 mm, bottom 78 mm, left 69,5 mm, right 69,5 mm
Front glass: ESG 5 mm Diamant
Front encapsulant: PVB
6 x 9 interconnected cells
Back encapsulant: PVB
Back side glass: ESG 5 mm, fully enamelled / Edges grinded / HS-Test
Bore hole Ø 5-30 mm: 3.

Total thickness: 11,52 mm
Nominal power Pmp @ STC: 227,26 Wp (+/-5%)
Isc: 8,34A
Imp: 7,89A
Voc: 34,07V
Vmp: 28,84V
Temp. Coeff. Voc: -0,32 %/K
Temp. Coeff. Isc: 0,05 %/K
Temp. Coeff. Pmp: -0,37 %/K
Weight : 47,237 kg
Module transparency: 25,33%

Figure 1: Module label, and information translated from the data sheet.

Electroluminescence imaging, see Figure 2.

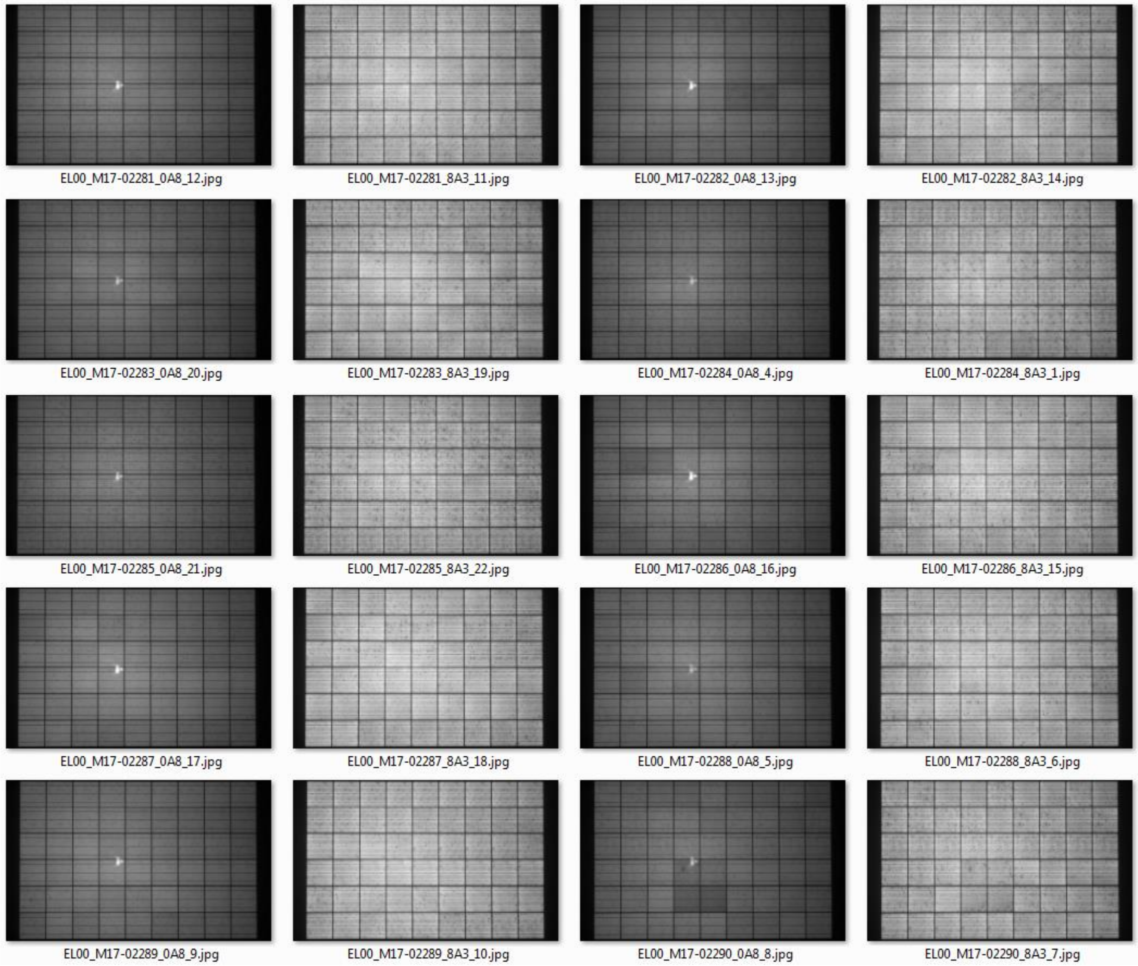

EL00_M17-02281_0A8_12.jpg
EL00_M17-02281_8A3_11.jpg
EL00_M17-02282_0A8_13.jpg
EL00_M17-02282_8A3_14.jpg
EL00_M17-02283_0A8_20.jpg
EL00_M17-02283_8A3_19.jpg
EL00_M17-02284_0A8_4.jpg
EL00_M17-02284_8A3_1.jpg
EL00_M17-02285_0A8_21.jpg
EL00_M17-02285_8A3_22.jpg
EL00_M17-02286_0A8_16.jpg
EL00_M17-02286_8A3_15.jpg
EL00_M17-02287_0A8_17.jpg
EL00_M17-02287_8A3_18.jpg
EL00_M17-02288_0A8_5.jpg
EL00_M17-02288_8A3_6.jpg
EL00_M17-02289_0A8_9.jpg
EL00_M17-02289_8A3_10.jpg
EL00_M17-02290_0A8_8.jpg
EL00_M17-02290_8A3_7.jpg

Figure 2: EL imaging pictures @ 100% Isc (8.3 A) and 10% Isc (0.8 A).

No cracks visible in EL images, approx. similar brightness of all cells, no edge wafers processed into the modules.

Note: The white spot left of the module center is a reflection only, and not a feature of the modules.

<u>IV-Characterization results summary</u>, see Table 1.

***Table 1:** Measurement of IV characteristics at standard test conditions STC (25°C, AM1.5, 1000 W/m²) before and after light soaking (stabilization), and at low irradiance conditions LIC (25°C, AM1.5, 200 W/m²)*

Serial number		Voc [V]	Vmp[V]	Isc[A]	Imp [A]	Pmax [W]	FF [%]	η_m [%]	η_c [%]
M17-02281	initial	34.304	28.090	8.300	7.859	220.74	77.53	12.54	16.80
	after preconditioning	34.238	27.998	8.288	7.843	219.57	77.38	12.48	16.71
	difference	-0.19%	-0.33%	-0.15%	-0.20%	-0.53%	-0.19%	-0.53%	-0.53%
	low irradiance (200m/W2)	31.969	26.984	1.655	1.563	42.17	79.72	11.98	16.05
M17-02282	initial	34.269	28.010	8.302	7.862	220.21	77.40	12.51	16.76
	after preconditioning	34.178	27.884	8.292	7.832	218.40	77.06	12.41	16.62
	difference	-0.27%	-0.45%	-0.12%	-0.37%	-0.82%	-0.44%	-0.82%	-0.82%
	low irradiance (200m/W2)	31.836	26.817	1.655	1.563	41.91	79.54	11.91	15.95
M17-02283	initial	34.242	27.964	8.316	7.866	219.97	77.25	12.50	16.74
	after preconditioning	34.141	27.827	8.290	7.832	217.93	77.00	12.38	16.58
	difference	-0.29%	-0.49%	-0.32%	-0.44%	-0.93%	-0.31%	-0.93%	-0.93%
	low irradiance (200m/W2)	31.836	26.817	1.655	1.563	41.91	79.54	11.91	15.95
M17-02284	initial	34.215	28.008	8.285	7.851	219.89	77.57	12.49	16.73
	after preconditioning	34.117	27.885	8.257	7.820	218.05	77.40	12.39	16.59
	difference	-0.29%	-0.44%	-0.33%	-0.40%	-0.83%	-0.22%	-0.83%	-0.83%
	low irradiance (200m/W2)	31.819	26.807	1.653	1.563	41.89	79.63	11.90	15.94
M17-02285	initial	34.250	28.002	8.292	7.851	219.84	77.41	12.49	16.73
	after preconditioning	34.140	27.854	8.273	7.829	218.06	77.21	12.39	16.59
	difference	-0.32%	-0.53%	-0.23%	-0.28%	-0.81%	-0.26%	-0.81%	-0.81%
	low irradiance (200m/W2)	31.844	26.884	1.646	1.561	41.96	80.03	11.92	15.96
M17-02286	initial	34.300	28.068	8.306	7.865	220.74	77.48	12.54	16.80
	after preconditioning	34.203	27.937	8.285	7.838	218.97	77.27	12.44	16.66
	difference	-0.28%	-0.46%	-0.25%	-0.34%	-0.80%	-0.27%	-0.80%	-0.80%
	low irradiance (200m/W2)	31.899	26.916	1.653	1.565	42.12	79.89	11.97	16.03
M17-02287	initial	34.303	28.014	8.378	7.911	221.62	77.12	12.59	16.86
	after preconditioning	34.212	27.894	8.346	7.883	219.89	77.01	12.49	16.73
	difference	-0.26%	-0.43%	-0.37%	-0.36%	-0.78%	-0.15%	-0.78%	-0.78%
	low irradiance (200m/W2)	31.909	26.912	1.666	1.573	42.32	79.59	12.02	16.10
M17-02288	initial	34.250	27.995	8.354	7.884	220.73	77.14	12.54	16.80
	after preconditioning	34.187	27.927	8.332	7.865	219.64	77.11	12.48	16.71
	difference	-0.18%	-0.24%	-0.26%	-0.25%	-0.49%	-0.04%	-0.49%	-0.49%
	low irradiance (200m/W2)	31.912	26.938	1.666	1.571	42.33	79.61	12.03	16.11
M17-02289	initial	34.258	28.023	8.308	7.867	220.46	77.46	12.53	16.78
	after preconditioning	34.201	27.942	8.293	7.854	219.46	77.38	12.47	16.70
	difference	-0.16%	-0.29%	-0.18%	-0.17%	-0.45%	-0.11%	-0.45%	-0.45%
	low irradiance (200m/W2)	31.905	26.916	1.656	1.566	42.16	79.79	11.98	16.04
M17-02290	initial	34.225	27.992	8.292	7.852	219.79	77.45	12.49	16.73
	after preconditioning	34.145	27.878	8.259	7.826	218.19	77.37	12.40	16.60
	difference	-0.23%	-0.41%	-0.39%	-0.33%	-0.73%	-0.11%	-0.73%	-0.73%
	low irradiance (200m/W2)	31.865	26.895	1.649	1.562	42.00	79.90	11.93	15.98

No irregularities, fill factor above 77% for all 10 modules, light induced degradation effects (LID after stabilization) below -1%.

Max. power comparison: rated power 227.26 W +/-5%, power after stabilization measured range from 219.89 to 217.93 W (-3.2% … -4.1%) is within the +/-5% tolerance.

Measurements according to Energy rating IEC 61853-1: Irradiance and temperature matrix

In Table 2 … 5 the measurement results are listed for maximum power Pmp, short circuit current Isc and open circuit voltage Voc, with parameters module temperature Tmod and irradiation G at AM1.5. As the modules are almost identical in output, and fairly linear, only a single module, serial number M17-02285, was measured. These data were also used to derive the relative thermal coefficients **γ** for max. power, **α** for short circuit current, and **β** for the open circuit voltage in %/°C, close to the ones given in the data sheet, see Figure 1.

Table 2: *Maximum power Pmp at different temperature Tmod and irradiance values G.*

Tmod / G	15°C	25°C	50°C	60°C
1100 W/m2	/	239.26	215.11	204.84
1000 W/m2	226.62	217.95	195.93	186.76
800 W/m2	181.48	174.61	156.80	149.55
600 W/m2	135.53	130.75	117.28	111.88
400 W/m2	89.21	86.08	77.15	/
200 W/m2	43.50	41.86	37.48	/
100 W/m2	20.85	20.75	/	/

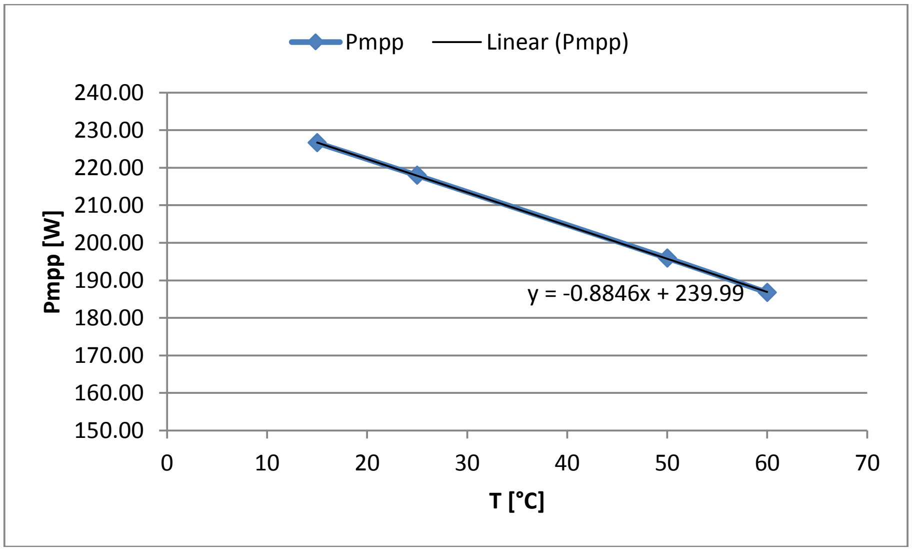


Figure 3: Linear fit for the Pmp temperature coefficient @ 1000W/m^2: **γ = -0,405%/°C**.

Table 3: *Short circuit current Isc at different temperature Tmod and irradiance values G.*

	15°C	25°C	50°C	60°C
1100 W/m2	/	9.08	9.18	9.22
1000 W/m2	8.22	8.26	8.35	8.39
800 W/m2	6.57	6.60	6.68	6.71
600 W/m2	4.92	4.95	5.01	5.03
400 W/m2	3.28	3.29	3.33	/
200 W/m2	1.63	1.64	1.67	/
100 W/m2	0.81	0.83	/	/

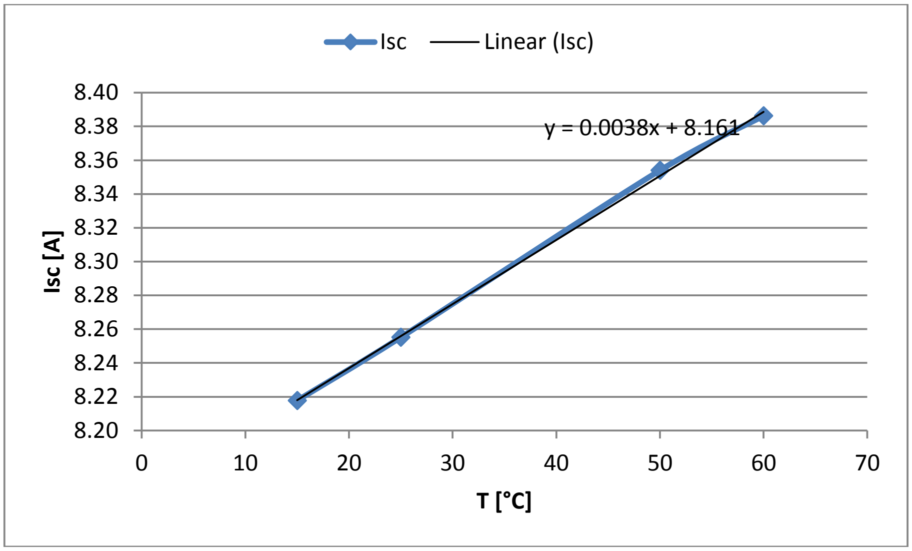


Figure 4: Linear fit for the Isc temperature coefficient @1000W/m²: **α = +0,046%/°C**.

Table 4: *Open circuit voltage Voc at different temperature Tmod and irradiance values G.*

	15°C	25°C	50°C	60°C
1100 W/m2	/	34.30	31.67	30.60
1000 W/m2	35.21	34.15	31.53	30.44
800 W/m2	34.88	33.83	31.15	30.08
600 W/m2	34.46	33.41	30.68	29.60
400 W/m2	33.86	32.81	30.04	/
200 W/m2	32.86	31.87	28.93	/
100 W/m2	31.81	31.56	/	/

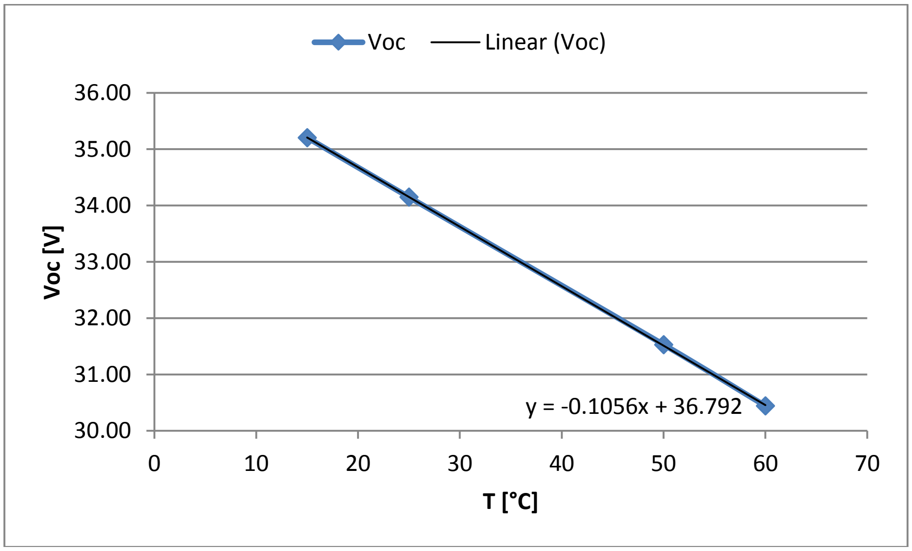


Figure 5: Linear fit for the Voc temperature coefficient @1000W/m²: **β = -0,309%/°C**.

Measurements according to Energy rating IEC 61853-2: Spectral response SR

The same module, as for the 61853-1 measurements was used for the spectral response measurement, with the results in Table 5 and the SR curves in Figure 6.

Table 5: *Spectral response measurement results: Filter center wavelengths in the range from 400 nm to 1100 nm (1st column); 4th column absolute spectral response of the Isc; 5th column: relative SR with the max. set to 100%.*

DUT id: Ertex VSG-L M17_02285

Filters (nm)	Isc Regr. [A]	Calibration factors	SR [(mA*m²)/W]	SR (a.u.) [%]
400	0.2200938	22.376	4.924883644	34%
450	0.5012851	14.796	7.416808061	52%
500	0.6930413	12.282	8.51191281	59%
550	0.752139	12.603	9.479010412	66%
600	0.7741655	13.473	10.43032995	73%
650	0.8273608	13.544	11.20558157	78%
700	0.7999826	15.059	12.04693502	84%
750	0.7184389	17.786	12.77784136	89%
800	0.7256255	18.620	13.51143778	94%
850	0.5285614	26.563	14.04008901	98%
900	0.5888012	24.384	14.35705475	100%
950	0.4293332	33.182	14.24626995	99%
1000	0.4992774	25.817	12.88980498	90%
1050	0.2377533	39.185	9.316417886	65%
1100	0.1064143	42.166	4.487042777	31%

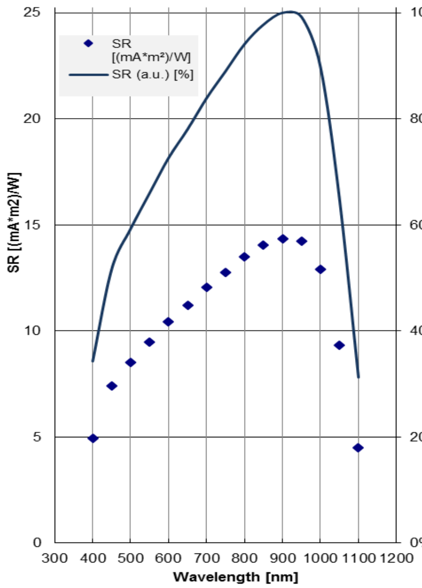


Figure 6: SR measurement.

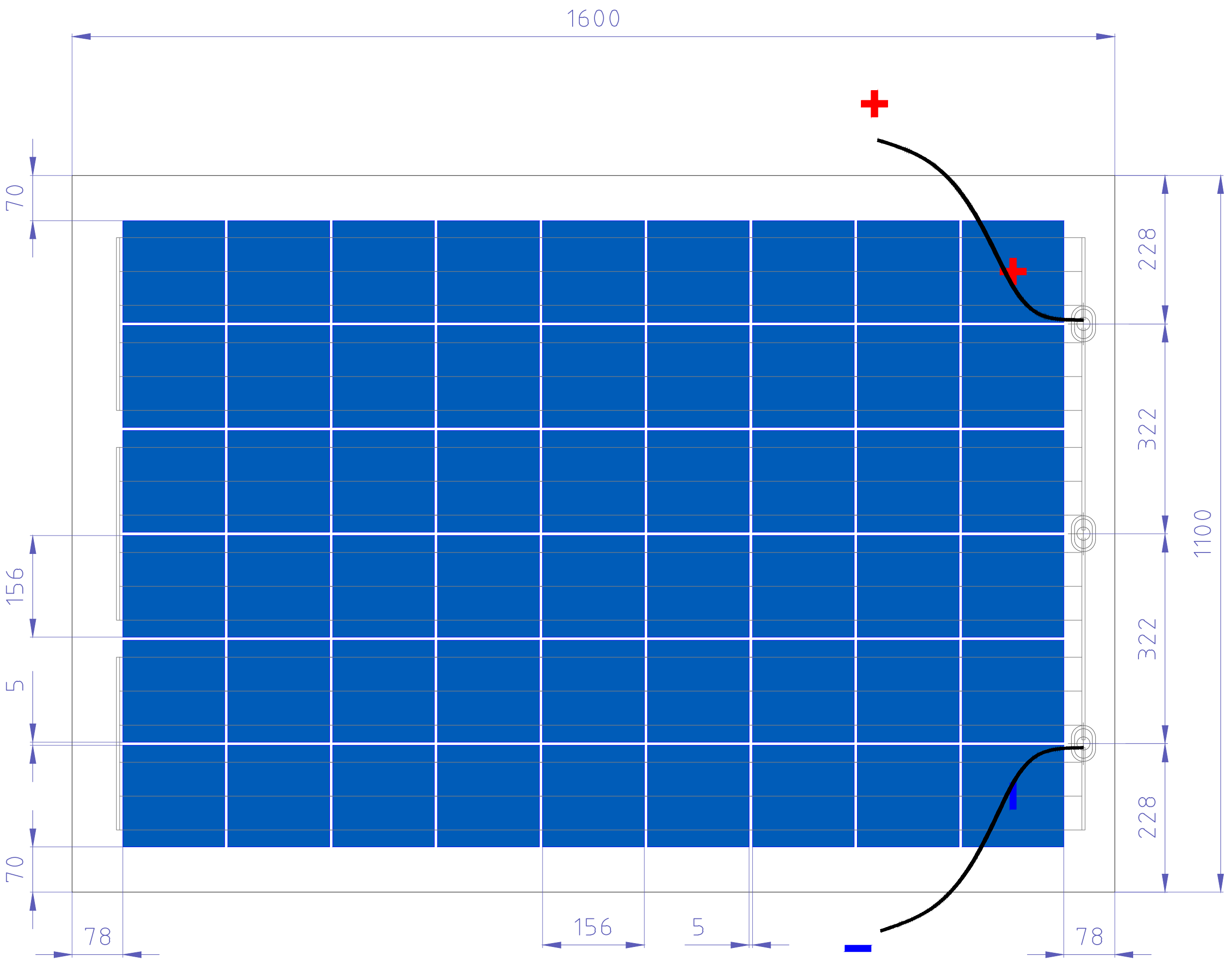
The SR measurement result is typical for modules based on multi-crystalline Silicon cells.

Note: In part 2 of the energy rating standard series a measurement procedure is defined for angular effects (angle of incidence AOI) by means of a special prepared device with a single cell contacted and surrounded by other cells with similar composition and cell spacing as in the full size modules. The determination of the coefficients for estimating the module's cell temperature T_m above ambient T_{amb}, depending on wind speed v (coefficient u_1) and irradiation G (coefficient u_0) with the formula $T_m – T_{amb} = G / (u_0 + u_1 v)$ and the normal module operating temperature NMOT is depending on the mounting conditions. As such a special device for AOI measurement was not available, and the mounting conditions will change from site to site, only the spectral response measurement SR was executed.

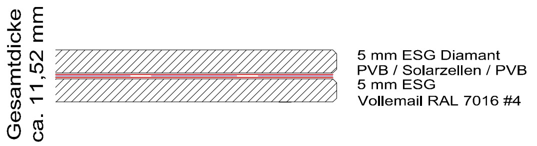
For the indoor characterization a Meyer Burger / PASAN High$^{\text{LIGHT}}$ A$^+$A$^+$A$^+$ flasher system with a max. DUT area of 3 m × 3 m was used.

Jan Slamberger, Karl A. Berger, AIT, August 2017

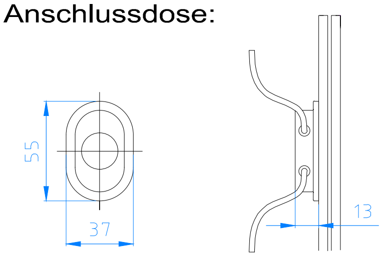
Appendix on following page: Dimensional drawing of this module type.



Glasaufbau:



Anschlussdose:



FERTIGUNGSTOLERANZEN
LT. VA AA-QW-025

Projekt. Nr.	Typ	Stückzahl
PRO17-XXX	Typ 1	8

Technische Daten

Zellen horizontal:	9
Zellen vertikal:	6
Zellen pro Diode:	18
Zellenanzahl:	54
Kabellänge:	noch klären
Steckertyp:	noch klären
Gesamtgewicht:	ca. 44 kg

Projekttitel
Task 15

Kunde/Bauherr
-

Maßstab u. Papier
1/- A4 Hochformat

Bemerkung
Ansicht=Aussenseite

Zeichnungsstatus: Entwurf

Name: Korn
Datum: 03.04.2017

Version
1.0

ertex solar
Energy Meets Architecture

Dateiname: X:\ertex-solar\20 CAD-Zeichnungen\Projekte 2017\PRO17-XXX_Task 15\ZE_MK_PRO17-XXX_Task 15_3.4.17_V1.0.dwg

NEW
From Solax
SINGLE PHASE
INVERTERS
SOLAX
POWER
X1 MINI
X1 AIR
X1 BOOST

	X1 MINI				X1 AIR			X1 BOOST				
	X1-0.7	X1-1.1	X1-1.5	X1-2.0	X1-2.5	X1-3.0	X1-3.3	X1-3.0T	X1-3.3T	X1-3.6T	X1-4.2T	X1-5.0T
INPUT (DC)												
Max.recommended DC power [W]	840	1250	1650	2200	2700	3200	3450	3250	3500	4000	4600	5200
Max. input DC voltage [V]	400	400	400	400	600	600	600	600	600	600	600	600
Max. input current [A]	10	10	10	10	10	10	10	12/12	12/12	12/12	12/12	12/12
MPPT voltage range [V]	50-380	55-380	55-380	55-380	100-580	100-580	100-580	125-580	125-580	125-580	125-580	125-580
Start output voltage [V]	70	70	70	70	120	120	120	150	150	150	150	150
Number of MPP tracker/strings per MPP tracker	1/1	1/1	1/1	1/1	1/1	1/1	1/1	2/1	2/1	2/1	2/1	2/1
OUTPUT (AC)												
AC norminal power [VA]	700	1100	1500	2000	2500	3000	3300	3000	3300	3680	4200	4600
Max. AC power [VA]	700	1100	1500	2000	2500	3000	3300	3000	3300	3680	4200	5000
Norminal AC voltage; range [V]	220/230/240; 180~280				220/230/240; 180~280			220/230/240; 180~280				
AC grid frequency; range) [Hz]	50/60; ±5				50/60; ±5			50/60; ±5				
Max. AC current [A]	3.5	5.5	7.5	9.5	12	14	15	14	15	16	19	21
Power factor (full load)	0.8 leading ~ 0.8 lagging				0.8 leading ~ 0.8 lagging			0.8 leading ~ 0.8 lagging				
Total harmonic distortion(THD) [%]	<1.5				<2			<2				
POWER CONSUMPTION												
Standby comsuption power[W]	<5				<10			<10				
EFFICIENCY												
MPPT efficiency[%]	99.9				99.9			99.9				
Euro efficiency[%]	95	95.5	96	96.5	96.8			97.0				
Max. efficiency[%]	97.1				97.6			97.8				
SAFETY & PROTECTION												
Over voltage protection	YES				YES			YES				
Over current protection	YES				YES			YES				
DC isolation impedance monitoring	YES				YES			YES				
Ground fault current monitoring	YES				YES			YES				
DC injection monitoring	YES				YES			YES				
RCD protection	YES				YES			YES				
Safety	EN62109-1/-2				EN62109-1/-2			IEC62109-1/-2				
EMC	EN61000-6-2;EN61000-6-3;EN61000-3-2;EN61000-3-3				EN61000-6-2;EN61000-6-3;EN61000-3-2;EN61000-3-3			EN 61000-6-1 / EN 61000-6-2 / EN 61000-6-3				
Certification	G83/2;EN50438				G83/2;AS4777.2-2015;VDE4105; EN50438; CQC			G83/2;G59/3;AS4777.2-2015;VDE4105; EN50438;CQC;VDE0126				
ENVIRONMENT LIMITS												
Protection class	IP65				IP65			IP65				
Operating temperature[°C]	-20~ +60(derating at 45)				-20~ +60(derating at 45)			-20~ +60(derating at 45)				
Humidity[%]	0~95 (no condensation)				0~95 (no condensation)			0~95 (no condensation)				
Attitude[m]	2000				2000			2000				
Storage temperature[°C]	-20~+60				-20~+60			-20~+60				
Noise emission[dB]	<25				<30			<25				
DIMENSION & WEIGHT												
Dimensions(W*H*D) [mm]	248*350*124				323*402*119			339*420*143				
Weight[Kg]	7				9.5			14.6	14.6	14.6	16.7	16.7
GENERAL DATA												
Topology	Transformerless				Transformerless			Transformerless				
Communication interface	RS485 / WIFI(optional) / DRM / USB				RS485 / WIFI(optional) / DRM / USB			RS485 / WIFI(optional) / DRM / USB / RF(optional) / Meter(optional)				
Display	6 LED				11 LED			Backlight 16*4 character				
Standard warranty[years]	5-10				5-10			5-10				
Cooling type	Natural				Natural			Natural				

**Can be modified without notice.(V2)*

D Appendix

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 11 10:44:20 2021
@author: ghaliry
"""
import datetime
from suntime import Sun, SunTimeException
import pandas as pd
from datetime import date

def sunrise_sunset_filtering(latitude,longitude,raw_data,date_col):
    #This function takes as arguments the following:
        # site latitude and longitude
        # the raw data
        # and the timestamp column name
    sun = Sun(latitude, longitude)
    raw_data[date_col]= pd.to_datetime(raw_data[date_col],
            format='%Y-%m-%d %H:%M:%S')
    days=raw_data[date_col].dt.day
    years=raw_data[date_col].dt.year
    months=raw_data[date_col].dt.month
    start_date = date(years.head(1), months.head(1), days.head(1))
    end_date = date(years.tail(1), months.tail(1), days.tail(1))
    duration = pd.date_range(start_date,end_date)

    new_data=pd.DataFrame(columns=raw_data.columns)
    for day in duration:
        target_day=datetime.date(int(day.year), int(day.month), int(day.day))
        target_day_sr = sun.get_local_sunrise_time(target_day)
        target_day_ss = sun.get_local_sunset_time(target_day)
        day_light = pd.date_range(target_day_sr,target_day_ss,freq='1min')
        day_light=pd.DataFrame(day_light.strftime('%Y-%m-%d %H:%M:%S'),
                               columns=['DATETIME'])
        new_data=pd.concat((new_data,day_light))

    raw_data[date_col]= raw_data[date_col].dt.strftime('%Y-%m-%d %H:%M')
    raw_data=raw_data.set_index(date_col)
    new_data[date_col]= pd.to_datetime(new_data['DATETIME'],
            format='%Y-%m-%d %H:%M:%S')
    new_data[date_col]= new_data[date_col].dt.strftime('%Y-%m-%d %H:%M')
    new_data=new_data.set_index(date_col)
    Filtered_data=raw_data.loc[new_data.index]
    return Filtered_data

raw_data=pd.read_excel('Data_from_DB_D3_all.xlsx', usecols=[0,1,3,4,])
#raw_data=pd.read_excel('Oct_Avg.xlsx', usecols=[0,4,5,7,9,11])

lat= 58.33461
long = 8.57586

filtered_data=sunrise_sunset_filtering(lat,long,raw_data,'DATETIME')
filtered_data.to_excel('D3_filtered.xlsx')
```

```
# -*- coding: utf-8 -*-
"""
Created on Wed Apr  7 11:05:08 2021
@author: Sigrid
"""
import pandas as pd
import datetime
import numpy as np

M2 = pd.read_excel('M2_filtered.xlsx')
U2 = pd.read_excel('U2_filtered.xlsx')
C3 = pd.read_excel('C3_filtered.xlsx')
D3 = pd.read_excel('D3_filtered.xlsx')

#deviding the data based on months.
MarM2 = M2.loc[0:22172,:]
AprM2 = M2.loc[22173:48280,:]
MayM2 = M2.loc[48281:79636,:]
JunM2 = M2.loc[79637:112341,:]
JulM2 = M2.loc[112342:144781,:]
AugM2 = M2.loc[144782:173187,:]
SepM2 = M2.loc[173188:196145,:]
OctM2 = M2.loc[196146:215112,:]
NovM2 = M2.loc[215113:229207,:]
DecM2 = M2.loc[229208:241305,:]
JanM2 = M2.loc[241306:254776,:]
FebM2 = M2.loc[254777:270488,:]
Mar21_M2 = M2.loc[270489:292502,:]

Mar_nan = MarM2.isna().sum().sum
Apr_nan = AprM2.isna().sum().sum
May_nan = MayM2.isna().sum().sum
Jun_nan = JunM2.isna().sum().sum
Jul_nan = JulM2.isna().sum().sum
Aug_nan = AugM2.isna().sum().sum
Sep_nan = SepM2.isna().sum().sum
Oct_nan = OctM2.isna().sum().sum
Nov_nan = NovM2.isna().sum().sum
Dec_nan = DecM2.isna().sum().sum
Jan_nan = JanM2.isna().sum().sum
Feb_nan = FebM2.isna().sum().sum
Mar21_nan= Mar21_M2.isna().sum().sum

print ('Count of NaN: ' + str(Feb_nan))

#The division into months is the same for each PV module. Therefore the
#same code may be used for all 4 systems, by changing the .loc .
```

```
# -*- coding: utf-8 -*-
"""
Created on Wed Mar 24 10:09:06 2021
@author: Sigrid
"""
#Yield caluclations for all systems over 13 months
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

MonoM2_20 = pd.read_excel('Data_from_DB_M2_all.xlsx')
MonoU2_20 = pd.read_excel('Data_from_DB_U2_all.xlsx')
PolyC3 = pd.read_excel('Data_from_DB_C3_all.xlsx')
PolyD3 = pd.read_excel('Data_from_DB_D3_all.xlsx')

#getting the ouptu power (Wh)
MarM2 = MonoM2_20.loc[0:21607,['PowerDC']]
AprM2 = MonoM2_20.loc[21608:47426,['PowerDC']]
MayM2 = MonoM2_20.loc[47427:78738,['PowerDC']]
JunM2 = MonoM2_20.loc[78739:110617,['PowerDC']]
JulM2 = MonoM2_20.loc[110618:142544,['PowerDC']]
AugM2 = MonoM2_20.loc[142545:170624,['PowerDC']]
SepM2 = MonoM2_20.loc[170625:193034,['PowerDC']]
OctM2 = MonoM2_20.loc[193035:211137,['PowerDC']]
NovM2 = MonoM2_20.loc[211138:224040,['PowerDC']]
DecM2 = MonoM2_20.loc[224041:231916,['PowerDC']]
JanM2 = MonoM2_20.loc[231917:244672,['PowerDC']]

MarM2_sum = np.sum(MarM2)*(1/60)
AprM2_sum = np.sum(AprM2)*(1/60)
MayM2_sum = np.sum(MayM2)*(1/60)
JunM2_sum = np.sum(JunM2)*(1/60)
JulM2_sum = np.sum(JulM2)*(1/60)
AugM2_sum = np.sum(AugM2)*(1/60)
SepM2_sum = np.sum(SepM2)*(1/60)
OctM2_sum = np.sum(OctM2)*(1/60)
NovM2_sum = np.sum(NovM2)*(1/60)
DecM2_sum = np.sum(DecM2)*(1/60)
JanM2_sum = np.sum(JanM2)*(1/60)

#calculating the Final yield
MarM2_yield = MarM2_sum/(315)
AprM2_yield = AprM2_sum/(315)
MayM2_yield = MayM2_sum/(315)
JunM2_yield = JunM2_sum/(315)
JulM2_yield = JulM2_sum/(315)
AugM2_yield = AugM2_sum/(315)
SepM2_yield = SepM2_sum/(315)
OctM2_yield = OctM2_sum/(315)
NovM2_yield = NovM2_sum/(315)
DecM2_yield = DecM2_sum/(315)
JanM2_yield = JanM2_sum/(315)

YieldM2 = np.array([MarM2_yield, AprM2_yield, MayM2_yield, JunM2_yield,
                    JulM2_yield, AugM2_yield, SepM2_yield, OctM2_yield,
                    NovM2_yield, DecM2_yield, JanM2_yield])
#getting the ouptu power (Wh)
```

```
MarU2 = MonoU2_20.loc[0:21538,['PowerDC']]
AprU2 = MonoU2_20.loc[21539:47426,['PowerDC']]
MayU2 = MonoU2_20.loc[47427:78689,['PowerDC']]
JunU2 = MonoU2_20.loc[78690:110565,['PowerDC']]
JulU2 = MonoU2_20.loc[110566:142469,['PowerDC']]
AugU2 = MonoU2_20.loc[142470:170499,['PowerDC']]
SepU2 = MonoU2_20.loc[170500:192866,['PowerDC']]
OctU2 = MonoU2_20.loc[192867:210887,['PowerDC']]
NovU2 = MonoU2_20.loc[210888:223907,['PowerDC']]
DecU2 = MonoU2_20.loc[223908:232838,['PowerDC']]
JanU2 = MonoU2_20.loc[232839:244671,['PowerDC']]

MarU2_sum = np.sum(MarU2)*(1/60)
AprU2_sum = np.sum(AprU2)*(1/60)
MayU2_sum = np.sum(MayU2)*(1/60)
JunU2_sum = np.sum(JunU2)*(1/60)
JulU2_sum = np.sum(JulU2)*(1/60)
AugU2_sum = np.sum(AugU2)*(1/60)
SepU2_sum = np.sum(SepU2)*(1/60)
OctU2_sum = np.sum(OctU2)*(1/60)
NovU2_sum = np.sum(NovU2)*(1/60)
DecU2_sum = np.sum(DecU2)*(1/60)
JanU2_sum = np.sum(JanU2)*(1/60)

#calculating the Final yield
MarU2_yield = MarU2_sum/(315)
AprU2_yield = AprU2_sum/(315)
MayU2_yield = MayU2_sum/(315)
JunU2_yield = JunU2_sum/(315)
JulU2_yield = JulU2_sum/(315)
AugU2_yield = AugU2_sum/(315)
SepU2_yield = SepU2_sum/(315)
OctU2_yield = OctU2_sum/(315)
NovU2_yield = NovU2_sum/(315)
DecU2_yield = DecU2_sum/(315)
JanU2_yield = JanU2_sum/(315)

YieldU2 = np.array([MarU2_yield, AprU2_yield, MayU2_yield,
                    JunU2_yield, JulU2_yield, AugU2_yield, SepU2_yield,
                    OctU2_yield, NovU2_yield, DecU2_yield, JanU2_yield])
#-----------------------------------------------------------------------
#Calculating yield for poly east, getting the ouptu power (Wh) Poly East
MarC3 = PolyC3.loc[0:21430,['PowerDC']]
AprC3 = PolyC3.loc[21431:47323,['PowerDC']]
MayC3 = PolyC3.loc[47324:78651,['PowerDC']]
JunC3 = PolyC3.loc[78652:110625,['PowerDC']]
JulC3 = PolyC3.loc[110626:142593,['PowerDC']]
AugC3 = PolyC3.loc[142594:170701,['PowerDC']]
SepC3 = PolyC3.loc[170702:193169,['PowerDC']]
OctC3 = PolyC3.loc[193170:211345,['PowerDC']]
NovC3 = PolyC3.loc[211346:224445,['PowerDC']]
DecC3 = PolyC3.loc[224446:233339,['PowerDC']]
JanC3 = PolyC3.loc[233340:251482,['PowerDC']]

MarC3_sum = np.sum(MarC3)*(1/60)
AprC3_sum = np.sum(AprC3)*(1/60)
MayC3_sum = np.sum(MayC3)*(1/60)
```

```
JunC3_sum = np.sum(JunC3)*(1/60)
JulC3_sum = np.sum(JulC3)*(1/60)
AugC3_sum = np.sum(AugC3)*(1/60)
SepC3_sum = np.sum(SepC3)*(1/60)
OctC3_sum = np.sum(OctC3)*(1/60)
NovC3_sum = np.sum(NovC3)*(1/60)
DecC3_sum = np.sum(DecC3)*(1/60)
JanC3_sum = np.sum(JanC3)*(1/60)

#calculating the Final yield
MarC3_yield = MarC3_sum/(270)
AprC3_yield = AprC3_sum/(270)
MayC3_yield = MayC3_sum/(270)
JunC3_yield = JunC3_sum/(270)
JulC3_yield = JulC3_sum/(270)
AugC3_yield = AugC3_sum/(270)
SepC3_yield = SepC3_sum/(270)
OctC3_yield = OctC3_sum/(270)
NovC3_yield = NovC3_sum/(270)
DecC3_yield = DecC3_sum/(270)
JanC3_yield = JanC3_sum/(270)

YieldC3 = np.array([MarC3_yield, AprC3_yield, MayC3_yield,
                  JunC3_yield, JulC3_yield, AugC3_yield, SepC3_yield,
                  OctC3_yield, NovC3_yield, DecC3_yield, JanC3_yield])
#------------------------------------------------------------------------------
#Calculations for poly west, getting the ouptu power (Wh)
MarD3 = PolyD3.loc[0:21334,['PowerDC']]
AprD3 = PolyD3.loc[21335:46796,['PowerDC']]
MayD3 = PolyD3.loc[46797:77546,['PowerDC']]
JunD3 = PolyD3.loc[77547:108859,['PowerDC']]
JulD3 = PolyD3.loc[108860:140267,['PowerDC']]
AugD3 = PolyD3.loc[140268:167841,['PowerDC']]
SepD3 = PolyD3.loc[167842:189924,['PowerDC']]
OctD3 = PolyD3.loc[189925:207751,['PowerDC']]
NovD3 = PolyD3.loc[207752:220502,['PowerDC']]
DecD3 = PolyD3.loc[220503:229097,['PowerDC']]
JanD3 = PolyD3.loc[229098:241482,['PowerDC']]

MarD3_sum = np.sum(MarD3)*(1/60)
AprD3_sum = np.sum(AprD3)*(1/60)
MayD3_sum = np.sum(MayD3)*(1/60)
JunD3_sum = np.sum(JunD3)*(1/60)
JulD3_sum = np.sum(JulD3)*(1/60)
AugD3_sum = np.sum(AugD3)*(1/60)
SepD3_sum = np.sum(SepD3)*(1/60)
OctD3_sum = np.sum(OctD3)*(1/60)
NovD3_sum = np.sum(NovD3)*(1/60)
DecD3_sum = np.sum(DecD3)*(1/60)
JanD3_sum = np.sum(JanD3)*(1/60)

#calculating the Final yield
MarD3_yield = MarD3_sum/(270)
AprD3_yield = AprD3_sum/(270)
MayD3_yield = MayD3_sum/(270)
JunD3_yield = JunD3_sum/(270)
JulD3_yield = JulD3_sum/(270)
```

```
AugD3_yield = AugD3_sum/(270)
SepD3_yield = SepD3_sum/(270)
OctD3_yield = OctD3_sum/(270)
NovD3_yield = NovD3_sum/(270)
DecD3_yield = DecD3_sum/(270)
JanD3_yield = JanD3_sum/(270)

YieldD3 = np.array([MarD3_yield, AprD3_yield, MayD3_yield,
                    JunD3_yield, JulD3_yield, AugD3_yield, SepD3_yield,
                    OctD3_yield,  NovD3_yield, DecD3_yield, JanD3_yield])
#------------------------------------------------------------------------------
#45deg
Mar = pd.read_excel('Filtered_Mar.xlsx')
Apr = pd.read_excel('Filtered_Apr.xlsx')
May = pd.read_excel('Filtered_May.xlsx')
Jun = pd.read_excel('Filtered_Jun.xlsx')
Jul = pd.read_excel('Filtered_Jul.xlsx')
Aug = pd.read_excel('Filtered_Aug.xlsx')
Sep = pd.read_excel('Filtered_Sep.xlsx')
Oct = pd.read_excel('Filtered_Oct.xlsx')
Nov = pd.read_excel('Filtered_Nov.xlsx')
Dec = pd.read_excel('Filtered_Dec.xlsx')
Jan = pd.read_excel('Filtered_Jan.xlsx')

Mar_itas = pd.read_excel('mar2020itasMinute.xlsx')
Apr_itas = pd.read_excel('apr2020itasMinute.xlsx')
May_itas = pd.read_excel('mai2020itasMinute.xlsx')
Jun_itas = pd.read_excel('jun2020itasMinute.xlsx')
Jul_itas = pd.read_excel('jul2020itasMinute.xlsx')
Aug_itas = pd.read_excel('aug2020itasMinute.xlsx')
Sep_itas = pd.read_excel('sep2020itasMinute.xlsx')
Oct_itas = pd.read_excel('okt2020itasMinute.xlsx')
Nov_itas = pd.read_excel('nov2020itasMinute.xlsx')
Dec_itas = pd.read_excel('des2020itasMinute.xlsx')
Jan_itas = pd.read_excel('jan2021itasMinute.xlsx')

Mar = Mar.dropna()
Apr = Apr.dropna()
May = May.dropna()
Jun = Jun.dropna()
Jul = Jul.dropna()
Aug = Aug.dropna()
Sep = Sep.dropna()
Oct = Oct.dropna()
Nov = Nov.dropna()
Dec = Dec.dropna()
Jan = Jan.dropna()

Mar['DateAndTime '] = pd.to_datetime(Mar['DateAndTime '])
Apr['DateAndTime '] = pd.to_datetime(Apr['DateAndTime '])
May['DateAndTime '] = pd.to_datetime(May['DateAndTime '])
Jun['DateAndTime '] = pd.to_datetime(Jun['DateAndTime '])
Jul['DateAndTime '] = pd.to_datetime(Jul['DateAndTime '])
Aug['DateAndTime '] = pd.to_datetime(Aug['DateAndTime '])
Sep['DateAndTime '] = pd.to_datetime(Sep['DateAndTime '])
Oct['DateAndTime '] = pd.to_datetime(Oct['DateAndTime '])
Nov['DateAndTime '] = pd.to_datetime(Nov['DateAndTime '])
```

```
Dec['DateAndTime '] = pd.to_datetime(Dec['DateAndTime '])
Jan['DateAndTime '] = pd.to_datetime(Jan['DateAndTime '])

Mar_itas['TIMESTAMP'] = pd.to_datetime(Mar_itas['TIMESTAMP'])
Apr_itas['TIMESTAMP'] = pd.to_datetime(Apr_itas['TIMESTAMP'])
May_itas['TIMESTAMP'] = pd.to_datetime(May_itas['TIMESTAMP'])
Jun_itas['TIMESTAMP'] = pd.to_datetime(Jun_itas['TIMESTAMP'])
Jul_itas['TIMESTAMP'] = pd.to_datetime(Jul_itas['TIMESTAMP'])
Aug_itas['TIMESTAMP'] = pd.to_datetime(Aug_itas['TIMESTAMP'])
Sep_itas['TIMESTAMP'] = pd.to_datetime(Sep_itas['TIMESTAMP'])
Oct_itas['TIMESTAMP'] = pd.to_datetime(Oct_itas['TIMESTAMP'])
Nov_itas['TIMESTAMP'] = pd.to_datetime(Nov_itas['TIMESTAMP'])
Dec_itas['TIMESTAMP'] = pd.to_datetime(Dec_itas['TIMESTAMP'])
Jan_itas['TIMESTAMP'] = pd.to_datetime(Jan_itas['TIMESTAMP'])

Mono_mar = Mar.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_apr = Apr.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_may = May.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jun = Jun.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jul = Jul.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_aug = Aug.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_sep = Sep.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_oct = Oct.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_nov = Nov.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_dec = Dec.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jan = Jan.loc[:,['DateAndTime ',' Pmax_M4 ']]

#power for mono (IBC)
Poly_mar = Mar.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_apr = Apr.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_may = May.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jun = Jun.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jul = Jul.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_aug = Aug.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_sep = Sep.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_oct = Oct.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_nov = Nov.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_dec = Dec.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jan = Jan.loc[:,['DateAndTime ',' Pmax_M6']]

#sum of all comuns
Mono_mar_sum = np.sum(Mono_mar)*(1/60)
Mono_apr_sum = np.sum(Mono_apr)*(1/60)
Mono_may_sum = np.sum(Mono_may)*(1/60)
Mono_jun_sum = np.sum(Mono_jun)*(1/60)
Mono_jul_sum = np.sum(Mono_jul)*(1/60)
Mono_aug_sum = np.sum(Mono_aug)*(1/60)
Mono_sep_sum = np.sum(Mono_sep)*(1/60)
Mono_oct_sum = np.sum(Mono_oct)*(1/60)
Mono_nov_sum = np.sum(Mono_nov)*(1/60)
Mono_dec_sum = np.sum(Mono_dec)*(1/60)
Mono_jan_sum = np.sum(Mono_jan)*(1/60)

Poly_mar_sum = np.sum(Poly_mar)*(1/60)
Poly_apr_sum = np.sum(Poly_apr)*(1/60)
Poly_may_sum = np.sum(Poly_may)*(1/60)
Poly_jun_sum = np.sum(Poly_jun)*(1/60)
```

```
Poly_jul_sum = np.sum(Poly_jul)*(1/60)
Poly_aug_sum = np.sum(Poly_aug)*(1/60)
Poly_sep_sum = np.sum(Poly_sep)*(1/60)
Poly_oct_sum = np.sum(Poly_oct)*(1/60)
Poly_nov_sum = np.sum(Poly_nov)*(1/60)
Poly_dec_sum = np.sum(Poly_dec)*(1/60)
Poly_jan_sum = np.sum(Poly_jan)*(1/60)

Mono_mar_yield = Mono_mar_sum[' Pmax_M4 ']/315
Mono_apr_yield = Mono_apr_sum[' Pmax_M4 ']/315
Mono_may_yield = Mono_may_sum[' Pmax_M4 ']/315
Mono_jun_yield = Mono_jun_sum[' Pmax_M4 ']/315
Mono_jul_yield = Mono_jul_sum[' Pmax_M4 ']/315
Mono_aug_yield = Mono_aug_sum[' Pmax_M4 ']/315
Mono_sep_yield = Mono_sep_sum[' Pmax_M4 ']/315
Mono_oct_yield = Mono_oct_sum[' Pmax_M4 ']/315
Mono_nov_yield = Mono_nov_sum[' Pmax_M4 ']/315
Mono_dec_yield = Mono_dec_sum[' Pmax_M4 ']/315
Mono_jan_yield = Mono_jan_sum[' Pmax_M4 ']/315

Poly_mar_yield = Poly_mar_sum[' Pmax_M6']/270
Poly_apr_yield = Poly_apr_sum[' Pmax_M6']/270
Poly_may_yield = Poly_may_sum[' Pmax_M6']/270
Poly_jun_yield = Poly_jun_sum[' Pmax_M6']/270
Poly_jul_yield = Poly_jul_sum[' Pmax_M6']/270
Poly_aug_yield = Poly_aug_sum[' Pmax_M6']/270
Poly_sep_yield = Poly_sep_sum[' Pmax_M6']/270
Poly_oct_yield = Poly_oct_sum[' Pmax_M6']/270
Poly_nov_yield = Poly_nov_sum[' Pmax_M6']/270
Poly_dec_yield = Poly_dec_sum[' Pmax_M6']/270
Poly_jan_yield = Poly_jan_sum[' Pmax_M6']/270

Yield_mono = np.array([Mono_mar_yield, Mono_apr_yield, Mono_may_yield,
                       Mono_jun_yield, Mono_jul_yield, Mono_aug_yield,
                       Mono_sep_yield, Mono_oct_yield, Mono_nov_yield,
                       Mono_dec_yield, Mono_jan_yield])

Yield_poly = np.array([Poly_mar_yield, Poly_apr_yield, Poly_may_yield,
                       Poly_jun_yield, Poly_jul_yield, Poly_aug_yield,
                       Poly_sep_yield, Poly_oct_yield, Poly_nov_yield,
                       Poly_dec_yield, Poly_jan_yield])
#-----------------------------------------------------------------------------
#BIPV
BIPV_mar = Mar.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_apr = Apr.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_may = May.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_jun = Jun.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_jul = Jul.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_aug = Aug.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_sep = Sep.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_oct = Oct.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_nov = Nov.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_dec = Dec.loc[:,['DateAndTime ',' Pmax_M2 ']]
BIPV_jan = Jan.loc[:,['DateAndTime ',' Pmax_M2 ']]

#Collecting form grey BIPV
GREY_mar = Mar.loc[:,['DateAndTime ',' Pmax_M1 ']]
```

```
GREY_apr = Apr.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_may = May.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_jun = Jun.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_jul = Jul.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_aug = Aug.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_sep = Sep.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_oct = Oct.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_nov = Nov.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_dec = Dec.loc[:,['DateAndTime ',' Pmax_M1 ']]
GREY_jan = Jan.loc[:,['DateAndTime ',' Pmax_M1 ']]

#sum of all comuns
BIPV_mar_sum = np.sum(BIPV_mar)*(1/60)
BIPV_apr_sum = np.sum(BIPV_apr)*(1/60)
BIPV_may_sum = np.sum(BIPV_may)*(1/60)
BIPV_jun_sum = np.sum(BIPV_jun)*(1/60)
BIPV_jul_sum = np.sum(BIPV_jul)*(1/60)
BIPV_aug_sum = np.sum(BIPV_aug)*(1/60)
BIPV_sep_sum = np.sum(BIPV_sep)*(1/60)
BIPV_oct_sum = np.sum(BIPV_oct)*(1/60)
BIPV_nov_sum = np.sum(BIPV_nov)*(1/60)
BIPV_dec_sum = np.sum(BIPV_dec)*(1/60)
BIPV_jan_sum = np.sum(BIPV_jan)*(1/60)

GREY_mar_sum = np.sum(GREY_mar)*(1/60)
GREY_apr_sum = np.sum(GREY_apr)*(1/60)
GREY_may_sum = np.sum(GREY_may)*(1/60)
GREY_jun_sum = np.sum(GREY_jun)*(1/60)
GREY_jul_sum = np.sum(GREY_jul)*(1/60)
GREY_aug_sum = np.sum(GREY_aug)*(1/60)
GREY_sep_sum = np.sum(GREY_sep)*(1/60)
GREY_oct_sum = np.sum(GREY_oct)*(1/60)
GREY_nov_sum = np.sum(GREY_nov)*(1/60)
GREY_dec_sum = np.sum(GREY_dec)*(1/60)
GREY_jan_sum = np.sum(GREY_jan)*(1/60)

#Final Yield BIPV normal
BIPV_mar_yield = BIPV_mar_sum[' Pmax_M2 ']/218.1
BIPV_apr_yield = BIPV_apr_sum[' Pmax_M2 ']/218.1
BIPV_may_yield = BIPV_may_sum[' Pmax_M2 ']/218.1
BIPV_jun_yield = BIPV_jun_sum[' Pmax_M2 ']/218.1
BIPV_jul_yield = BIPV_jul_sum[' Pmax_M2 ']/218.1
BIPV_aug_yield = BIPV_aug_sum[' Pmax_M2 ']/218.1
BIPV_sep_yield = BIPV_sep_sum[' Pmax_M2 ']/218.1
BIPV_oct_yield = BIPV_oct_sum[' Pmax_M2 ']/218.1
BIPV_nov_yield = BIPV_nov_sum[' Pmax_M2 ']/218.1
BIPV_dec_yield = BIPV_dec_sum[' Pmax_M2 ']/218.1
BIPV_jan_yield = BIPV_jan_sum[' Pmax_M2 ']/218.1

#Final Yield BIPV grey
GREY_mar_yield = GREY_mar_sum[' Pmax_M1 ']/193
GREY_apr_yield = GREY_apr_sum[' Pmax_M1 ']/193
GREY_may_yield = GREY_may_sum[' Pmax_M1 ']/193
GREY_jun_yield = GREY_jun_sum[' Pmax_M1 ']/193
GREY_jul_yield = GREY_jul_sum[' Pmax_M1 ']/193
GREY_aug_yield = GREY_aug_sum[' Pmax_M1 ']/193
GREY_sep_yield = GREY_sep_sum[' Pmax_M1 ']/193
```

```
GREY_oct_yield = GREY_oct_sum[' Pmax_M1 ']/193
GREY_nov_yield = GREY_nov_sum[' Pmax_M1 ']/193
GREY_dec_yield = GREY_dec_sum[' Pmax_M1 ']/193
GREY_jan_yield = GREY_jan_sum[' Pmax_M1 ']/193

Yield_BIPV = np.array([BIPV_mar_yield, BIPV_apr_yield, BIPV_may_yield,
                       BIPV_jun_yield, BIPV_jul_yield, BIPV_aug_yield,
                       BIPV_sep_yield, BIPV_oct_yield, BIPV_nov_yield,
                       BIPV_dec_yield, BIPV_jan_yield])

Yield_GREY = np.array([GREY_mar_yield, GREY_apr_yield, GREY_may_yield,
                       GREY_jun_yield, GREY_jul_yield, GREY_aug_yield,
                       GREY_sep_yield, GREY_oct_yield, GREY_nov_yield,
                       GREY_dec_yield, GREY_jan_yield])
#------------------------------------------------------------------------------
Months=np.array(['Mar 20','Apr 20','May 20', 'Jun 20', 'Jul 20',
                    'Aug 20', 'Sep 20', 'Oct 20',
                    'Nov 20', 'Dec 20', 'Jan 21'])

plt.plot(Months, YieldM2, marker='o', color='#1874cd', label='10deg mono east')
plt.plot(Months, YieldU2, marker='o', color='#ff7f24', label='10deg mono west')
plt.plot(Months, YieldC3, marker='o', color='grey', label='10deg poly east')
plt.plot(Months, YieldD3,  marker='o', color='#ffa500',label='10deg poly west')
plt.plot(Months, Yield_mono, marker='o', color='red', label='45deg mono')
plt.plot(Months, Yield_poly,  marker='o', color='#228b22', label='45deg poly')
plt.plot(Months, Yield_BIPV, marker='o', color='#ee3a8c', label='BIPV normal')
plt.plot(Months, Yield_GREY,  marker='o', color='k', label='BIPV grey')
plt.ylabel('Spesific Yield [kWh/kWp]', fontsize=18)
plt.yticks(fontsize=16)
plt.grid(axis='y')
plt.xticks(rotation=45, fontsize=16)
plt.legend(fontsize=16)
plt.show()
```

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 16 10:26:23 2021
@author: Sigrid
"""
#PR calculations for 45deg (both) with data filtering
#same code can be used for all PV orientation, only changing the input data
#files.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

Mar = pd.read_excel('Filtered_Mar.xlsx')
Apr = pd.read_excel('Filtered_Apr.xlsx')
May = pd.read_excel('Filtered_May.xlsx')
Jun = pd.read_excel('Filtered_Jun2.xlsx')
Jul = pd.read_excel('Filtered_Jul2.xlsx')
Aug = pd.read_excel('Filtered_Aug.xlsx')
Sep = pd.read_excel('Filtered_Sep.xlsx')
Oct = pd.read_excel('Filtered_Oct.xlsx')
Nov = pd.read_excel('Filtered_Nov.xlsx')
Dec = pd.read_excel('Filtered_Dec.xlsx')
Jan = pd.read_excel('Filtered_Jan.xlsx')
Feb = pd.read_excel('Filtered_Feb.xlsx')
Mar21 = pd.read_excel('Filtered_Mar21.xlsx')

Mar_itas = pd.read_excel('mar2020itasMinute.xlsx')
Apr_itas = pd.read_excel('apr2020itasMinute.xlsx')
May_itas = pd.read_excel('mai2020itasMinute.xlsx')
Jun_itas = pd.read_excel('jun2020itasMinute.xlsx')
Jul_itas = pd.read_excel('jul2020itasMinute.xlsx')
Aug_itas = pd.read_excel('aug2020itasMinute.xlsx')
Sep_itas = pd.read_excel('sep2020itasMinute.xlsx')
Oct_itas = pd.read_excel('okt2020itasMinute.xlsx')
Nov_itas = pd.read_excel('nov2020itasMinute.xlsx')
Dec_itas = pd.read_excel('des2020itasMinute.xlsx')
Jan_itas = pd.read_excel('jan2021itasMinute.xlsx')
Feb_itas = pd.read_excel('feb2021itasMinute.xlsx')
Mar21_itas = pd.read_excel('mar2021itasMinute.xlsx')

Mar = Mar.dropna()
Apr = Apr.dropna()
May = May.dropna()
Jun = Jun.dropna()
Jul = Jul.dropna()
Aug = Aug.dropna()
Sep = Sep.dropna()
Oct = Oct.dropna()
Nov = Nov.dropna()
Dec = Dec.dropna()
Jan = Jan.dropna()
Feb = Feb.dropna()
Mar21 = Mar21.dropna()

Mar['DateAndTime '] = pd.to_datetime(Mar['DateAndTime '])
Apr['DateAndTime '] = pd.to_datetime(Apr['DateAndTime '])
May['DateAndTime '] = pd.to_datetime(May['DateAndTime '])
```

```
Jun['DateAndTime '] = pd.to_datetime(Jun['DateAndTime '])
Jul['DateAndTime '] = pd.to_datetime(Jul['DateAndTime '])
Aug['DateAndTime '] = pd.to_datetime(Aug['DateAndTime '])
Sep['DateAndTime '] = pd.to_datetime(Sep['DateAndTime '])
Oct['DateAndTime '] = pd.to_datetime(Oct['DateAndTime '])
Nov['DateAndTime '] = pd.to_datetime(Nov['DateAndTime '])
Dec['DateAndTime '] = pd.to_datetime(Dec['DateAndTime '])
Jan['DateAndTime '] = pd.to_datetime(Jan['DateAndTime '])
Feb['DateAndTime '] = pd.to_datetime(Feb['DateAndTime '])
Mar21['DateAndTime '] = pd.to_datetime(Mar21['DateAndTime '])

Mar_itas['TIMESTAMP'] = pd.to_datetime(Mar_itas['TIMESTAMP'])
Apr_itas['TIMESTAMP'] = pd.to_datetime(Apr_itas['TIMESTAMP'])
May_itas['TIMESTAMP'] = pd.to_datetime(May_itas['TIMESTAMP'])
Jun_itas['TIMESTAMP'] = pd.to_datetime(Jun_itas['TIMESTAMP'])
Jul_itas['TIMESTAMP'] = pd.to_datetime(Jul_itas['TIMESTAMP'])
Aug_itas['TIMESTAMP'] = pd.to_datetime(Aug_itas['TIMESTAMP'])
Sep_itas['TIMESTAMP'] = pd.to_datetime(Sep_itas['TIMESTAMP'])
Oct_itas['TIMESTAMP'] = pd.to_datetime(Oct_itas['TIMESTAMP'])
Nov_itas['TIMESTAMP'] = pd.to_datetime(Nov_itas['TIMESTAMP'])
Dec_itas['TIMESTAMP'] = pd.to_datetime(Dec_itas['TIMESTAMP'])
Jan_itas['TIMESTAMP'] = pd.to_datetime(Jan_itas['TIMESTAMP'])
Feb_itas['TIMESTAMP'] = pd.to_datetime(Feb_itas['TIMESTAMP'])
Mar21_itas['TIMESTAMP'] = pd.to_datetime(Mar21_itas['TIMESTAMP'])

Mono_mar = Mar.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_apr = Apr.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_may = May.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jun = Jun.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jul = Jul.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_aug = Aug.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_sep = Sep.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_oct = Oct.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_nov = Nov.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_dec = Dec.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_jan = Jan.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_feb = Feb.loc[:,['DateAndTime ',' Pmax_M4 ']]
Mono_mar21 = Mar21.loc[:,['DateAndTime ',' Pmax_M4 ']]

#power for mono (IBC)
Poly_mar = Mar.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_apr = Apr.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_may = May.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jun = Jun.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jul = Jul.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_aug = Aug.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_sep = Sep.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_oct = Oct.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_nov = Nov.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_dec = Dec.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_jan = Jan.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_feb = Feb.loc[:,['DateAndTime ',' Pmax_M6']]
Poly_mar21 = Mar21.loc[:,['DateAndTime ',' Pmax_M6']]

#irradiance from ITAS data
Mar_irr = Mar_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Apr_irr = Apr_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
```

```
May_irr = May_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Jun_irr = Jun_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Jul_irr = Jul_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Aug_irr = Aug_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Sep_irr = Sep_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Oct_irr = Oct_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Nov_irr = Nov_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Dec_irr = Dec_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Jan_irr = Jan_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Feb_irr = Feb_itas.loc[:,['TIMESTAMP','POAO1_Avg']]
Mar21_irr = Mar21_itas.loc[:,['TIMESTAMP','POAO1_Avg']]

#merge power and irradiance into one df
Mar_mono= Mar_irr.merge(Mono_mar, left_on='TIMESTAMP', right_on='DateAndTime ')
Apr_mono= Apr_irr.merge(Mono_apr, left_on='TIMESTAMP', right_on='DateAndTime ')
May_mono= May_irr.merge(Mono_may, left_on='TIMESTAMP', right_on='DateAndTime ')
Jun_mono= Jun_irr.merge(Mono_jun, left_on='TIMESTAMP', right_on='DateAndTime ')
Jul_mono= Jul_irr.merge(Mono_jul, left_on='TIMESTAMP', right_on='DateAndTime ')
Aug_mono= Aug_irr.merge(Mono_aug, left_on='TIMESTAMP', right_on='DateAndTime ')
Sep_mono= Sep_irr.merge(Mono_sep, left_on='TIMESTAMP', right_on='DateAndTime ')
Oct_mono= Oct_irr.merge(Mono_oct, left_on='TIMESTAMP', right_on='DateAndTime ')
Nov_mono= Nov_irr.merge(Mono_nov, left_on='TIMESTAMP', right_on='DateAndTime ')
Dec_mono= Dec_irr.merge(Mono_dec, left_on='TIMESTAMP', right_on='DateAndTime ')
Jan_mono= Jan_irr.merge(Mono_jan, left_on='TIMESTAMP', right_on='DateAndTime ')
Feb_mono= Feb_irr.merge(Mono_feb, left_on='TIMESTAMP', right_on='DateAndTime ')
Mar21_mono = Mar21_irr.merge(Mono_mar21, left_on='TIMESTAMP',
                             right_on='DateAndTime ')

Mar_poly= Mar_irr.merge(Poly_mar, left_on='TIMESTAMP', right_on='DateAndTime ')
Apr_poly= Apr_irr.merge(Poly_apr, left_on='TIMESTAMP', right_on='DateAndTime ')
May_poly= May_irr.merge(Poly_may, left_on='TIMESTAMP', right_on='DateAndTime ')
Jun_poly= Jun_irr.merge(Poly_jun, left_on='TIMESTAMP', right_on='DateAndTime ')
Jul_poly= Jul_irr.merge(Poly_jul, left_on='TIMESTAMP', right_on='DateAndTime ')
Aug_poly= Aug_irr.merge(Poly_aug, left_on='TIMESTAMP', right_on='DateAndTime ')
Sep_poly= Sep_irr.merge(Poly_sep, left_on='TIMESTAMP', right_on='DateAndTime ')
Oct_poly= Oct_irr.merge(Poly_oct, left_on='TIMESTAMP', right_on='DateAndTime ')
Nov_poly= Nov_irr.merge(Poly_nov, left_on='TIMESTAMP', right_on='DateAndTime ')
Dec_poly= Dec_irr.merge(Poly_dec, left_on='TIMESTAMP', right_on='DateAndTime ')
Jan_poly= Jan_irr.merge(Poly_jan, left_on='TIMESTAMP', right_on='DateAndTime ')
Feb_poly= Feb_irr.merge(Poly_feb, left_on='TIMESTAMP', right_on='DateAndTime ')
Mar21_poly = Mar21_irr.merge(Poly_mar21, left_on='TIMESTAMP',
                             right_on='DateAndTime ')
#sum of all comuns
Mono_mar_sum = np.sum(Mar_mono)*(1/60)
Mono_apr_sum = np.sum(Apr_mono)*(1/60)
Mono_may_sum = np.sum(May_mono)*(1/60)
Mono_jun_sum = np.sum(Jun_mono)*(1/60)
Mono_jul_sum = np.sum(Jul_mono)*(1/60)
Mono_aug_sum = np.sum(Aug_mono)*(1/60)
Mono_sep_sum = np.sum(Sep_mono)*(1/60)
Mono_oct_sum = np.sum(Oct_mono)*(1/60)
Mono_nov_sum = np.sum(Nov_mono)*(1/60)
Mono_dec_sum = np.sum(Dec_mono)*(1/60)
Mono_jan_sum = np.sum(Jan_mono)*(1/60)
Mono_feb_sum = np.sum(Feb_mono)*(1/60)
Mono_mar21_sum = np.sum(Mar21_mono)*(1/60)
```

```
Poly_mar_sum = np.sum(Mar_poly)*(1/60)
Poly_apr_sum = np.sum(Apr_poly)*(1/60)
Poly_may_sum = np.sum(May_poly)*(1/60)
Poly_jun_sum = np.sum(Jun_poly)*(1/60)
Poly_jul_sum = np.sum(Jul_poly)*(1/60)
Poly_aug_sum = np.sum(Aug_poly)*(1/60)
Poly_sep_sum = np.sum(Sep_poly)*(1/60)
Poly_oct_sum = np.sum(Oct_poly)*(1/60)
Poly_nov_sum = np.sum(Nov_poly)*(1/60)
Poly_dec_sum = np.sum(Dec_poly)*(1/60)
Poly_jan_sum = np.sum(Jan_poly)*(1/60)
Poly_feb_sum = np.sum(Feb_poly)*(1/60)
Poly_mar21_sum = np.sum(Mar21_poly)*(1/60)

Mono_mar_yield = Mono_mar_sum[' Pmax_M4 ']/315
Mono_apr_yield = Mono_apr_sum[' Pmax_M4 ']/315
Mono_may_yield = Mono_may_sum[' Pmax_M4 ']/315
Mono_jun_yield = Mono_jun_sum[' Pmax_M4 ']/315
Mono_jul_yield = Mono_jul_sum[' Pmax_M4 ']/315
Mono_aug_yield = Mono_aug_sum[' Pmax_M4 ']/315
Mono_sep_yield = Mono_sep_sum[' Pmax_M4 ']/315
Mono_oct_yield = Mono_oct_sum[' Pmax_M4 ']/315
Mono_nov_yield = Mono_nov_sum[' Pmax_M4 ']/315
Mono_dec_yield = Mono_dec_sum[' Pmax_M4 ']/315
Mono_jan_yield = Mono_jan_sum[' Pmax_M4 ']/315
Mono_feb_yield = Mono_feb_sum[' Pmax_M4 ']/315
Mono_mar21_yield = Mono_mar21_sum[' Pmax_M4 ']/315

Poly_mar_yield = Poly_mar_sum[' Pmax_M6']/270
Poly_apr_yield = Poly_apr_sum[' Pmax_M6']/270
Poly_may_yield = Poly_may_sum[' Pmax_M6']/270
Poly_jun_yield = Poly_jun_sum[' Pmax_M6']/270
Poly_jul_yield = Poly_jul_sum[' Pmax_M6']/270
Poly_aug_yield = Poly_aug_sum[' Pmax_M6']/270
Poly_sep_yield = Poly_sep_sum[' Pmax_M6']/270
Poly_oct_yield = Poly_oct_sum[' Pmax_M6']/270
Poly_nov_yield = Poly_nov_sum[' Pmax_M6']/270
Poly_dec_yield = Poly_dec_sum[' Pmax_M6']/270
Poly_jan_yield = Poly_jan_sum[' Pmax_M6']/270
Poly_feb_yield = Poly_feb_sum[' Pmax_M6']/270
Poly_mar21_yield = Poly_mar21_sum[' Pmax_M6']/270

#reference yield
Mar_YR = Poly_mar_sum['POA01_Avg']/1000
Apr_YR = Poly_apr_sum['POA01_Avg']/1000
May_YR = Poly_may_sum['POA01_Avg']/1000
Jun_YR = Poly_jun_sum['POA01_Avg']/1000
Jul_YR = Poly_jul_sum['POA01_Avg']/1000
Aug_YR = Poly_aug_sum['POA01_Avg']/1000
Sep_YR = Poly_sep_sum['POA01_Avg']/1000
Oct_YR = Poly_oct_sum['POA01_Avg']/1000
Nov_YR = Poly_nov_sum['POA01_Avg']/1000
Dec_YR = Poly_dec_sum['POA01_Avg']/1000
Jan_YR = Poly_jan_sum['POA01_Avg']/1000
Feb_YR = Poly_feb_sum['POA01_Avg']/1000
Mar21_YR = Poly_mar21_sum['POA01_Avg']/1000
```

```
#Performance ratio
Mar_PR_mono = Mono_mar_yield/Mar_YR
Apr_PR_mono = Mono_apr_yield/Apr_YR
May_PR_mono = Mono_may_yield/May_YR
Jun_PR_mono = Mono_jun_yield/Jun_YR
Jul_PR_mono = Mono_jul_yield/Jul_YR
Aug_PR_mono = Mono_aug_yield/Aug_YR
Sep_PR_mono = Mono_sep_yield/Sep_YR
Oct_PR_mono = Mono_oct_yield/Oct_YR
Nov_PR_mono = Mono_nov_yield/Nov_YR
Dec_PR_mono = Mono_dec_yield/Dec_YR
Jan_PR_mono = Mono_jan_yield/Jan_YR
Feb_PR_mono = Mono_feb_yield/Feb_YR
Mar21_PR_mono = Mono_mar21_yield/Mar21_YR

PR_mono = np.array([Mar_PR_mono, Apr_PR_mono, May_PR_mono, Jun_PR_mono,
                    Jul_PR_mono, Aug_PR_mono, Sep_PR_mono,
                    Oct_PR_mono, Nov_PR_mono, Dec_PR_mono, Jan_PR_mono,
                    Feb_PR_mono, Mar21_PR_mono])

Mar_PR_poly = Poly_mar_yield/Mar_YR
Apr_PR_poly = Poly_apr_yield/Apr_YR
May_PR_poly = Poly_may_yield/May_YR
Jun_PR_poly = Poly_jun_yield/Jun_YR
Jul_PR_poly = Poly_jul_yield/Jul_YR
Aug_PR_poly = Poly_aug_yield/Aug_YR
Sep_PR_poly = Poly_sep_yield/Sep_YR
Oct_PR_poly = Poly_oct_yield/Oct_YR
Nov_PR_poly = Poly_nov_yield/Nov_YR
Dec_PR_poly = Poly_dec_yield/Dec_YR
Jan_PR_poly = Poly_jan_yield/Jan_YR
Feb_PR_poly = Poly_feb_yield/Feb_YR
Mar21_PR_poly = Poly_mar21_yield/Mar21_YR

PR_poly = np.array([Mar_PR_poly, Apr_PR_poly, May_PR_poly, Jun_PR_poly,
                    Jul_PR_poly, Aug_PR_poly, Sep_PR_poly,
                    Oct_PR_poly, Nov_PR_poly, Dec_PR_poly, Jan_PR_poly,
                    Feb_PR_poly, Mar21_PR_poly])

Months = np.array(['Mar 20', 'Apr 20', 'May 20', 'Jun 20', 'Jul 20', 'Aug 20', 'Sep 20',
                   'Oct 20', 'Nov 20',
                   'Dec 20', 'Jan 21', 'Feb 21', 'Mar 21'])

#-----------------------------------------------------------------------------
plt.plot(Months, PR_mono,'r', marker='o', label='PR 45deg Mono')
plt.plot(Months, PR_poly,'k', marker='o',label='PR 45deg Poly')
plt.ylabel('Performance Ratio', fontsize=18)
plt.yticks(fontsize=18)
plt.grid(axis='y')
plt.ylim(ymax=1.1, ymin=0.2)
plt.xticks(rotation=45, fontsize=16)
plt.legend(fontsize=16, loc='lower left')
plt.show()
#----------------------------------------------------------------------------
```

```python
# -*- coding: utf-8 -*-
"""
Created on Mon Apr  5 13:32:10 2021
@author: Sigrid
"""
#Code to calcultate power-correction for April 19.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import datetime

#collect data
Apr_mono_U2 = pd.read_excel('Data_from_DB_U2_all.xlsx') #west
Apr_mono_V5 = pd.read_excel('Data_from_DB_V5_all.xlsx') #west
Apr_mono_M2 = pd.read_excel('Data_from_DB_M2_all.xlsx') #east
Apr_mono_K2 = pd.read_excel('Data_from_DB_K2_all.xlsx') #east
Apr_poly_D3 = pd.read_excel('Data_from_DB_D3_all.xlsx') #west
Apr_poly_F8 = pd.read_excel('Data_from_DB_F8_all.xlsx') #west
Apr45 = pd.read_excel('Apr_Isc_filtered.xlsx')

Apr_itas = pd.read_excel('Apr_itas.xlsx')

#Datetime
Apr_itas['TIMESTAMP'] = pd.to_datetime(Apr_itas['TIMESTAMP'])
Apr_mono_U2['DATETIME'] = pd.to_datetime(Apr_mono_U2['DATETIME'])
Apr_mono_V5['DATETIME'] = pd.to_datetime(Apr_mono_V5['DATETIME'])
Apr_mono_M2['DATETIME'] = pd.to_datetime(Apr_mono_M2['DATETIME'])
Apr_mono_K2['DATETIME'] = pd.to_datetime(Apr_mono_K2['DATETIME'])
Apr_poly_D3['DATETIME'] = pd.to_datetime(Apr_poly_D3['DATETIME'])
Apr_poly_F8['DATETIME'] = pd.to_datetime(Apr_poly_F8['DATETIME'])
Apr45['DateAndTime '] = pd.to_datetime(Apr45['DateAndTime '])

#Power
PowerU2 = Apr_mono_U2.loc[36521:37408,['DATETIME','PowerDC','CurrentDC']]
PowerV5 = Apr_mono_V5.loc[35977:36865,['DATETIME','PowerDC', 'CurrentDC']]
PowerM2 = Apr_mono_M2.loc[36536:37430,['DATETIME','PowerDC','CurrentDC']]
PowerK2 = Apr_mono_K2.loc[35812:36708,['DATETIME','PowerDC','CurrentDC']]
PowerD3 = Apr_poly_D3.loc[36102:36974,['DATETIME','PowerDC','CurrentDC']]
PowerF8 = Apr_poly_F8.loc[35442:36310,['DATETIME','PowerDC','CurrentDC']]
Power45m = Apr45.loc[:,['DateAndTime ',' Pmax_M4 ',' Isc_M4 ']]
Power45p = Apr45.loc[:,['DateAndTime ',' Pmax_M6',' Isc_M6 ']]
PowerBIPV = Apr45.loc[:,['DateAndTime ',' Pmax_M2 ',' Isc_M2 ']]
PowerGREY = Apr45.loc[:,['DateAndTime ',' Pmax_M1 ',' Isc_M1 ']]

#Module temperature, GPOA,
Apr_U2 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT04_Avg','PoA05_Avg']]
Apr_V5 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT04_Avg','PoA05_Avg']]
Apr_M2 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT03_Avg','POA04_Avg']]
Apr_K2 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT03_Avg','POA04_Avg']]
Apr_D3 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT02_Avg','PoA05_Avg']]
Apr_F8 = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT02_Avg','PoA05_Avg']]
Apr_45m = Apr_itas.loc[15126:16012,['TIMESTAMP','PVT14_Avg', 'POAO1_Avg']]
Apr_45p = Apr_itas.loc[15126:16012,['TIMESTAMP','PVT16_Avg', 'POAO1_Avg']]
Apr_45p = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT16_Avg', 'POAO1_Avg']]
Apr_BIPV = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT09_Avg','PVT10_Avg',
                                     'POAO2_Avg']]
Apr_GREY = Apr_itas.loc[15126:16013,['TIMESTAMP','PVT11_Avg','PVT12_Avg',
                                     'POAO2_Avg']]
#merge to one dataframe
APRIL_U2 = PowerU2.merge(Apr_U2, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_V5 = PowerV5.merge(Apr_V5, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_M2 = PowerM2.merge(Apr_M2, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_K2 = PowerK2.merge(Apr_K2, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_D3 = PowerD3.merge(Apr_D3, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_F8 = PowerF8.merge(Apr_F8, left_on='DATETIME', right_on='TIMESTAMP')
APRIL_45m = Power45m.merge(Apr_45m, left_on='DateAndTime ',
                           right_on='TIMESTAMP')
APRIL_45p = Power45p.merge(Apr_45p, left_on='DateAndTime ',
                           right_on='TIMESTAMP')
APRIL_BIPV = PowerBIPV.merge(Apr_BIPV, left_on='DateAndTime ',
                             right_on='TIMESTAMP')
APRIL_GREY = PowerGREY.merge(Apr_GREY, left_on='DateAndTime ',
                             right_on='TIMESTAMP')
#Time
Time_U2 = APRIL_U2['DATETIME']
Time_V5 = APRIL_V5['DATETIME']
Time_45m = APRIL_45m['DateAndTime ']
Time_45p = APRIL_45p['DateAndTime ']
Time_BIPV = APRIL_BIPV['DateAndTime ']
Time_GREY = APRIL_GREY['DateAndTime ']
Time_M2 = APRIL_M2['DATETIME']
Time_K2 = APRIL_K2['DATETIME']
Time_D3 = APRIL_D3['DATETIME']
Time_F8 = APRIL_F8['DATETIME']

#Pcorr
Tcoeff_mono = -0.38/100
Tcoeff_poly = -0.411/100
Tcoeff_BIPV = -0.405/100

Power_U2 = APRIL_U2['PowerDC']
TempU2 = APRIL_U2['PVT04_Avg']
Pcorr_U2 = APRIL_U2['PowerDC']*(1 - (Tcoeff_mono)*(APRIL_U2['PVT04_Avg'] - 25))

Power_V5 = APRIL_V5['PowerDC']
TempV5 = APRIL_V5['PVT04_Avg']
Pcorr_V5 = APRIL_V5['PowerDC']*(1 - (Tcoeff_mono)*(APRIL_V5['PVT04_Avg'] - 25))

Power_M2 = APRIL_M2['PowerDC']
TempM2 = APRIL_M2['PVT03_Avg']
Pcorr_M2 = APRIL_M2['PowerDC']*(1 - (Tcoeff_mono)*(APRIL_M2['PVT03_Avg'] - 25))

Power_K2 = APRIL_K2['PowerDC']
TempK2 = APRIL_K2['PVT03_Avg']
Pcorr_K2 = APRIL_K2['PowerDC']*(1 - (Tcoeff_mono)*(APRIL_K2['PVT03_Avg'] - 25))

Power_D3 = APRIL_D3['PowerDC']
TempD3 = APRIL_D3['PVT02_Avg']
Pcorr_D3 = APRIL_D3['PowerDC']*(1 - (Tcoeff_poly)*(APRIL_D3['PVT02_Avg'] - 25))

Power_F8 = APRIL_F8['PowerDC']
TempF8 = APRIL_F8['PVT02_Avg']
Pcorr_F8 = APRIL_F8['PowerDC']*(1 - (Tcoeff_poly)*(APRIL_F8['PVT02_Avg'] - 25))
```

```
Power_45m = APRIL_45m[' Pmax_M4 ']
Temp_45m = APRIL_45m['PVT14_Avg']
Pcorr_45m = APRIL_45m[' Pmax_M4 '] * (1 - (Tcoeff_mono) *
                     (APRIL_45m['PVT14_Avg'] - 25))

Power_45p = APRIL_45p[' Pmax_M6']
Temp_45p = APRIL_45p['PVT16_Avg']
Pcorr_45p =  APRIL_45p[' Pmax_M6'] * (1 - (Tcoeff_poly) *
                      (APRIL_45p['PVT16_Avg'] - 25))

Power_BIPV = APRIL_BIPV[' Pmax_M2 ']
Temp_BIPV =( APRIL_BIPV['PVT09_Avg'] + APRIL_BIPV['PVT10_Avg'] ) /2
Pcorr_BIPV =  APRIL_BIPV[' Pmax_M2 '] * (1 - (Tcoeff_BIPV) *
                        (APRIL_BIPV['PVT10_Avg'] - 25))

Power_GREY = APRIL_GREY[' Pmax_M1 ']
Temp_GREY = (APRIL_GREY['PVT11_Avg'] + APRIL_GREY['PVT12_Avg'] ) / 2
Pcorr_GREY = APRIL_GREY[' Pmax_M1 ']*(1 - (Tcoeff_BIPV) *
                       (APRIL_GREY['PVT12_Avg'] - 25))

#Power corr. T + G
Pcorr_TG_U2 = Pcorr_U2 * (1000/APRIL_U2['PoA05_Avg'])
Pcorr_TG_V5 = Pcorr_V5 * (1000/APRIL_V5['PoA05_Avg'])

Pcorr_TG_M2 = Pcorr_M2 * (1000/APRIL_M2['POA04_Avg'])
Pcorr_TG_K2 = Pcorr_K2 * (1000/APRIL_K2['POA04_Avg'])

Pcorr_TG_D3 = Pcorr_D3 * (1000/APRIL_D3['PoA05_Avg'])
Pcorr_TG_F8 = Pcorr_F8 * (1000/APRIL_F8['PoA05_Avg'])

Pcorr_TG_45m = Pcorr_45m * (1000/APRIL_45m['POAO1_Avg'])
Pcorr_TG_45p = Pcorr_45p * (1000/APRIL_45p['POAO1_Avg'])

Pcorr_TG_BIPV = Pcorr_BIPV * (1000/APRIL_BIPV['POAO2_Avg'])
Pcorr_TG_GREY = Pcorr_GREY * (1000/APRIL_GREY['POAO2_Avg'])

#corr spectrum
SF_U2 = (APRIL_U2['CurrentDC']/10.02)/(APRIL_U2['PoA05_Avg']/1000)
Pcorr_TGSF_U2 = Pcorr_TG_U2 / SF_U2

SF_V5 = (APRIL_V5['CurrentDC']/10.02)/(APRIL_V5['PoA05_Avg']/1000)
Pcorr_TGSF_V5 = Pcorr_TG_V5 / SF_V5

SF_M2 = (APRIL_M2['CurrentDC']/10.02)/(APRIL_M2['POA04_Avg']/1000)
Pcorr_TGSF_M2 = Pcorr_TG_M2 / SF_M2

SF_K2 = (APRIL_K2['CurrentDC']/10.02)/(APRIL_K2['POA04_Avg']/1000)
Pcorr_TGSF_K2 = Pcorr_TG_K2 / SF_K2

SF_D3 = (APRIL_D3['CurrentDC']/9.08)/(APRIL_D3['PoA05_Avg']/1000)
Pcorr_TGSF_D3 = Pcorr_TG_D3 / SF_D3

SF_F8 = (APRIL_F8['CurrentDC']/9.08)/(APRIL_F8['PoA05_Avg']/1000)
Pcorr_TGSF_F8 = Pcorr_TG_F8 / SF_F8

SF_45m = (APRIL_45m[' Isc_M4 ']/10.02)/(APRIL_45m['POAO1_Avg']/1000)
Pcorr_TGSF_45m = Pcorr_TG_45m / SF_45m
```

```
SF_45p = (APRIL_45p[' Isc_M6 ']/9.23)/(APRIL_45p['POAO1_Avg']/1000)
Pcorr_TGSF_45p = Pcorr_TG_45p / SF_45p

SF_BIPV = (APRIL_BIPV[' Isc_M2 ']/8.26)/(APRIL_BIPV['POAO2_Avg']/1000)
Pcorr_TGSF_BIPV = Pcorr_TG_BIPV / SF_BIPV

SF_GREY = (APRIL_GREY[' Isc_M1 ']/7.31)/(APRIL_GREY['POAO2_Avg']/1000)
Pcorr_TGSF_GREY = Pcorr_TG_GREY / SF_GREY
#------------------------------------------------------------------------------
#10 mono U2
fig, (ax1, ax6) = plt.subplots(1,2)

color = 'tab:red'
ax1.set_xlabel('10deg mono west (U2)', fontsize=18)
ax1.set_ylabel('Temperature', color=color, fontsize=18)
ax1.plot(Time_U2, TempU2, color=color, label='Temperature')
ax1.set_ylim(ymax=35)
ax1.tick_params(axis='y', labelcolor=color, labelsize=18)
ax1.tick_params(axis='x', labelsize=14, rotation=35)

ax2 = ax1.twinx()  # instantiate a second axes that shares the same x-axis
color = 'tab:blue'
ax2.set_ylabel('Power (W)', color=color, fontsize=18)
ax2.plot(Time_U2, Power_U2, color=color, label='Power')
ax2.set_ylim (ymax=400, ymin=-10)
ax2.tick_params(axis='y', labelcolor=color, labelsize=18)

ax3 = ax2
color = 'tab:green'
ax3.plot(Time_U2, Pcorr_U2, color=color, label='Pcorr T')

ax4 = ax2
color = 'k'
ax4.plot(Time_U2, Pcorr_TG_U2, color=color, label='Pcorr T+G')

ax5 = ax2
color='orange'
ax5.plot(Time_U2, Pcorr_TGSF_U2, color=color, label='Pcorr T+G+SF')

plt.grid(axis='y')
plt.show()
#-----------------------------------------------------------------------------
#10 mono V5
color = 'tab:red'
ax6.set_xlabel('10deg mono west (V5)', fontsize=18)
ax6.set_ylabel('Temperature', color=color, fontsize=18)
ax6.plot(Time_V5, TempV5, color=color)
ax6.set_ylim(ymax=35)
ax6.tick_params(axis='y', labelcolor=color, labelsize=18)
ax6.tick_params(axis='x', labelsize=14, rotation=35)

ax7 = ax6.twinx()  # instantiate a second axes that shares the same x-axis
color = 'tab:blue'
ax7.set_ylabel('Power (W)', color=color, fontsize=18)
ax7.plot(Time_V5, Power_V5, color=color)
ax7.set_ylim (ymax=400, ymin=-10)
```

```
ax7.tick_params(axis='y', labelcolor=color, labelsize=18)

ax8 = ax7
color = 'tab:green'
ax8.plot(Time_V5, Pcorr_V5, color=color)

ax9 = ax7
color = 'k'
ax9.plot(Time_V5, Pcorr_TG_V5, color=color)

ax10 = ax7
color='orange'
ax10.plot(Time_V5, Pcorr_TGSF_V5, color=color)

plt.grid(axis='y')
fig.tight_layout(w_pad=-7)  # otherwise the right y-label is slightly clipped
fig.legend(fontsize=16)
plt.show()
#------------------------------------------------------------------------------
#same plt code can be used for all plots by changing the parameter on the
#x- and y- axis.
```

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Apr 28 12:57:58 2021
@author: Sigrid
"""
#code to make linear regression and predict output power for missing values in
#45deg and BIPV.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

Mar_df = pd.read_excel('Filtered_Mar.xlsx')
Apr_df = pd.read_excel('Filtered_Apr.xlsx')
May_df = pd.read_excel('Filtered_May.xlsx')
Jun_df = pd.read_excel('Filtered_Jun.xlsx')
Jul_df = pd.read_excel('Filtered_Jul.xlsx')
Aug_df = pd.read_excel('Filtered_Aug.xlsx')
Sep_df = pd.read_excel('Filtered_Sep.xlsx')
Oct_df = pd.read_excel('Filtered_Oct.xlsx')
Nov_df = pd.read_excel('Filtered_Nov.xlsx')
Dec_df = pd.read_excel('Filtered_Dec.xlsx')
Jan_df = pd.read_excel('Filtered_Jan.xlsx')
Feb_df = pd.read_excel('Filtered_Feb.xlsx')
Mar21_df = pd.read_excel('Filtered_Mar21.xlsx')

Mar_itas = pd.read_excel('Mar_mean2.xlsx')
Apr_itas = pd.read_excel('Apr_mean2.xlsx')
May_itas = pd.read_excel('May_mean2.xlsx')
Jun_itas = pd.read_excel('Jun_mean2.xlsx')
Jul_itas = pd.read_excel('Jul_mean2.xlsx')
Aug_itas = pd.read_excel('Aug_mean2.xlsx')
Sep_itas = pd.read_excel('Sep_mean2.xlsx')
Oct_itas = pd.read_excel('Oct_mean2.xlsx')
Nov_itas = pd.read_excel('Nov_mean2.xlsx')
Dec_itas = pd.read_excel('Dec_mean2.xlsx')
Jan_itas = pd.read_excel('Jan_mean3.xlsx')
Feb_itas = pd.read_excel('Feb_mean2.xlsx')
Mar21_itas = pd.read_excel('Mar21_mean2.xlsx')

Mar_itas = Mar_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Apr_itas = Apr_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
May_itas = May_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Jun_itas = Jun_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Jul_itas = Jul_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Aug_itas = Aug_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Sep_itas = Sep_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Oct_itas = Oct_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Nov_itas = Nov_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Dec_itas = Dec_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Jan_itas = Jan_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Feb_itas = Feb_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                           'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]
Ma21_itas = Mar21_itas.loc[:,['TIMESTAMP','POAO1_Avg','POAO2_Avg','PVT14_Avg',
                              'PVT16_Avg', 'Temp BIPV', 'Temp GREY']]

Mar_itas['TIMESTAMP'] = pd.to_datetime(Mar_itas['TIMESTAMP'])
Apr_itas['TIMESTAMP'] = pd.to_datetime(Apr_itas['TIMESTAMP'])
May_itas['TIMESTAMP'] = pd.to_datetime(May_itas['TIMESTAMP'])
Jun_itas['TIMESTAMP'] = pd.to_datetime(Jun_itas['TIMESTAMP'])
Jul_itas['TIMESTAMP'] = pd.to_datetime(Jul_itas['TIMESTAMP'])
Aug_itas['TIMESTAMP'] = pd.to_datetime(Aug_itas['TIMESTAMP'])
Sep_itas['TIMESTAMP'] = pd.to_datetime(Sep_itas['TIMESTAMP'])
Oct_itas['TIMESTAMP'] = pd.to_datetime(Oct_itas['TIMESTAMP'])
Nov_itas['TIMESTAMP'] = pd.to_datetime(Nov_itas['TIMESTAMP'])
Dec_itas['TIMESTAMP'] = pd.to_datetime(Dec_itas['TIMESTAMP'])
Jan_itas['TIMESTAMP'] = pd.to_datetime(Jan_itas['TIMESTAMP'])
Feb_itas['TIMESTAMP'] = pd.to_datetime(Feb_itas['TIMESTAMP'])
Mar21_itas['TIMESTAMP'] = pd.to_datetime(Mar21_itas['TIMESTAMP'])

Mar_df['DateAndTime '] = pd.to_datetime(Mar_df['DateAndTime '])
Apr_df['DateAndTime '] = pd.to_datetime(Apr_df['DateAndTime '])
May_df['DateAndTime '] = pd.to_datetime(May_df['DateAndTime '])
Jun_df['DateAndTime '] = pd.to_datetime(Jun_df['DateAndTime '])
Jul_df['DateAndTime '] = pd.to_datetime(Jul_df['DateAndTime '])
Aug_df['DateAndTime '] = pd.to_datetime(Aug_df['DateAndTime '])
Sep_df['DateAndTime '] = pd.to_datetime(Sep_df['DateAndTime '])
Oct_df['DateAndTime '] = pd.to_datetime(Oct_df['DateAndTime '])
Nov_df['DateAndTime '] = pd.to_datetime(Nov_df['DateAndTime '])
Dec_df['DateAndTime '] = pd.to_datetime(Dec_df['DateAndTime '])
Jan_df['DateAndTime '] = pd.to_datetime(Jan_df['DateAndTime '])
Feb_df['DateAndTime '] = pd.to_datetime(Feb_df['DateAndTime '])
Mar21_df['DateAndTime '] = pd.to_datetime(Mar21_df['DateAndTime '])

Mar = Mar_df.merge(Mar_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Apr = Apr_df.merge(Apr_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
May = May_df.merge(May_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Jun = Jun_df.merge(Jun_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Jul = Jul_df.merge(Jul_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Aug = Aug_df.merge(Aug_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Sep = Sep_df.merge(Sep_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Oct = Oct_df.merge(Oct_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Nov = Nov_df.merge(Nov_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Dec = Dec_df.merge(Dec_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Jan = Jan_df.merge(Jan_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Feb = Feb_df.merge(Feb_itas, left_on='DateAndTime ', right_on='TIMESTAMP')
Mar21= Mar21_df.merge(Mar21_itas, left_on='DateAndTime ', right_on='TIMESTAMP')

Mar_drop = Mar.dropna()
Apr_drop = Apr.dropna()
May_drop = May.dropna()
Jun_drop = Jun.dropna()
Jul_drop = Jul.dropna()
```

```
Aug_drop = Aug.dropna()
Sep_drop = Sep.dropna()
Oct_drop = Oct.dropna()
Nov_drop = Nov.dropna()
Dec_drop = Dec.dropna()
Jan_drop = Jan.dropna()
Feb_drop = Feb.dropna()
Mar21_drop = Mar21.dropna()

#temperature coefficients
Tcoeff_mono = -0.38/100
Tcoeff_poly = -0.411/100
Tcoeff_B = -0.405/100
Tcoeff_G = -0.405/100

#temperature corrected power, 45deg mono M4
Pcorr_mar =Mar_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Mar_drop['PVT14_Avg']-25))
Pcorr_apr =Apr_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Apr_drop['PVT14_Avg']-25))
Pcorr_may =May_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (May_drop['PVT14_Avg']-25))
Pcorr_jun =Jun_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Jun_drop['PVT14_Avg']-25))
Pcorr_jul =Jul_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Jul_drop['PVT14_Avg']-25))
Pcorr_aug =Aug_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Aug_drop['PVT14_Avg']-25))
Pcorr_sep =Sep_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Sep_drop['PVT14_Avg']-25))
Pcorr_oct =Oct_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Oct_drop['PVT14_Avg']-25))
Pcorr_nov =Nov_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Nov_drop['PVT14_Avg']-25))
Pcorr_dec =Dec_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Dec_drop['PVT14_Avg']-25))
Pcorr_jan =Jan_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Jan_drop['PVT14_Avg']-25))
Pcorr_feb =Feb_drop[' Pmax_M4 ']/(1+(Tcoeff_mono) * (Feb_drop['PVT14_Avg']-25))
Pcorr_mar21 = Mar21_drop[' Pmax_M4 '] / (1 + (Tcoeff_mono) *
                        (Mar21_drop['PVT14_Avg'] - 25))
#45deg poly M6
PcorrM6_mar = Mar_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Mar_drop['PVT14_Avg']-25))
PcorrM6_apr = Apr_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Apr_drop['PVT14_Avg']-25))
PcorrM6_may = May_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(May_drop['PVT14_Avg']-25))
PcorrM6_jun = Jun_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Jun_drop['PVT14_Avg']-25))
PcorrM6_jul = Jul_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Jul_drop['PVT14_Avg']-25))
PcorrM6_aug = Aug_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Aug_drop['PVT14_Avg']-25))
PcorrM6_sep = Sep_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Sep_drop['PVT14_Avg']-25))
PcorrM6_oct = Oct_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Oct_drop['PVT14_Avg']-25))
PcorrM6_nov = Nov_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Nov_drop['PVT14_Avg']-25))
PcorrM6_dec = Dec_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Dec_drop['PVT14_Avg']-25))
PcorrM6_jan = Jan_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Jan_drop['PVT14_Avg']-25))
PcorrM6_feb = Feb_drop[' Pmax_M6']/(1+(Tcoeff_poly)*(Feb_drop['PVT14_Avg']-25))
PcorrM6_mar21 = Mar21_drop[' Pmax_M6'] / (1 + (Tcoeff_poly) *
                          (Mar21_drop['PVT14_Avg'] - 25))
#BIPV
Pcorr_mar_BIPV =Mar_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Mar_drop['Temp BIPV']-25))
Pcorr_apr_BIPV =Apr_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Apr_drop['Temp BIPV']-25))
Pcorr_may_BIPV =May_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(May_drop['Temp BIPV']-25))
Pcorr_jun_BIPV =Jun_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Jun_drop['Temp BIPV']-25))
Pcorr_jul_BIPV =Jul_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Jul_drop['Temp BIPV']-25))
Pcorr_aug_BIPV =Aug_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Aug_drop['Temp BIPV']-25))
Pcorr_sep_BIPV =Sep_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Sep_drop['Temp BIPV']-25))
Pcorr_oct_BIPV =Oct_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Oct_drop['Temp BIPV']-25))
Pcorr_nov_BIPV =Nov_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Nov_drop['Temp BIPV']-25))
Pcorr_dec_BIPV =Dec_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Dec_drop['Temp BIPV']-25))
Pcorr_jan_BIPV =Jan_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Jan_drop['Temp BIPV']-25))
```

```
Pcorr_feb_BIPV =Feb_drop[' Pmax_M2 ']/(1+(Tcoeff_B)*(Feb_drop['Temp BIPV']-25))
Pcorr_mar21_BIPV = Mar21_drop[' Pmax_M2 '] / (1 + (Tcoeff_B) *
                             (Mar21_drop['Temp BIPV'] - 25))
#BIPV GREY
Pcorr_mar_GREY =Mar_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Mar_drop['Temp GREY']-25))
Pcorr_apr_GREY =Apr_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Apr_drop['Temp GREY']-25))
Pcorr_may_GREY =May_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(May_drop['Temp GREY']-25))
Pcorr_jun_GREY =Jun_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Jun_drop['Temp GREY']-25))
Pcorr_jul_GREY =Jul_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Jul_drop['Temp GREY']-25))
Pcorr_aug_GREY =Aug_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Aug_drop['Temp GREY']-25))
Pcorr_sep_GREY =Sep_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Sep_drop['Temp GREY']-25))
Pcorr_oct_GREY =Oct_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Oct_drop['Temp GREY']-25))
Pcorr_nov_GREY =Nov_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Nov_drop['Temp GREY']-25))
Pcorr_dec_GREY =Dec_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Dec_drop['Temp GREY']-25))
Pcorr_jan_GREY =Jan_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Jan_drop['Temp GREY']-25))
Pcorr_feb_GREY =Feb_drop[' Pmax_M1 ']/(1+(Tcoeff_G)*(Feb_drop['Temp GREY']-25))
Pcorr_mar21_GREY = Mar21_drop[' Pmax_M1 '] / (1 + (Tcoeff_G) *
                             (Mar21_drop['Temp GREY'] - 25))

Mar_irrM4 = Mar_drop.loc[:,['POAO1_Avg']]
Apr_irrM4 = Apr_drop.loc[:,['POAO1_Avg']]
May_irrM4 = May_drop.loc[:,['POAO1_Avg']]
Jun_irrM4 = Jun_drop.loc[:,['POAO1_Avg']]
Jul_irrM4 = Jul_drop.loc[:,['POAO1_Avg']]
Aug_irrM4 = Aug_drop.loc[:,['POAO1_Avg']]
Sep_irrM4 = Sep_drop.loc[:,['POAO1_Avg']]
Oct_irrM4 = Oct_drop.loc[:,['POAO1_Avg']]
Nov_irrM4 = Nov_drop.loc[:,['POAO1_Avg']]
Dec_irrM4 = Dec_drop.loc[:,['POAO1_Avg']]
Jan_irrM4 = Jan_drop.loc[:,['POAO1_Avg']]
Feb_irrM4 = Feb_drop.loc[:,['POAO1_Avg']]
Mar21_irrM4 = Mar21_drop.loc[:,['POAO1_Avg']]

Mar_irr = Mar_drop.loc[:,['POAO2_Avg']]
Apr_irr = Apr_drop.loc[:,['POAO2_Avg']]
May_irr = May_drop.loc[:,['POAO2_Avg']]
Jun_irr = Jun_drop.loc[:,['POAO2_Avg']]
Jul_irr = Jul_drop.loc[:,['POAO2_Avg']]
Aug_irr = Aug_drop.loc[:,['POAO2_Avg']]
Sep_irr = Sep_drop.loc[:,['POAO2_Avg']]
Oct_irr = Oct_drop.loc[:,['POAO2_Avg']]
Nov_irr = Nov_drop.loc[:,['POAO2_Avg']]
Dec_irr = Dec_drop.loc[:,['POAO2_Avg']]
Jan_irr = Jan_drop.loc[:,['POAO2_Avg']]
Feb_irr = Feb_drop.loc[:,['POAO2_Avg']]
Mar21_irr = Mar21_drop.loc[:,['POAO2_Avg']]


#Linear regression
#45deg Mono M4
LR_mar = LinearRegression().fit(Mar_irrM4, Pcorr_mar)
LR_apr = LinearRegression().fit(Apr_irrM4, Pcorr_apr)
LR_may = LinearRegression().fit(May_irrM4, Pcorr_may)
LR_jun = LinearRegression().fit(Jun_irrM4, Pcorr_jun)
LR_jul = LinearRegression().fit(Jul_irrM4, Pcorr_jul)
LR_aug = LinearRegression().fit(Aug_irrM4, Pcorr_aug)
LR_sep = LinearRegression().fit(Sep_irrM4, Pcorr_sep)
```

```
LR_oct = LinearRegression().fit(Oct_irrM4, Pcorr_oct)
LR_nov = LinearRegression().fit(Nov_irrM4, Pcorr_nov)
LR_dec = LinearRegression().fit(Dec_irrM4, Pcorr_dec)
LR_jan = LinearRegression().fit(Jan_irrM4, Pcorr_jan)
LR_feb = LinearRegression().fit(Feb_irrM4, Pcorr_feb)
LR_mar21 = LinearRegression().fit(Mar21_irrM4, Pcorr_mar21)
#45deg poly M6
LR_marM6 = LinearRegression().fit(Mar_irrM4, PcorrM6_mar)
LR_aprM6 = LinearRegression().fit(Apr_irrM4, PcorrM6_apr)
LR_mayM6 = LinearRegression().fit(May_irrM4, PcorrM6_may)
LR_junM6 = LinearRegression().fit(Jun_irrM4, PcorrM6_jun)
LR_julM6 = LinearRegression().fit(Jul_irrM4, PcorrM6_jul)
LR_augM6 = LinearRegression().fit(Aug_irrM4, PcorrM6_aug)
LR_sepM6 = LinearRegression().fit(Sep_irrM4, PcorrM6_sep)
LR_octM6 = LinearRegression().fit(Oct_irrM4, PcorrM6_oct)
LR_novM6 = LinearRegression().fit(Nov_irrM4, PcorrM6_nov)
LR_decM6 = LinearRegression().fit(Dec_irrM4, PcorrM6_dec)
LR_janM6 = LinearRegression().fit(Jan_irrM4, PcorrM6_jan)
LR_febM6 = LinearRegression().fit(Feb_irrM4, PcorrM6_feb)
LR_mar21M6 = LinearRegression().fit(Mar21_irrM4, PcorrM6_mar21)
#BIPV
LR_mar_BIPV = LinearRegression().fit(Mar_irr, Pcorr_mar_BIPV)
LR_apr_BIPV = LinearRegression().fit(Apr_irr, Pcorr_apr_BIPV)
LR_may_BIPV = LinearRegression().fit(May_irr, Pcorr_may_BIPV)
LR_jun_BIPV = LinearRegression().fit(Jun_irr, Pcorr_jun_BIPV)
LR_jul_BIPV = LinearRegression().fit(Jul_irr, Pcorr_jul_BIPV)
LR_aug_BIPV = LinearRegression().fit(Aug_irr, Pcorr_aug_BIPV)
LR_sep_BIPV = LinearRegression().fit(Sep_irr, Pcorr_sep_BIPV)
LR_oct_BIPV = LinearRegression().fit(Oct_irr, Pcorr_oct_BIPV)
LR_nov_BIPV = LinearRegression().fit(Nov_irr, Pcorr_nov_BIPV)
LR_dec_BIPV = LinearRegression().fit(Dec_irr, Pcorr_dec_BIPV)
LR_jan_BIPV = LinearRegression().fit(Jan_irr, Pcorr_jan_BIPV)
LR_feb_BIPV = LinearRegression().fit(Feb_irr, Pcorr_feb_BIPV)
LR_mar21_BIPV = LinearRegression().fit(Mar21_irr, Pcorr_mar21_BIPV)
#BIPV GREY
LR_mar_GREY = LinearRegression().fit(Mar_irr, Pcorr_mar_GREY)
LR_apr_GREY = LinearRegression().fit(Apr_irr, Pcorr_apr_GREY)
LR_may_GREY = LinearRegression().fit(May_irr, Pcorr_may_GREY)
LR_jun_GREY = LinearRegression().fit(Jun_irr, Pcorr_jun_GREY)
LR_jul_GREY = LinearRegression().fit(Jul_irr, Pcorr_jul_GREY)
LR_aug_GREY = LinearRegression().fit(Aug_irr, Pcorr_aug_GREY)
LR_sep_GREY = LinearRegression().fit(Sep_irr, Pcorr_sep_GREY)
LR_oct_GREY = LinearRegression().fit(Oct_irr, Pcorr_oct_GREY)
LR_nov_GREY = LinearRegression().fit(Nov_irr, Pcorr_nov_GREY)
LR_dec_GREY = LinearRegression().fit(Dec_irr, Pcorr_dec_GREY)
LR_jan_GREY = LinearRegression().fit(Jan_irr, Pcorr_jan_GREY)
LR_feb_GREY = LinearRegression().fit(Feb_irr, Pcorr_feb_GREY)
LR_mar21_GREY = LinearRegression().fit(Mar21_irr, Pcorr_mar21_GREY)
#45deg mono M4
Y_pred_mar = LR_mar.predict(Mar_irrM4)
Y_pred_apr = LR_apr.predict(Apr_irrM4)
Y_pred_may = LR_may.predict(May_irrM4)
Y_pred_jun = LR_jun.predict(Jun_irrM4)
Y_pred_jul = LR_jul.predict(Jul_irrM4)
Y_pred_aug = LR_aug.predict(Aug_irrM4)
Y_pred_sep = LR_sep.predict(Sep_irrM4)
Y_pred_oct = LR_oct.predict(Oct_irrM4)
```

```
Y_pred_nov = LR_nov.predict(Nov_irrM4)
Y_pred_dec = LR_dec.predict(Dec_irrM4)
Y_pred_jan = LR_jan.predict(Jan_irrM4)
Y_pred_feb = LR_feb.predict(Feb_irrM4)
Y_pred_mar21 = LR_mar21.predict(Mar21_irrM4)
#45deg poly M6
Y_pred_marM6 = LR_marM6.predict(Mar_irrM4)
Y_pred_aprM6 = LR_aprM6.predict(Apr_irrM4)
Y_pred_mayM6 = LR_mayM6.predict(May_irrM4)
Y_pred_junM6 = LR_junM6.predict(Jun_irrM4)
Y_pred_julM6 = LR_julM6.predict(Jul_irrM4)
Y_pred_augM6 = LR_augM6.predict(Aug_irrM4)
Y_pred_sepM6 = LR_sepM6.predict(Sep_irrM4)
Y_pred_octM6 = LR_octM6.predict(Oct_irrM4)
Y_pred_novM6 = LR_novM6.predict(Nov_irrM4)
Y_pred_decM6 = LR_decM6.predict(Dec_irrM4)
Y_pred_janM6 = LR_janM6.predict(Jan_irrM4)
Y_pred_febM6 = LR_febM6.predict(Feb_irrM4)
Y_pred_mar21M6 = LR_mar21M6.predict(Mar21_irrM4)
#BIPV
Y_pred_mar_BIPV = LR_mar_BIPV.predict(Mar_irr)
Y_pred_apr_BIPV = LR_apr_BIPV.predict(Apr_irr)
Y_pred_may_BIPV = LR_may_BIPV.predict(May_irr)
Y_pred_jun_BIPV = LR_jun_BIPV.predict(Jun_irr)
Y_pred_jul_BIPV = LR_jul_BIPV.predict(Jul_irr)
Y_pred_aug_BIPV = LR_aug_BIPV.predict(Aug_irr)
Y_pred_sep_BIPV = LR_sep_BIPV.predict(Sep_irr)
Y_pred_oct_BIPV = LR_oct_BIPV.predict(Oct_irr)
Y_pred_nov_BIPV = LR_nov_BIPV.predict(Nov_irr)
Y_pred_dec_BIPV = LR_dec_BIPV.predict(Dec_irr)
Y_pred_jan_BIPV = LR_jan_BIPV.predict(Jan_irr)
Y_pred_feb_BIPV = LR_feb_BIPV.predict(Feb_irr)
Y_pred_mar21_BIPV = LR_mar21_BIPV.predict(Mar21_irr)

#BIPV GREY
Y_pred_mar_GREY = LR_mar_GREY.predict(Mar_irr)
Y_pred_apr_GREY = LR_apr_GREY.predict(Apr_irr)
Y_pred_may_GREY = LR_may_GREY.predict(May_irr)
Y_pred_jun_GREY = LR_jun_GREY.predict(Jun_irr)
Y_pred_jul_GREY = LR_jul_GREY.predict(Jul_irr)
Y_pred_aug_GREY = LR_aug_GREY.predict(Aug_irr)
Y_pred_sep_GREY = LR_sep_GREY.predict(Sep_irr)
Y_pred_oct_GREY = LR_oct_GREY.predict(Oct_irr)
Y_pred_nov_GREY = LR_nov_GREY.predict(Nov_irr)
Y_pred_dec_GREY = LR_dec_GREY.predict(Dec_irr)
Y_pred_jan_GREY = LR_jan_GREY.predict(Jan_irr)
Y_pred_feb_GREY = LR_feb_GREY.predict(Feb_irr)
Y_pred_mar21_GREY = LR_mar21_GREY.predict(Mar21_irr)

#45deg mono M4
LR_mar_coeff = LR_mar.coef_
LR_apr_coeff = LR_apr.coef_
LR_may_coeff = LR_may.coef_
LR_jun_coeff = LR_jun.coef_
LR_jul_coeff = LR_jul.coef_
LR_aug_coeff = LR_aug.coef_
LR_sep_coeff = LR_sep.coef_
```

```
LR_oct_coeff = LR_oct.coef_
LR_nov_coeff = LR_nov.coef_
LR_dec_coeff = LR_dec.coef_
LR_jan_coeff = LR_jan.coef_
LR_feb_coeff = LR_feb.coef_
LR_mar21_coeff = LR_mar21.coef_

#45deg poly M6
LR_marM6_coeff = LR_marM6.coef_
LR_aprM6_coeff = LR_aprM6.coef_
LR_mayM6_coeff = LR_mayM6.coef_
LR_junM6_coeff = LR_junM6.coef_
LR_julM6_coeff = LR_julM6.coef_
LR_augM6_coeff = LR_augM6.coef_
LR_sepM6_coeff = LR_sepM6.coef_
LR_octM6_coeff = LR_octM6.coef_
LR_novM6_coeff = LR_novM6.coef_
LR_decM6_coeff = LR_decM6.coef_
LR_janM6_coeff = LR_janM6.coef_
LR_febM6_coeff = LR_febM6.coef_
LR_mar21M6_coeff = LR_mar21M6.coef_

#BIPV
LR_mar_BIPV_coeff = LR_mar_BIPV.coef_
LR_apr_BIPV_coeff = LR_apr_BIPV.coef_
LR_may_BIPV_coeff = LR_may_BIPV.coef_
LR_jun_BIPV_coeff = LR_jun_BIPV.coef_
LR_jul_BIPV_coeff = LR_jul_BIPV.coef_
LR_aug_BIPV_coeff = LR_aug_BIPV.coef_
LR_sep_BIPV_coeff = LR_sep_BIPV.coef_
LR_oct_BIPV_coeff = LR_oct_BIPV.coef_
LR_nov_BIPV_coeff = LR_nov_BIPV.coef_
LR_dec_BIPV_coeff = LR_dec_BIPV.coef_
LR_jan_BIPV_coeff = LR_jan_BIPV.coef_
LR_feb_BIPV_coeff = LR_feb_BIPV.coef_
LR_mar21_BIPV_coeff = LR_mar21_BIPV.coef_

#BIPV GREY
LR_mar_GREY_coeff = LR_mar_GREY.coef_
LR_apr_GREY_coeff = LR_apr_GREY.coef_
LR_may_GREY_coeff = LR_may_GREY.coef_
LR_jun_GREY_coeff = LR_jun_GREY.coef_
LR_jul_GREY_coeff = LR_jul_GREY.coef_
LR_aug_GREY_coeff = LR_aug_GREY.coef_
LR_sep_GREY_coeff = LR_sep_GREY.coef_
LR_oct_GREY_coeff = LR_oct_GREY.coef_
LR_nov_GREY_coeff = LR_nov_GREY.coef_
LR_dec_GREY_coeff = LR_dec_GREY.coef_
LR_jan_GREY_coeff = LR_jan_GREY.coef_
LR_feb_GREY_coeff = LR_feb_GREY.coef_
LR_mar21_GREY_coeff = LR_mar21_GREY.coef_

#45deg mono M4
Pred_mar = pd.DataFrame(LR_mar_coeff * Mar['POA01_Avg'])
Pred_apr = pd.DataFrame(LR_apr_coeff * Apr['POA01_Avg'])
Pred_may = pd.DataFrame(LR_may_coeff * May['POA01_Avg'])
Pred_jun = pd.DataFrame(LR_jun_coeff * Jun['POA01_Avg'])
```

```
Pred_jul = pd.DataFrame(LR_jul_coeff * Jul['POA01_Avg'])
Pred_aug = pd.DataFrame(LR_aug_coeff * Aug['POA01_Avg'])
Pred_sep = pd.DataFrame(LR_sep_coeff * Sep['POA01_Avg'])
Pred_oct = pd.DataFrame(LR_oct_coeff * Oct['POA01_Avg'])
Pred_nov = pd.DataFrame(LR_nov_coeff * Nov['POA01_Avg'])
Pred_dec = pd.DataFrame(LR_dec_coeff * Dec['POA01_Avg'])
Pred_jan = pd.DataFrame(LR_jan_coeff * Jan['POA01_Avg'])
Pred_feb = pd.DataFrame(LR_feb_coeff * Feb['POA01_Avg'])
Pred_mar21 = pd.DataFrame(LR_mar21_coeff * Mar21['POA01_Avg'])

#45deg poly M6
Pred_marM6 = pd.DataFrame(LR_marM6_coeff * Mar['POA01_Avg'])
Pred_aprM6 = pd.DataFrame(LR_aprM6_coeff * Apr['POA01_Avg'])
Pred_mayM6 = pd.DataFrame(LR_mayM6_coeff * May['POA01_Avg'])
Pred_junM6 = pd.DataFrame(LR_junM6_coeff * Jun['POA01_Avg'])
Pred_julM6 = pd.DataFrame(LR_julM6_coeff * Jul['POA01_Avg'])
Pred_augM6 = pd.DataFrame(LR_augM6_coeff * Aug['POA01_Avg'])
Pred_sepM6 = pd.DataFrame(LR_sepM6_coeff * Sep['POA01_Avg'])
Pred_octM6 = pd.DataFrame(LR_octM6_coeff * Oct['POA01_Avg'])
Pred_novM6 = pd.DataFrame(LR_novM6_coeff * Nov['POA01_Avg'])
Pred_decM6 = pd.DataFrame(LR_decM6_coeff * Dec['POA01_Avg'])
Pred_janM6 = pd.DataFrame(LR_janM6_coeff * Jan['POA01_Avg'])
Pred_febM6 = pd.DataFrame(LR_febM6_coeff * Feb['POA01_Avg'])
Pred_mar21M6 = pd.DataFrame(LR_mar21M6_coeff * Mar21['POA01_Avg'])

#BIPV
Pred_mar_BIPV = pd.DataFrame(LR_mar_BIPV_coeff * Mar['POA02_Avg'])
Pred_apr_BIPV = pd.DataFrame(LR_apr_BIPV_coeff * Apr['POA02_Avg'])
Pred_may_BIPV = pd.DataFrame(LR_may_BIPV_coeff * May['POA02_Avg'])
Pred_jun_BIPV = pd.DataFrame(LR_jun_BIPV_coeff * Jun['POA02_Avg'])
Pred_jul_BIPV = pd.DataFrame(LR_jul_BIPV_coeff * Jul['POA02_Avg'])
Pred_aug_BIPV = pd.DataFrame(LR_aug_BIPV_coeff * Aug['POA02_Avg'])
Pred_sep_BIPV = pd.DataFrame(LR_sep_BIPV_coeff * Sep['POA02_Avg'])
Pred_oct_BIPV = pd.DataFrame(LR_oct_BIPV_coeff * Oct['POA02_Avg'])
Pred_nov_BIPV = pd.DataFrame(LR_nov_BIPV_coeff * Nov['POA02_Avg'])
Pred_dec_BIPV = pd.DataFrame(LR_dec_BIPV_coeff * Dec['POA02_Avg'])
Pred_jan_BIPV = pd.DataFrame(LR_jan_BIPV_coeff * Jan['POA02_Avg'])
Pred_feb_BIPV = pd.DataFrame(LR_feb_BIPV_coeff * Feb['POA02_Avg'])
Pred_mar21_BIPV = pd.DataFrame(LR_mar21_BIPV_coeff * Mar21['POA02_Avg'])

#BIPV GREY
Pred_mar_GREY = pd.DataFrame(LR_mar_GREY_coeff * Mar['POA02_Avg'])
Pred_apr_GREY = pd.DataFrame(LR_apr_GREY_coeff * Apr['POA02_Avg'])
Pred_may_GREY = pd.DataFrame(LR_may_GREY_coeff * May['POA02_Avg'])
Pred_jun_GREY = pd.DataFrame(LR_jun_GREY_coeff * Jun['POA02_Avg'])
Pred_jul_GREY = pd.DataFrame(LR_jul_GREY_coeff * Jul['POA02_Avg'])
Pred_aug_GREY = pd.DataFrame(LR_aug_GREY_coeff * Aug['POA02_Avg'])
Pred_sep_GREY = pd.DataFrame(LR_sep_GREY_coeff * Sep['POA02_Avg'])
Pred_oct_GREY = pd.DataFrame(LR_oct_GREY_coeff * Oct['POA02_Avg'])
Pred_nov_GREY = pd.DataFrame(LR_nov_GREY_coeff * Nov['POA02_Avg'])
Pred_dec_GREY = pd.DataFrame(LR_dec_GREY_coeff * Dec['POA02_Avg'])
Pred_jan_GREY = pd.DataFrame(LR_jan_GREY_coeff * Jan['POA02_Avg'])
Pred_feb_GREY = pd.DataFrame(LR_feb_GREY_coeff * Feb['POA02_Avg'])
Pred_mar21_GREY = pd.DataFrame(LR_mar21_GREY_coeff * Mar21['POA02_Avg'])

#change namer of predicted power, 45deg mono M4
Pred_mar = Pred_mar.rename(columns = {'POA01_Avg':'predicted mar'})
```

```
Pred_apr = Pred_apr.rename(columns = {'POA01_Avg':'predicted apr'})
Pred_may = Pred_may.rename(columns = {'POA01_Avg':'predicted may'})
Pred_jun = Pred_jun.rename(columns = {'POA01_Avg':'predicted jun'})
Pred_jul = Pred_jul.rename(columns = {'POA01_Avg':'predicted jul'})
Pred_aug = Pred_aug.rename(columns = {'POA01_Avg':'predicted aug'})
Pred_sep = Pred_sep.rename(columns = {'POA01_Avg':'predicted sep'})
Pred_oct = Pred_oct.rename(columns = {'POA01_Avg':'predicted oct'})
Pred_nov = Pred_nov.rename(columns = {'POA01_Avg':'predicted nov'})
Pred_dec = Pred_dec.rename(columns = {'POA01_Avg':'predicted dec'})
Pred_jan = Pred_jan.rename(columns = {'POA01_Avg':'predicted jan'})
Pred_feb = Pred_feb.rename(columns = {'POA01_Avg':'predicted feb'})
Pred_mar21 = Pred_mar21.rename(columns = {'POA01_Avg':'predicted mar21'})

#45deg poly M6
Pred_marM6 = Pred_marM6.rename(columns = {'POA01_Avg':'predicted mar'})
Pred_aprM6 = Pred_aprM6.rename(columns = {'POA01_Avg':'predicted apr'})
Pred_mayM6 = Pred_mayM6.rename(columns = {'POA01_Avg':'predicted may'})
Pred_junM6 = Pred_junM6.rename(columns = {'POA01_Avg':'predicted jun'})
Pred_julM6 = Pred_julM6.rename(columns = {'POA01_Avg':'predicted jul'})
Pred_augM6 = Pred_augM6.rename(columns = {'POA01_Avg':'predicted aug'})
Pred_sepM6 = Pred_sepM6.rename(columns = {'POA01_Avg':'predicted sep'})
Pred_octM6 = Pred_octM6.rename(columns = {'POA01_Avg':'predicted oct'})
Pred_novM6 = Pred_novM6.rename(columns = {'POA01_Avg':'predicted nov'})
Pred_decM6 = Pred_decM6.rename(columns = {'POA01_Avg':'predicted dec'})
Pred_janM6 = Pred_janM6.rename(columns = {'POA01_Avg':'predicted jan'})
Pred_febM6 = Pred_febM6.rename(columns = {'POA01_Avg':'predicted feb'})
Pred_mar21M6 = Pred_mar21M6.rename(columns = {'POA01_Avg':'predicted mar21'})

#BIPV
Pred_mar_BIPV = Pred_mar_BIPV.rename(columns = {'POA02_Avg':'predicted mar'})
Pred_apr_BIPV = Pred_apr_BIPV.rename(columns = {'POA02_Avg':'predicted apr'})
Pred_may_BIPV = Pred_may_BIPV.rename(columns = {'POA02_Avg':'predicted may'})
Pred_jun_BIPV = Pred_jun_BIPV.rename(columns = {'POA02_Avg':'predicted jun'})
Pred_jul_BIPV = Pred_jul_BIPV.rename(columns = {'POA02_Avg':'predicted jul'})
Pred_aug_BIPV = Pred_aug_BIPV.rename(columns = {'POA02_Avg':'predicted aug'})
Pred_sep_BIPV = Pred_sep_BIPV.rename(columns = {'POA02_Avg':'predicted sep'})
Pred_oct_BIPV = Pred_oct_BIPV.rename(columns = {'POA02_Avg':'predicted oct'})
Pred_nov_BIPV = Pred_nov_BIPV.rename(columns = {'POA02_Avg':'predicted nov'})
Pred_dec_BIPV = Pred_dec_BIPV.rename(columns = {'POA02_Avg':'predicted dec'})
Pred_jan_BIPV = Pred_jan_BIPV.rename(columns = {'POA02_Avg':'predicted jan'})
Pred_feb_BIPV = Pred_feb_BIPV.rename(columns = {'POA02_Avg':'predicted feb'})
Pred_mar21_BIPV=Pred_mar21_BIPV.rename(columns={'POA02_Avg':'predicted mar21'})

#BIPV GREY
Pred_mar_GREY = Pred_mar_GREY.rename(columns = {'POA02_Avg':'predicted mar'})
Pred_apr_GREY = Pred_apr_GREY.rename(columns = {'POA02_Avg':'predicted apr'})
Pred_may_GREY = Pred_may_GREY.rename(columns = {'POA02_Avg':'predicted may'})
Pred_jun_GREY = Pred_jun_GREY.rename(columns = {'POA02_Avg':'predicted jun'})
Pred_jul_GREY = Pred_jul_GREY.rename(columns = {'POA02_Avg':'predicted jul'})
Pred_aug_GREY = Pred_aug_GREY.rename(columns = {'POA02_Avg':'predicted aug'})
Pred_sep_GREY = Pred_sep_GREY.rename(columns = {'POA02_Avg':'predicted sep'})
Pred_oct_GREY = Pred_oct_GREY.rename(columns = {'POA02_Avg':'predicted oct'})
Pred_nov_GREY = Pred_nov_GREY.rename(columns = {'POA02_Avg':'predicted nov'})
Pred_dec_GREY = Pred_dec_GREY.rename(columns = {'POA02_Avg':'predicted dec'})
Pred_jan_GREY = Pred_jan_GREY.rename(columns = {'POA02_Avg':'predicted jan'})
Pred_feb_GREY = Pred_feb_GREY.rename(columns = {'POA02_Avg':'predicted feb'})
Pred_mar21_GREY= red_mar21_GREY.rename(columns={'POA02_Avg':'predicted mar21'})
```

```
df_mar = pd.concat([Mar, Pred_mar, Pcorr_mar], axis=1)
df_apr = pd.concat([Apr, Pred_apr, Pcorr_apr], axis=1)
df_may = pd.concat([May, Pred_may, Pcorr_may], axis=1)
df_jun = pd.concat([Jun, Pred_jun, Pcorr_jun], axis=1)
df_jul = pd.concat([Jul, Pred_jul, Pcorr_jul], axis=1)
df_aug = pd.concat([Aug, Pred_aug, Pcorr_aug], axis=1)
df_sep = pd.concat([Sep, Pred_sep, Pcorr_sep], axis=1)
df_oct = pd.concat([Oct, Pred_oct, Pcorr_oct], axis=1)
df_nov = pd.concat([Nov, Pred_nov, Pcorr_nov], axis=1)
df_dec = pd.concat([Dec, Pred_dec, Pcorr_dec], axis=1)
df_jan = pd.concat([Jan, Pred_jan, Pcorr_jan], axis=1)
df_feb = pd.concat([Feb, Pred_feb, Pcorr_feb], axis=1)
df_mar21 = pd.concat([Mar21, Pred_mar21, Pcorr_mar21], axis=1)

df_marM6 = pd.concat([Mar, Pred_marM6, PcorrM6_mar], axis=1)
df_aprM6 = pd.concat([Apr, Pred_aprM6, PcorrM6_apr], axis=1)
df_mayM6 = pd.concat([May, Pred_mayM6, PcorrM6_may], axis=1)
df_junM6 = pd.concat([Jun, Pred_junM6, PcorrM6_jun], axis=1)
df_julM6 = pd.concat([Jul, Pred_julM6, PcorrM6_jul], axis=1)
df_augM6 = pd.concat([Aug, Pred_augM6, PcorrM6_aug], axis=1)
df_sepM6 = pd.concat([Sep, Pred_sepM6, PcorrM6_sep], axis=1)
df_octM6 = pd.concat([Oct, Pred_octM6, PcorrM6_oct], axis=1)
df_novM6 = pd.concat([Nov, Pred_novM6, PcorrM6_nov], axis=1)
df_decM6 = pd.concat([Dec, Pred_decM6, PcorrM6_dec], axis=1)
df_janM6 = pd.concat([Jan, Pred_janM6, PcorrM6_jan], axis=1)
df_febM6 = pd.concat([Feb, Pred_febM6, PcorrM6_feb], axis=1)
df_mar21M6 = pd.concat([Mar21, Pred_mar21M6, PcorrM6_mar21], axis=1)

df_mar_BIPV = pd.concat([Mar, Pred_mar_BIPV, Pcorr_mar_BIPV], axis=1)
df_apr_BIPV = pd.concat([Apr, Pred_apr_BIPV, Pcorr_apr_BIPV], axis=1)
df_may_BIPV = pd.concat([May, Pred_may_BIPV, Pcorr_may_BIPV], axis=1)
df_jun_BIPV = pd.concat([Jun, Pred_jun_BIPV, Pcorr_jun_BIPV], axis=1)
df_jul_BIPV = pd.concat([Jul, Pred_jul_BIPV, Pcorr_jul_BIPV], axis=1)
df_aug_BIPV = pd.concat([Aug, Pred_aug_BIPV, Pcorr_aug_BIPV], axis=1)
df_sep_BIPV = pd.concat([Sep, Pred_sep_BIPV, Pcorr_sep_BIPV], axis=1)
df_oct_BIPV = pd.concat([Oct, Pred_oct_BIPV, Pcorr_oct_BIPV], axis=1)
df_nov_BIPV = pd.concat([Nov, Pred_nov_BIPV, Pcorr_nov_BIPV], axis=1)
df_dec_BIPV = pd.concat([Dec, Pred_dec_BIPV, Pcorr_dec_BIPV], axis=1)
df_jan_BIPV = pd.concat([Jan, Pred_jan_BIPV, Pcorr_jan_BIPV], axis=1)
df_feb_BIPV = pd.concat([Feb, Pred_feb_BIPV, Pcorr_feb_BIPV], axis=1)
df_mar21_BIPV = pd.concat([Mar21, Pred_mar21_BIPV, Pcorr_mar21_BIPV], axis=1)

df_mar_GREY = pd.concat([Mar, Pred_mar_GREY, Pcorr_mar_GREY], axis=1)
df_apr_GREY = pd.concat([Apr, Pred_apr_GREY, Pcorr_apr_GREY], axis=1)
df_may_GREY = pd.concat([May, Pred_may_GREY, Pcorr_may_GREY], axis=1)
df_jun_GREY = pd.concat([Jun, Pred_jun_GREY, Pcorr_jun_GREY], axis=1)
df_jul_GREY = pd.concat([Jul, Pred_jul_GREY, Pcorr_jul_GREY], axis=1)
df_aug_GREY = pd.concat([Aug, Pred_aug_GREY, Pcorr_aug_GREY], axis=1)
df_sep_GREY = pd.concat([Sep, Pred_sep_GREY, Pcorr_sep_GREY], axis=1)
df_oct_GREY = pd.concat([Oct, Pred_oct_GREY, Pcorr_oct_GREY], axis=1)
df_nov_GREY = pd.concat([Nov, Pred_nov_GREY, Pcorr_nov_GREY], axis=1)
df_dec_GREY = pd.concat([Dec, Pred_dec_GREY, Pcorr_dec_GREY], axis=1)
df_jan_GREY = pd.concat([Jan, Pred_jan_GREY, Pcorr_jan_GREY], axis=1)
df_feb_GREY = pd.concat([Feb, Pred_feb_GREY, Pcorr_feb_GREY], axis=1)
df_mar21_GREY = pd.concat([Mar21, Pred_mar21_GREY, Pcorr_mar21_GREY], axis=1)
```

```
df_mar = df_mar.rename(columns = {0:'P_tc mar'})
df_apr = df_apr.rename(columns = {0:'P_tc apr'})
df_may = df_may.rename(columns = {0:'P_tc may'})
df_jun = df_jun.rename(columns = {0:'P_tc jun'})
df_jul = df_jul.rename(columns = {0:'P_tc jul'})
df_aug = df_aug.rename(columns = {0:'P_tc aug'})
df_sep = df_sep.rename(columns = {0:'P_tc sep'})
df_oct = df_oct.rename(columns = {0:'P_tc oct'})
df_nov = df_nov.rename(columns = {0:'P_tc nov'})
df_dec = df_dec.rename(columns = {0:'P_tc dec'})
df_jan = df_jan.rename(columns = {0:'P_tc jan'})
df_feb = df_feb.rename(columns = {0:'P_tc feb'})
df_mar21 = df_mar21.rename(columns = {0:'P_tc mar21'})

df_marM6 = df_marM6.rename(columns = {0:'P_tc mar'})
df_aprM6 = df_aprM6.rename(columns = {0:'P_tc apr'})
df_mayM6 = df_mayM6.rename(columns = {0:'P_tc may'})
df_junM6 = df_junM6.rename(columns = {0:'P_tc jun'})
df_julM6 = df_julM6.rename(columns = {0:'P_tc jul'})
df_augM6 = df_augM6.rename(columns = {0:'P_tc aug'})
df_sepM6 = df_sepM6.rename(columns = {0:'P_tc sep'})
df_octM6 = df_octM6.rename(columns = {0:'P_tc oct'})
df_novM6 = df_novM6.rename(columns = {0:'P_tc nov'})
df_decM6 = df_decM6.rename(columns = {0:'P_tc dec'})
df_janM6 = df_janM6.rename(columns = {0:'P_tc jan'})
df_febM6 = df_febM6.rename(columns = {0:'P_tc feb'})
df_mar21M6 = df_mar21M6.rename(columns = {0:'P_tc mar21'})

df_mar_BIPV = df_mar_BIPV.rename(columns = {0:'P_tc mar'})
df_apr_BIPV = df_apr_BIPV.rename(columns = {0:'P_tc apr'})
df_may_BIPV = df_may_BIPV.rename(columns = {0:'P_tc may'})
df_jun_BIPV = df_jun_BIPV.rename(columns = {0:'P_tc jun'})
df_jul_BIPV = df_jul_BIPV.rename(columns = {0:'P_tc jul'})
df_aug_BIPV = df_aug_BIPV.rename(columns = {0:'P_tc aug'})
df_sep_BIPV = df_sep_BIPV.rename(columns = {0:'P_tc sep'})
df_oct_BIPV = df_oct_BIPV.rename(columns = {0:'P_tc oct'})
df_nov_BIPV = df_nov_BIPV.rename(columns = {0:'P_tc nov'})
df_dec_BIPV = df_dec_BIPV.rename(columns = {0:'P_tc dec'})
df_jan_BIPV = df_jan_BIPV.rename(columns = {0:'P_tc jan'})
df_feb_BIPV = df_feb_BIPV.rename(columns = {0:'P_tc feb'})
df_mar21_BIPV = df_mar21_BIPV.rename(columns = {0:'P_tc mar21'})

df_mar_GREY = df_mar_GREY.rename(columns = {0:'P_tc mar'})
df_apr_GREY = df_apr_GREY.rename(columns = {0:'P_tc apr'})
df_may_GREY = df_may_GREY.rename(columns = {0:'P_tc may'})
df_jun_GREY = df_jun_GREY.rename(columns = {0:'P_tc jun'})
df_jul_GREY = df_jul_GREY.rename(columns = {0:'P_tc jul'})
df_aug_GREY = df_aug_GREY.rename(columns = {0:'P_tc aug'})
df_sep_GREY = df_sep_GREY.rename(columns = {0:'P_tc sep'})
df_oct_GREY = df_oct_GREY.rename(columns = {0:'P_tc oct'})
df_nov_GREY = df_nov_GREY.rename(columns = {0:'P_tc nov'})
df_dec_GREY = df_dec_GREY.rename(columns = {0:'P_tc dec'})
df_jan_GREY = df_jan_GREY.rename(columns = {0:'P_tc jan'})
df_feb_GREY = df_feb_GREY.rename(columns = {0:'P_tc feb'})
df_mar21_GREY = df_mar21_GREY.rename(columns = {0:'P_tc mar21'})

MarM4 = df_mar.mask(df_mar['P_tc mar'] < 0.1)
```

```
AprM4 = df_apr.mask(df_apr['P_tc apr'] < 0.1)
MayM4 = df_may.mask(df_may['P_tc may'] < 0.1)
JunM4 = df_jun.mask(df_jun['P_tc jun'] < 0.1)
JulM4 = df_jul.mask(df_jul['P_tc jul'] < 0.1)
AugM4 = df_aug.mask(df_aug['P_tc aug'] < 0.1)
SepM4 = df_sep.mask(df_sep['P_tc sep'] < 0.1)
OctM4 = df_oct.mask(df_oct['P_tc oct'] < 0.1)
NovM4 = df_nov.mask(df_nov['P_tc nov'] < 0.1)
DecM4 = df_dec.mask(df_dec['P_tc dec'] < 0.1)
JanM4 = df_jan.mask(df_jan['P_tc jan'] < 0.1)
FebM4 = df_feb.mask(df_feb['P_tc feb'] < 0.1)
Mar21M4 = df_mar21.mask(df_mar21['P_tc mar21'] < 0.1)

MarM6 = df_marM6.mask(df_marM6['P_tc mar'] < 0.1)
AprM6 = df_aprM6.mask(df_aprM6['P_tc apr'] < 0.1)
MayM6 = df_mayM6.mask(df_mayM6['P_tc may'] < 0.1)
JunM6 = df_junM6.mask(df_junM6['P_tc jun'] < 0.1)
JulM6 = df_julM6.mask(df_julM6['P_tc jul'] < 0.1)
AugM6 = df_augM6.mask(df_augM6['P_tc aug'] < 0.1)
SepM6 = df_sepM6.mask(df_sepM6['P_tc sep'] < 0.1)
OctM6 = df_octM6.mask(df_octM6['P_tc oct'] < 0.1)
NovM6 = df_novM6.mask(df_novM6['P_tc nov'] < 0.1)
DecM6 = df_decM6.mask(df_decM6['P_tc dec'] < 0.1)
JanM6 = df_janM6.mask(df_janM6['P_tc jan'] < 0.1)
FebM6 = df_febM6.mask(df_febM6['P_tc feb'] < 0.1)
Mar21M6 = df_mar21M6.mask(df_mar21M6['P_tc mar21'] < 0.1)

Mar_BIPV = df_mar_BIPV.mask(df_mar_BIPV['P_tc mar'] < 0.1)
Apr_BIPV = df_apr_BIPV.mask(df_apr_BIPV['P_tc apr'] < 0.1)
May_BIPV = df_may_BIPV.mask(df_may_BIPV['P_tc may'] < 0.1)
Jun_BIPV = df_jun_BIPV.mask(df_jun_BIPV['P_tc jun'] < 0.1)
Jul_BIPV = df_jul_BIPV.mask(df_jul_BIPV['P_tc jul'] < 0.1)
Aug_BIPV = df_aug_BIPV.mask(df_aug_BIPV['P_tc aug'] < 0.1)
Sep_BIPV = df_sep_BIPV.mask(df_sep_BIPV['P_tc sep'] < 0.1)
Oct_BIPV = df_oct_BIPV.mask(df_oct_BIPV['P_tc oct'] < 0.1)
Nov_BIPV = df_nov_BIPV.mask(df_nov_BIPV['P_tc nov'] < 0.1)
Dec_BIPV = df_dec_BIPV.mask(df_dec_BIPV['P_tc dec'] < 0.1)
Jan_BIPV = df_jan_BIPV.mask(df_jan_BIPV['P_tc jan'] < 0.1)
Feb_BIPV = df_feb_BIPV.mask(df_feb_BIPV['P_tc feb'] < 0.1)
Mar21_BIPV = df_mar21_BIPV.mask(df_mar21_BIPV['P_tc mar21'] < 0.1)

Mar_GREY = df_mar_GREY.mask(df_mar_GREY['P_tc mar'] < 0.1)
Apr_GREY = df_apr_GREY.mask(df_apr_GREY['P_tc apr'] < 0.1)
May_GREY = df_may_GREY.mask(df_may_GREY['P_tc may'] < 0.1)
Jun_GREY = df_jun_GREY.mask(df_jun_GREY['P_tc jun'] < 0.1)
Jul_GREY = df_jul_GREY.mask(df_jul_GREY['P_tc jul'] < 0.1)
Aug_GREY = df_aug_GREY.mask(df_aug_GREY['P_tc aug'] < 0.1)
Sep_GREY = df_sep_GREY.mask(df_sep_GREY['P_tc sep'] < 0.1)
Oct_GREY = df_oct_GREY.mask(df_oct_GREY['P_tc oct'] < 0.1)
Nov_GREY = df_nov_GREY.mask(df_nov_GREY['P_tc nov'] < 0.1)
Dec_GREY = df_dec_GREY.mask(df_dec_GREY['P_tc dec'] < 0.1)
Jan_GREY = df_jan_GREY.mask(df_jan_GREY['P_tc jan'] < 0.1)
Feb_GREY = df_feb_GREY.mask(df_feb_GREY['P_tc feb'] < 0.1)
Mar21_GREY = df_mar21_GREY.mask(df_mar21_GREY['P_tc mar21'] < 0.1)

MarM4 = MarM4.loc[:,['P_tc mar']]
AprM4 = AprM4.loc[:,['P_tc apr']]
```

```
MayM4 = MayM4.loc[:,['P_tc may']]
JunM4 = JunM4.loc[:,['P_tc jun']]
JulM4 = JulM4.loc[:,['P_tc jul']]
augM4 = AugM4.loc[:,['P_tc aug']]
SepM4 = SepM4.loc[:,['P_tc sep']]
OctM4 = OctM4.loc[:,['P_tc oct']]
NovM4 = NovM4.loc[:,['P_tc nov']]
DecM4 = DecM4.loc[:,['P_tc dec']]
JanM4 = JanM4.loc[:,['P_tc jan']]
FebM4 = FebM4.loc[:,['P_tc feb']]
Mar21M4 = Mar21M4.loc[:,['P_tc mar21']]

MarM6 = MarM6.loc[:,['P_tc mar']]
AprM6 = AprM6.loc[:,['P_tc apr']]
MayM6 = MayM6.loc[:,['P_tc may']]
JunM6 = JunM6.loc[:,['P_tc jun']]
JulM6 = JulM6.loc[:,['P_tc jul']]
augM6 = AugM6.loc[:,['P_tc aug']]
SepM6 = SepM6.loc[:,['P_tc sep']]
OctM6 = OctM6.loc[:,['P_tc oct']]
NovM6 = NovM6.loc[:,['P_tc nov']]
DecM6 = DecM6.loc[:,['P_tc dec']]
JanM6 = JanM6.loc[:,['P_tc jan']]
FebM6 = FebM6.loc[:,['P_tc feb']]
Mar21M6 = Mar21M6.loc[:,['P_tc mar21']]

Mar_BIPV = Mar_BIPV.loc[:,['P_tc mar']]
Apr_BIPV = Apr_BIPV.loc[:,['P_tc apr']]
May_BIPV = May_BIPV.loc[:,['P_tc may']]
Jun_BIPV = Jun_BIPV.loc[:,['P_tc jun']]
Jul_BIPV = Jul_BIPV.loc[:,['P_tc jul']]
aug_BIPV = Aug_BIPV.loc[:,['P_tc aug']]
Sep_BIPV = Sep_BIPV.loc[:,['P_tc sep']]
Oct_BIPV = Oct_BIPV.loc[:,['P_tc oct']]
Nov_BIPV = Nov_BIPV.loc[:,['P_tc nov']]
Dec_BIPV = Dec_BIPV.loc[:,['P_tc dec']]
Jan_BIPV = Jan_BIPV.loc[:,['P_tc jan']]
Feb_BIPV = Feb_BIPV.loc[:,['P_tc feb']]
Mar21_BIPV = Mar21_BIPV.loc[:,['P_tc mar21']]

Mar_GREY = Mar_GREY.loc[:,['P_tc mar']]
Apr_GREY = Apr_GREY.loc[:,['P_tc apr']]
May_GREY = May_GREY.loc[:,['P_tc may']]
Jun_GREY = Jun_GREY.loc[:,['P_tc jun']]
Jul_GREY = Jul_GREY.loc[:,['P_tc jul']]
aug_GREY = Aug_GREY.loc[:,['P_tc aug']]
Sep_GREY = Sep_GREY.loc[:,['P_tc sep']]
Oct_GREY = Oct_GREY.loc[:,['P_tc oct']]
Nov_GREY = Nov_GREY.loc[:,['P_tc nov']]
Dec_GREY = Dec_GREY.loc[:,['P_tc dec']]
Jan_GREY = Jan_GREY.loc[:,['P_tc jan']]
Feb_GREY = Feb_GREY.loc[:,['P_tc feb']]
Mar21_GREY = Mar21_GREY.loc[:,['P_tc mar21']]

MarM4 = MarM4.rename(columns = {'P_tc mar':'Power mar'})
AprM4 = AprM4.rename(columns = {'P_tc apr':'Power apr'})
MayM4 = MayM4.rename(columns = {'P_tc may':'Power may'})
```

```
JunM4 = JunM4.rename(columns = {'P_tc jun':'Power jun'})
JulM4 = JulM4.rename(columns = {'P_tc jul':'Power jul'})
AugM4 = AugM4.rename(columns = {'P_tc aug':'Power aug'})
SepM4 = SepM4.rename(columns = {'P_tc sep':'Power sep'})
OctM4 = OctM4.rename(columns = {'P_tc oct':'Power oct'})
NovM4 = NovM4.rename(columns = {'P_tc nov':'Power nov'})
DecM4 = DecM4.rename(columns = {'P_tc dec':'Power dec'})
JanM4 = JanM4.rename(columns = {'P_tc jan':'Power jan'})
FebM4 = FebM4.rename(columns = {'P_tc feb':'Power feb'})
Mar21M4 = Mar21M4.rename(columns = {'P_tc mar21':'Power mar21'})

MarM6 = MarM6.rename(columns = {'P_tc mar':'Power mar'})
AprM6 = AprM6.rename(columns = {'P_tc apr':'Power apr'})
MayM6 = MayM6.rename(columns = {'P_tc may':'Power may'})
JunM6 = JunM6.rename(columns = {'P_tc jun':'Power jun'})
JulM6 = JulM6.rename(columns = {'P_tc jul':'Power jul'})
AugM6 = AugM6.rename(columns = {'P_tc aug':'Power aug'})
SepM6 = SepM6.rename(columns = {'P_tc sep':'Power sep'})
OctM6 = OctM6.rename(columns = {'P_tc oct':'Power oct'})
NovM6 = NovM6.rename(columns = {'P_tc nov':'Power nov'})
DecM6 = DecM6.rename(columns = {'P_tc dec':'Power dec'})
JanM6 = JanM6.rename(columns = {'P_tc jan':'Power jan'})
FebM6 = FebM6.rename(columns = {'P_tc feb':'Power feb'})
Mar21M6 = Mar21M6.rename(columns = {'P_tc mar21':'Power mar21'})

Mar_BIPV = Mar_BIPV.rename(columns = {'P_tc mar':'Power mar'})
Apr_BIPV = Apr_BIPV.rename(columns = {'P_tc apr':'Power apr'})
May_BIPV = May_BIPV.rename(columns = {'P_tc may':'Power may'})
Jun_BIPV = Jun_BIPV.rename(columns = {'P_tc jun':'Power jun'})
Jul_BIPV = Jul_BIPV.rename(columns = {'P_tc jul':'Power jul'})
Aug_BIPV = Aug_BIPV.rename(columns = {'P_tc aug':'Power aug'})
Sep_BIPV = Sep_BIPV.rename(columns = {'P_tc sep':'Power sep'})
Oct_BIPV = Oct_BIPV.rename(columns = {'P_tc oct':'Power oct'})
Nov_BIPV = Nov_BIPV.rename(columns = {'P_tc nov':'Power nov'})
Dec_BIPV = Dec_BIPV.rename(columns = {'P_tc dec':'Power dec'})
Jan_BIPV = Jan_BIPV.rename(columns = {'P_tc jan':'Power jan'})
Feb_BIPV = Feb_BIPV.rename(columns = {'P_tc feb':'Power feb'})
Mar21_BIPV = Mar21_BIPV.rename(columns = {'P_tc mar21':'Power mar21'})

Mar_GREY = Mar_GREY.rename(columns = {'P_tc mar':'Power mar'})
Apr_GREY = Apr_GREY.rename(columns = {'P_tc apr':'Power apr'})
May_GREY = May_GREY.rename(columns = {'P_tc may':'Power may'})
Jun_GREY = Jun_GREY.rename(columns = {'P_tc jun':'Power jun'})
Jul_GREY = Jul_GREY.rename(columns = {'P_tc jul':'Power jul'})
Aug_GREY = Aug_GREY.rename(columns = {'P_tc aug':'Power aug'})
Sep_GREY = Sep_GREY.rename(columns = {'P_tc sep':'Power sep'})
Oct_GREY = Oct_GREY.rename(columns = {'P_tc oct':'Power oct'})
Nov_GREY = Nov_GREY.rename(columns = {'P_tc nov':'Power nov'})
Dec_GREY = Dec_GREY.rename(columns = {'P_tc dec':'Power dec'})
Jan_GREY = Jan_GREY.rename(columns = {'P_tc jan':'Power jan'})
Feb_GREY = Feb_GREY.rename(columns = {'P_tc feb':'Power feb'})
Mar21_GREY = Mar21_GREY.rename(columns = {'P_tc mar21':'Power mar21'})

df_mar = pd.concat([df_mar, MarM4], axis=1)
df_apr = pd.concat([df_apr, AprM4], axis=1)
df_may = pd.concat([df_may, MayM4], axis=1)
df_jun = pd.concat([df_jun, JunM4], axis=1)
```

```
df_jul = pd.concat([df_jul, JulM4], axis=1)
df_aug = pd.concat([df_aug, AugM4], axis=1)
df_sep = pd.concat([df_sep, SepM4], axis=1)
df_oct = pd.concat([df_oct, OctM4], axis=1)
df_nov = pd.concat([df_nov, NovM4], axis=1)
df_dec = pd.concat([df_dec, DecM4], axis=1)
df_jan = pd.concat([df_jan, JanM4], axis=1)
df_feb = pd.concat([df_feb, FebM4], axis=1)
df_mar21 = pd.concat([df_mar21, Mar21M4], axis=1)

df_marM6 = pd.concat([df_marM6, MarM6], axis=1)
df_aprM6 = pd.concat([df_aprM6, AprM6], axis=1)
df_mayM6 = pd.concat([df_mayM6, MayM6], axis=1)
df_junM6 = pd.concat([df_junM6, JunM6], axis=1)
df_julM6 = pd.concat([df_julM6, JulM6], axis=1)
df_augM6 = pd.concat([df_augM6, AugM6], axis=1)
df_sepM6 = pd.concat([df_sepM6, SepM6], axis=1)
df_octM6 = pd.concat([df_octM6, OctM6], axis=1)
df_novM6 = pd.concat([df_novM6, NovM6], axis=1)
df_decM6 = pd.concat([df_decM6, DecM6], axis=1)
df_janM6 = pd.concat([df_janM6, JanM6], axis=1)
df_febM6 = pd.concat([df_febM6, FebM6], axis=1)
df_mar21M6 = pd.concat([df_mar21M6, Mar21M6], axis=1)

df_mar_BIPV = pd.concat([df_mar_BIPV, Mar_BIPV], axis=1)
df_apr_BIPV = pd.concat([df_apr_BIPV, Apr_BIPV], axis=1)
df_may_BIPV = pd.concat([df_may_BIPV, May_BIPV], axis=1)
df_jun_BIPV = pd.concat([df_jun_BIPV, Jun_BIPV], axis=1)
df_jul_BIPV = pd.concat([df_jul_BIPV, Jul_BIPV], axis=1)
df_aug_BIPV = pd.concat([df_aug_BIPV, Aug_BIPV], axis=1)
df_sep_BIPV = pd.concat([df_sep_BIPV, Sep_BIPV], axis=1)
df_oct_BIPV = pd.concat([df_oct_BIPV, Oct_BIPV], axis=1)
df_nov_BIPV = pd.concat([df_nov_BIPV, Nov_BIPV], axis=1)
df_dec_BIPV = pd.concat([df_dec_BIPV, Dec_BIPV], axis=1)
df_jan_BIPV = pd.concat([df_jan_BIPV, Jan_BIPV], axis=1)
df_feb_BIPV = pd.concat([df_feb_BIPV, Feb_BIPV], axis=1)
df_mar21_BIPV = pd.concat([df_mar21_BIPV, Mar21_BIPV], axis=1)

df_mar_GREY = pd.concat([df_mar_GREY, Mar_GREY], axis=1)
df_apr_GREY = pd.concat([df_apr_GREY, Apr_GREY], axis=1)
df_may_GREY = pd.concat([df_may_GREY, May_GREY], axis=1)
df_jun_GREY = pd.concat([df_jun_GREY, Jun_GREY], axis=1)
df_jul_GREY = pd.concat([df_jul_GREY, Jul_GREY], axis=1)
df_aug_GREY = pd.concat([df_aug_GREY, Aug_GREY], axis=1)
df_sep_GREY = pd.concat([df_sep_GREY, Sep_GREY], axis=1)
df_oct_GREY = pd.concat([df_oct_GREY, Oct_GREY], axis=1)
df_nov_GREY = pd.concat([df_nov_GREY, Nov_GREY], axis=1)
df_dec_GREY = pd.concat([df_dec_GREY, Dec_GREY], axis=1)
df_jan_GREY = pd.concat([df_jan_GREY, Jan_GREY], axis=1)
df_feb_GREY = pd.concat([df_feb_GREY, Feb_GREY], axis=1)
df_mar21_GREY = pd.concat([df_mar21_GREY, Mar21_GREY], axis=1)

#fill inn missing nan values with predicted values from
df_mar['Power mar'].fillna(df_mar['predicted mar'], inplace=True)
df_apr['Power apr'].fillna(df_apr['predicted apr'], inplace=True)
df_may['Power may'].fillna(df_may['predicted may'], inplace=True)
df_jun['Power jun'].fillna(df_jun['predicted jun'], inplace=True)
```

```
df_jul['Power jul'].fillna(df_jul['predicted jul'], inplace=True)
df_aug['Power aug'].fillna(df_aug['predicted aug'], inplace=True)
df_sep['Power sep'].fillna(df_sep['predicted sep'], inplace=True)
df_oct['Power oct'].fillna(df_oct['predicted oct'], inplace=True)
df_nov['Power nov'].fillna(df_nov['predicted nov'], inplace=True)
df_dec['Power dec'].fillna(df_dec['predicted dec'], inplace=True)
df_jan['Power jan'].fillna(df_jan['predicted jan'], inplace=True)
df_feb['Power feb'].fillna(df_feb['predicted feb'], inplace=True)
df_mar21['Power mar21'].fillna(df_mar21['predicted mar21'], inplace=True)


df_mar = df_mar.loc[:,['Power mar']]
df_apr = df_apr.loc[:,['Power apr']]
df_may = df_may.loc[:,['Power may']]
df_jun = df_jun.loc[:,['Power jun']]
df_jul = df_jul.loc[:,['Power jul']]
df_aug = df_aug.loc[:,['Power aug']]
df_sep = df_sep.loc[:,['Power sep']]
df_oct = df_oct.loc[:,['Power oct']]
df_nov = df_nov.loc[:,['Power nov']]
df_dec = df_dec.loc[:,['Power dec']]
df_jan = df_jan.loc[:,['Power jan']]
df_feb = df_feb.loc[:,['Power feb']]
df_mar21 = df_mar21.loc[:,['Power mar21']]

df_M4 = pd.concat([df_mar, df_apr, df_may, df_jun, df_jul, df_aug,
                   df_sep, df_oct, df_nov, df_dec, df_jan, df_feb,
                   df_mar21], axis=1)

df_marM6['Power mar'].fillna(df_marM6['predicted mar'], inplace=True)
df_aprM6['Power apr'].fillna(df_aprM6['predicted apr'], inplace=True)
df_mayM6['Power may'].fillna(df_mayM6['predicted may'], inplace=True)
df_junM6['Power jun'].fillna(df_junM6['predicted jun'], inplace=True)
df_julM6['Power jul'].fillna(df_julM6['predicted jul'], inplace=True)
df_augM6['Power aug'].fillna(df_augM6['predicted aug'], inplace=True)
df_sepM6['Power sep'].fillna(df_sepM6['predicted sep'], inplace=True)
df_octM6['Power oct'].fillna(df_octM6['predicted oct'], inplace=True)
df_novM6['Power nov'].fillna(df_novM6['predicted nov'], inplace=True)
df_decM6['Power dec'].fillna(df_decM6['predicted dec'], inplace=True)
df_janM6['Power jan'].fillna(df_janM6['predicted jan'], inplace=True)
df_febM6['Power feb'].fillna(df_febM6['predicted feb'], inplace=True)
df_mar21M6['Power mar21'].fillna(df_mar21M6['predicted mar21'], inplace=True)

df_marM6 = df_marM6.loc[:,['Power mar']]
df_aprM6 = df_aprM6.loc[:,['Power apr']]
df_mayM6 = df_mayM6.loc[:,['Power may']]
df_junM6 = df_junM6.loc[:,['Power jun']]
df_julM6 = df_julM6.loc[:,['Power jul']]
df_augM6 = df_augM6.loc[:,['Power aug']]
df_sepM6 = df_sepM6.loc[:,['Power sep']]
df_octM6 = df_octM6.loc[:,['Power oct']]
df_novM6 = df_novM6.loc[:,['Power nov']]
df_decM6 = df_decM6.loc[:,['Power dec']]
df_janM6 = df_janM6.loc[:,['Power jan']]
df_febM6 = df_febM6.loc[:,['Power feb']]
df_mar21M6 = df_mar21M6.loc[:,['Power mar21']]
```

```
df_M6 = pd.concat([df_marM6, df_aprM6, df_mayM6, df_junM6, df_julM6,
                   df_augM6, df_sepM6, df_octM6, df_novM6, df_decM6,
                   df_janM6, df_febM6, df_mar21M6], axis=1)

df_mar_BIPV['Power mar'].fillna(df_mar_BIPV['predicted mar'], inplace=True)
df_apr_BIPV['Power apr'].fillna(df_apr_BIPV['predicted apr'], inplace=True)
df_may_BIPV['Power may'].fillna(df_may_BIPV['predicted may'], inplace=True)
df_jun_BIPV['Power jun'].fillna(df_jun_BIPV['predicted jun'], inplace=True)
df_jul_BIPV['Power jul'].fillna(df_jul_BIPV['predicted jul'], inplace=True)
df_aug_BIPV['Power aug'].fillna(df_aug_BIPV['predicted aug'], inplace=True)
df_sep_BIPV['Power sep'].fillna(df_sep_BIPV['predicted sep'], inplace=True)
df_oct_BIPV['Power oct'].fillna(df_oct_BIPV['predicted oct'], inplace=True)
df_nov_BIPV['Power nov'].fillna(df_nov_BIPV['predicted nov'], inplace=True)
df_dec_BIPV['Power dec'].fillna(df_dec_BIPV['predicted dec'], inplace=True)
df_jan_BIPV['Power jan'].fillna(df_jan_BIPV['predicted jan'], inplace=True)
df_feb_BIPV['Power feb'].fillna(df_feb_BIPV['predicted feb'], inplace=True)
df_mar21_BIPV['Power mar21'].fillna(df_mar21_BIPV['predicted mar21'],
             inplace=True)

df_mar_BIPV = df_mar_BIPV.loc[:,['Power mar']]
df_apr_BIPV = df_apr_BIPV.loc[:,['Power apr']]
df_may_BIPV = df_may_BIPV.loc[:,['Power may']]
df_jun_BIPV = df_jun_BIPV.loc[:,['Power jun']]
df_jul_BIPV = df_jul_BIPV.loc[:,['Power jul']]
df_aug_BIPV = df_aug_BIPV.loc[:,['Power aug']]
df_sep_BIPV = df_sep_BIPV.loc[:,['Power sep']]
df_oct_BIPV = df_oct_BIPV.loc[:,['Power oct']]
df_nov_BIPV = df_nov_BIPV.loc[:,['Power nov']]
df_dec_BIPV = df_dec_BIPV.loc[:,['Power dec']]
df_jan_BIPV = df_jan_BIPV.loc[:,['Power jan']]
df_feb_BIPV = df_feb_BIPV.loc[:,['Power feb']]
df_mar21_BIPV = df_mar21_BIPV.loc[:,['Power mar21']]

df_BIPV = pd.concat([df_mar_BIPV, df_apr_BIPV, df_may_BIPV, df_jun_BIPV,
                     df_jul_BIPV, df_aug_BIPV, df_sep_BIPV, df_oct_BIPV,
                     df_nov_BIPV, df_dec_BIPV, df_jan_BIPV, df_feb_BIPV,
                     df_mar21_BIPV], axis=1)

df_mar_GREY['Power mar'].fillna(df_mar_GREY['predicted mar'], inplace=True)
df_apr_GREY['Power apr'].fillna(df_apr_GREY['predicted apr'], inplace=True)
df_may_GREY['Power may'].fillna(df_may_GREY['predicted may'], inplace=True)
df_jun_GREY['Power jun'].fillna(df_jun_GREY['predicted jun'], inplace=True)
df_jul_GREY['Power jul'].fillna(df_jul_GREY['predicted jul'], inplace=True)
df_aug_GREY['Power aug'].fillna(df_aug_GREY['predicted aug'], inplace=True)
df_sep_GREY['Power sep'].fillna(df_sep_GREY['predicted sep'], inplace=True)
df_oct_GREY['Power oct'].fillna(df_oct_GREY['predicted oct'], inplace=True)
df_nov_GREY['Power nov'].fillna(df_nov_GREY['predicted nov'], inplace=True)
df_dec_GREY['Power dec'].fillna(df_dec_GREY['predicted dec'], inplace=True)
df_jan_GREY['Power jan'].fillna(df_jan_GREY['predicted jan'], inplace=True)
df_feb_GREY['Power feb'].fillna(df_feb_GREY['predicted feb'], inplace=True)
df_mar21_GREY['Power mar21'].fillna(df_mar21_GREY['predicted mar21'],
             inplace=True)

df_mar_GREY = df_mar_GREY.loc[:,['Power mar']]
df_apr_GREY = df_apr_GREY.loc[:,['Power apr']]
df_may_GREY = df_may_GREY.loc[:,['Power may']]
df_jun_GREY = df_jun_GREY.loc[:,['Power jun']]
```

```
df_jul_GREY = df_jul_GREY.loc[:,['Power jul']]
df_aug_GREY = df_aug_GREY.loc[:,['Power aug']]
df_sep_GREY = df_sep_GREY.loc[:,['Power sep']]
df_oct_GREY = df_oct_GREY.loc[:,['Power oct']]
df_nov_GREY = df_nov_GREY.loc[:,['Power nov']]
df_dec_GREY = df_dec_GREY.loc[:,['Power dec']]
df_jan_GREY = df_jan_GREY.loc[:,['Power jan']]
df_feb_GREY = df_feb_GREY.loc[:,['Power feb']]
df_mar21_GREY = df_mar21_GREY.loc[:,['Power mar21']]

df_GREY = pd.concat([df_mar_GREY, df_apr_GREY, df_may_GREY, df_jun_GREY,
                     df_jul_GREY, df_aug_GREY, df_sep_GREY, df_oct_GREY,
                     df_nov_GREY, df_dec_GREY, df_jan_GREY,
                     df_feb_GREY, df_mar21_GREY], axis=1)

df_M4.to_excel('Pred_M4_2.xlsx')
df_M6.to_excel('Pred_M6_2.xlsx')
df_BIPV.to_excel('Pred_BIPV_2.xlsx')
df_GREY.to_excel('Pred_GREY_2.xlsx')
```

```
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 29 16:49:14 2021
@author: Sigrid
"""
#code to estimate missing module temperarture for April and March
#45deg and BIPV
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

Apr_mean = pd.read_excel('APR_mean.xlsx')
Mar_itas = pd.read_excel('mar2020itasMinute.xlsx')

Apr_mean['TIMESTAMP'] = pd.to_datetime(Apr_mean['TIMESTAMP'])
Mar_itas['TIMESTAMP'] = pd.to_datetime(Mar_itas['TIMESTAMP'])

Apr_time =  Apr_mean.loc[:,['TIMESTAMP']]
Apr_irr  = Apr_mean.loc[:,['POAO1_Avg','POAO2_Avg']]
WSP_apr = Apr_mean.loc[:,['WSP_Avg']]
AmbTemp_apr = Apr_mean.loc[:,['AT02_Avg']]
ModTemp_apr = Apr_mean.loc[:,['PVT14_Avg']]
ModTemp_apr_BIPV = Apr_mean.loc[:,['Temp BIPV']]
ModTemp_apr_GREY = Apr_mean.loc[:,['Temp GREY']]

Mar_time =  Mar_itas.loc[:,['TIMESTAMP']]
Mar_irr  = Mar_itas.loc[:,['POAO2_Avg', 'POAO1_Avg']]
WSP_mar = Mar_itas.loc[:,['WSP_Avg']]
AmbTemp_mar = Mar_itas.loc[:,['AT02_Avg']]

Apr = pd.concat([Apr_time, Apr_irr, WSP_apr, AmbTemp_apr], axis=1)
Mar = pd.concat([Mar_time, Mar_irr, WSP_mar, AmbTemp_mar], axis=1)

a = -3.6
b = -0.075
a_bipv = -3.8
b_bipv = 0.0471
a_grey = -3.8
b_grey = 0.0471

#equation 3
Tm_apr = Apr['POAO1_Avg'] * np.exp(a+b*Apr['WSP_Avg']) + Apr['AT02_Avg']
Tm_mar = Mar['POAO1_Avg'] * np.exp(a+b*Mar['WSP_Avg']) + Mar['AT02_Avg']

Tm_apr_BIPV  = Apr['POAO2_Avg'] * np.exp(a_bipv+b_bipv*Apr['WSP_Avg']) + Apr['AT02_Avg']
Tm_mar_BIPV = Mar['POAO2_Avg'] * np.exp(a_bipv+b_bipv*Mar['WSP_Avg']) + Mar['AT02_Avg']

Tm_apr_GREY  = Apr['POAO2_Avg'] * np.exp(a_grey +b_grey *Apr['WSP_Avg']) + Apr['AT02_Avg']
Tm_mar_GREY = Mar['POAO2_Avg'] * np.exp(a_grey +b_grey *Mar['WSP_Avg']) + Mar['AT02_Avg']

#45deg mono
Mar2 = pd.concat([Mar_itas, Tm_mar], axis=1)
Mar2 = Mar2.rename(columns = {0:'Module Temp M4'})
Mar2['PVT14_Avg'].fillna(Mar2['Module Temp M4'], inplace=True)

Apr2 = pd.concat([Apr_mean, Tm_apr], axis=1)
Apr2 = Apr2.rename(columns = {0:'Module Temp M4'})
```

```
Apr2['PVT14_Avg'].fillna(Apr2['Module Temp M4'], inplace=True)

#45dg poly
Mar_45p = pd.concat([Mar_itas, Tm_mar], axis=1)
Mar_45p = Mar_45p.rename(columns = {0:'Module Temp M4'})
Mar_45p['PVT16_Avg'].fillna(Mar_45p['Module Temp M4'], inplace=True)

Apr_45p = pd.concat([Apr_mean, Tm_apr], axis=1)
Apr_45p = Apr_45p.rename(columns = {0:'Module Temp M4'})
Apr_45p['PVT16_Avg'].fillna(Apr_45p['Module Temp M4'], inplace=True)

#BIPV
Mar_BIPV = pd.concat([Mar_itas, Tm_mar_BIPV], axis=1)
Mar_BIPV = Mar_BIPV.rename(columns = {0:'Module Temp BIPV'})
Mar_BIPV['Temp BIPV'].fillna(Mar_BIPV['Module Temp BIPV'], inplace=True)

Apr_BIPV = pd.concat([Apr_mean, Tm_apr_BIPV], axis=1)
Apr_BIPV = Apr_BIPV.rename(columns = {0:'Module Temp BIPV'})
Apr_BIPV['Temp BIPV'].fillna(Apr_BIPV['Module Temp BIPV'], inplace=True)

#BIPV GREY
Mar_GREY = pd.concat([Mar_itas, Tm_mar_GREY], axis=1)
Mar_GREY = Mar_GREY.rename(columns = {0:'Module Temp GREY'})
Mar_GREY['Temp GREY'].fillna(Mar_GREY['Module Temp GREY'], inplace=True)

Apr_GREY = pd.concat([Apr_mean, Tm_apr_GREY], axis=1)
Apr_GREY = Apr_GREY.rename(columns = {0:'Module Temp GREY'})
Apr_GREY['Temp GREY'].fillna(Apr_GREY['Module Temp GREY'], inplace=True)

plt.plot(Tm_apr, label='Estimated temp a = -3.45')
plt.plot(ModTemp_apr, label='Measured temp')
plt.ylabel('Temperature', fontsize=18)
plt.legend(fontsize=16)
plt.yticks(fontsize=18)
plt.show()

plt.plot(Tm_apr_BIPV, label='Estimated temp')
plt.plot(ModTemp_apr_BIPV, label='Measured temp')
plt.ylabel('Temperature', fontsize=18)
plt.legend(fontsize=16)
plt.yticks(fontsize=18)
plt.show()

Apr2.to_excel('Apr_45m.xlsx')
Apr_45p.to_excel('Apr_45p.xlsx')
Apr_BIPV.to_excel('Apr_BIPV.xlsx')
Apr_GREY.to_excel('Apr_GREY.xlsx')

Mar2.to_excel('Mar_45m.xlsx')
Mar_45p.to_excel('Mar_45p.xlsx')
Mar_BIPV.to_excel('Mar_BIPV.xlsx')
Mar_GREY.to_excel('Mar_GREY.xlsx')
```

```
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 29 08:09:00 2021
@author: Sigrid
"""
#code to converte the power back from Ptcorr to not temperature corrected
#power
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df_M4 = pd.read_excel('Pred_M4_2.xlsx')
df_M6 = pd.read_excel('Pred_M6_2.xlsx')
df_BIPV = pd.read_excel('Pred_BIPV_2.xlsx')
df_GREY = pd.read_excel('Pred_GREY_2.xlsx')

Mar_df = pd.read_excel('Filtered_Mar.xlsx')
Apr_df = pd.read_excel('Filtered_Apr.xlsx')
May_df = pd.read_excel('Filtered_May.xlsx')
Jun_df = pd.read_excel('Filtered_Jun.xlsx')
Jul_df = pd.read_excel('Filtered_Jul.xlsx')
Aug_df = pd.read_excel('Filtered_Aug.xlsx')
Sep_df = pd.read_excel('Filtered_Sep.xlsx')
Oct_df = pd.read_excel('Filtered_Oct.xlsx')
Nov_df = pd.read_excel('Filtered_Nov.xlsx')
Dec_df = pd.read_excel('Filtered_Dec.xlsx')
Jan_df = pd.read_excel('Filtered_Jan.xlsx')
Feb_df = pd.read_excel('Filtered_Feb.xlsx')
Mar21_df = pd.read_excel('Filtered_Mar21.xlsx')

Mar_itas = pd.read_excel('mar2020itasMinute.xlsx')
Apr_itas = pd.read_excel('Apr_mean.xlsx')
May_itas = pd.read_excel('May_mean.xlsx')
Jun_itas = pd.read_excel('Jun_mean.xlsx')
Jul_itas = pd.read_excel('Jul_mean.xlsx')
Aug_itas = pd.read_excel('Aug_mean.xlsx')
    Sep_itas = pd.read_excel('Sep_mean.xlsx')
    Oct_itas = pd.read_excel('Oct_mean.xlsx')
    Nov_itas = pd.read_excel('Nov_mean.xlsx')
    Dec_itas = pd.read_excel('Dec_mean.xlsx')
    Jan_itas = pd.read_excel('Jan_mean.xlsx')
    Feb_itas = pd.read_excel('Feb_mean.xlsx')
    Mar21_itas = pd.read_excel('Mar21_mean.xlsx')

#temperature coefficients
Tcoeff_mono = -0.38/100
Tcoeff_poly = -0.411/100
Tcoeff_B = -0.405/100
Tcoeff_G = -0.405/100

P_marM4=df_M4['Power mar'] * (1 + (Tcoeff_mono) * (Mar_itas['PVT14_Avg'] - 25))
P_aprM4=df_M4['Power apr'] * (1 + (Tcoeff_mono) * (Apr_itas['PVT14_Avg'] - 25))
P_mayM4=df_M4['Power may'] * (1 + (Tcoeff_mono) * (May_itas['PVT14_Avg'] - 25))
P_junM4=df_M4['Power jun'] * (1 + (Tcoeff_mono) * (Jun_itas['PVT14_Avg'] - 25))
P_julM4=df_M4['Power jul'] * (1 + (Tcoeff_mono) * (Jul_itas['PVT14_Avg'] - 25))
P_augM4=df_M4['Power aug'] * (1 + (Tcoeff_mono) * (Aug_itas['PVT14_Avg'] - 25))
P_sepM4=df_M4['Power sep'] * (1 + (Tcoeff_mono) * (Sep_itas['PVT14_Avg'] - 25))
```

```
P_octM4=df_M4['Power oct'] * (1 + (Tcoeff_mono) * (Oct_itas['PVT14_Avg'] - 25))
P_novM4=df_M4['Power nov'] * (1 + (Tcoeff_mono) * (Nov_itas['PVT14_Avg'] - 25))
P_decM4=df_M4['Power dec'] * (1 + (Tcoeff_mono) * (Dec_itas['PVT14_Avg'] - 25))
P_janM4=df_M4['Power jan'] * (1 + (Tcoeff_mono) * (Jan_itas['PVT14_Avg'] - 25))
P_febM4=df_M4['Power feb'] * (1 + (Tcoeff_mono) * (Feb_itas['PVT14_Avg'] - 25))
P_mar21M4=df_M4['Power mar21'] * (1 + (Tcoeff_mono) * (Mar21_itas['PVT14_Avg']
            - 25))

P_marM6=df_M6['Power mar'] * (1 + (Tcoeff_poly) * (Mar_itas['PVT16_Avg'] - 25))
P_aprM6=df_M6['Power apr'] * (1 + (Tcoeff_poly) * (Apr_itas['PVT16_Avg'] - 25))
P_mayM6=df_M6['Power may'] * (1 + (Tcoeff_poly) * (May_itas['PVT16_Avg'] - 25))
P_junM6=df_M6['Power jun'] * (1 + (Tcoeff_poly) * (Jun_itas['PVT16_Avg'] - 25))
P_julM6=df_M6['Power jul'] * (1 + (Tcoeff_poly) * (Jul_itas['PVT16_Avg'] - 25))
P_augM6=df_M6['Power aug'] * (1 + (Tcoeff_poly) * (Aug_itas['PVT16_Avg'] - 25))
P_sepM6=df_M6['Power sep'] * (1 + (Tcoeff_poly) * (Sep_itas['PVT16_Avg'] - 25))
P_octM6=df_M6['Power oct'] * (1 + (Tcoeff_poly) * (Oct_itas['PVT16_Avg'] - 25))
P_novM6=df_M6['Power nov'] * (1 + (Tcoeff_poly) * (Nov_itas['PVT16_Avg'] - 25))
P_decM6=df_M6['Power dec'] * (1 + (Tcoeff_poly) * (Dec_itas['PVT16_Avg'] - 25))
P_janM6=df_M6['Power jan'] * (1 + (Tcoeff_poly) * (Jan_itas['PVT16_Avg'] - 25))
P_febM6=df_M6['Power feb'] * (1 + (Tcoeff_poly) * (Feb_itas['PVT16_Avg'] - 25))
P_mar21M6=df_M6['Power mar21'] * (1 + (Tcoeff_poly) * (Mar21_itas['PVT16_Avg']
             - 25))

P_marBIPV=df_BIPV['Power mar']* (1 + (Tcoeff_B) * (Mar_itas['Temp BIPV'] - 25))
P_aprBIPV=df_BIPV['Power apr']* (1 + (Tcoeff_B) * (Apr_itas['Temp BIPV'] - 25))
P_mayBIPV=df_BIPV['Power may']* (1 + (Tcoeff_B) * (May_itas['Temp BIPV'] - 25))
P_junBIPV=df_BIPV['Power jun']* (1 + (Tcoeff_B) * (Jun_itas['Temp BIPV'] - 25))
P_julBIPV=df_BIPV['Power jul']* (1 + (Tcoeff_B) * (Jul_itas['Temp BIPV'] - 25))
P_augBIPV=df_BIPV['Power aug']* (1 + (Tcoeff_B) * (Aug_itas['Temp BIPV'] - 25))
P_sepBIPV=df_BIPV['Power sep']* (1 + (Tcoeff_B) * (Sep_itas['Temp BIPV'] - 25))
P_octBIPV=df_BIPV['Power oct']* (1 + (Tcoeff_B) * (Oct_itas['Temp BIPV'] - 25))
P_novBIPV=df_BIPV['Power nov']* (1 + (Tcoeff_B) * (Nov_itas['Temp BIPV'] - 25))
P_decBIPV=df_BIPV['Power dec']* (1 + (Tcoeff_B) * (Dec_itas['Temp BIPV'] - 25))
P_janBIPV=df_BIPV['Power jan']* (1 + (Tcoeff_B) * (Jan_itas['Temp BIPV'] - 25))
P_febBIPV=df_BIPV['Power feb']* (1 + (Tcoeff_B) * (Feb_itas['Temp BIPV'] - 25))
P_mar21BIPV=df_BIPV['Power mar21'] * (1 + (Tcoeff_B) * (Mar21_itas['Temp BIPV']
             - 25))

P_marGREY=df_GREY['Power mar']* (1 + (Tcoeff_G) * (Mar_itas['Temp GREY'] - 25))
P_aprGREY=df_GREY['Power apr']* (1 + (Tcoeff_G) * (Apr_itas['Temp GREY'] - 25))
P_mayGREY=df_GREY['Power may']* (1 + (Tcoeff_G) * (May_itas['Temp GREY'] - 25))
P_junGREY=df_GREY['Power jun']* (1 + (Tcoeff_G) * (Jun_itas['Temp GREY'] - 25))
P_julGREY=df_GREY['Power jul']* (1 + (Tcoeff_G) * (Jul_itas['Temp GREY'] - 25))
P_augGREY=df_GREY['Power aug']* (1 + (Tcoeff_G) * (Aug_itas['Temp GREY'] - 25))
P_sepGREY=df_GREY['Power sep']* (1 + (Tcoeff_G) * (Sep_itas['Temp GREY'] - 25))
P_octGREY=df_GREY['Power oct']* (1 + (Tcoeff_G) * (Oct_itas['Temp GREY'] - 25))
P_novGREY=df_GREY['Power nov']* (1 + (Tcoeff_G) * (Nov_itas['Temp GREY'] - 25))
P_decGREY=df_GREY['Power dec']* (1 + (Tcoeff_G) * (Dec_itas['Temp GREY'] - 25))
P_janGREY=df_GREY['Power jan']* (1 + (Tcoeff_G) * (Jan_itas['Temp GREY'] - 25))
P_febGREY=df_GREY['Power feb']* (1 + (Tcoeff_G) * (Feb_itas['Temp GREY'] - 25))
P_mar21GREY=df_GREY['Power mar21'] * (1 + (Tcoeff_G) * (Mar21_itas['Temp GREY']
             - 25))

df_mar = pd.concat([Mar_df, P_marM4, P_marM6, P_marBIPV, P_marGREY], axis=1)
df_apr = pd.concat([Apr_df, P_aprM4, P_aprM6, P_aprBIPV, P_aprGREY], axis=1)
df_may = pd.concat([May_df, P_mayM4, P_mayM6, P_mayBIPV, P_mayGREY], axis=1)
df_jun = pd.concat([Jun_df, P_junM4, P_junM6, P_junBIPV, P_junGREY], axis=1)
```

```
df_jul = pd.concat([Jul_df, P_julM4, P_julM6, P_julBIPV, P_julGREY], axis=1)
df_aug = pd.concat([Aug_df, P_augM4, P_augM6, P_augBIPV, P_augGREY], axis=1)
df_sep = pd.concat([Sep_df, P_sepM4, P_sepM6, P_sepBIPV, P_sepGREY], axis=1)
df_oct = pd.concat([Oct_df, P_octM4, P_octM6, P_octBIPV, P_octGREY], axis=1)
df_nov = pd.concat([Nov_df, P_novM4, P_novM6, P_novBIPV, P_novGREY], axis=1)
df_dec = pd.concat([Dec_df, P_decM4, P_decM6, P_decBIPV, P_decGREY], axis=1)
df_jan = pd.concat([Jan_df, P_janM4, P_janM6, P_janBIPV, P_janGREY], axis=1)
df_feb = pd.concat([Feb_df, P_febM4, P_febM6, P_febBIPV, P_febGREY], axis=1)
df_mar21= pd.concat([Mar21_df, P_mar21M4, P_mar21M6, P_mar21BIPV, P_mar21GREY],
                    axis=1)

df_mar = df_mar.rename(columns = {0:'Power M4'})
df_apr = df_apr.rename(columns = {0:'Power M4'})
df_may = df_may.rename(columns = {0:'Power M4'})
df_jun = df_jun.rename(columns = {0:'Power M4'})
df_jul = df_jul.rename(columns = {0:'Power M4'})
df_aug = df_aug.rename(columns = {0:'Power M4'})
df_sep = df_sep.rename(columns = {0:'Power M4'})
df_oct = df_oct.rename(columns = {0:'Power M4'})
df_nov = df_nov.rename(columns = {0:'Power M4'})
df_dec = df_dec.rename(columns = {0:'Power M4'})
df_jan = df_jan.rename(columns = {0:'Power M4'})
df_feb = df_feb.rename(columns = {0:'Power M4'})
df_mar21 = df_mar21.rename(columns = {0:'Power M4'})

df_mar = df_mar.rename(columns = {1:'Power M6'})
df_apr = df_apr.rename(columns = {1:'Power M6'})
df_may = df_may.rename(columns = {1:'Power M6'})
df_jun = df_jun.rename(columns = {1:'Power M6'})
df_jul = df_jul.rename(columns = {1:'Power M6'})
df_aug = df_aug.rename(columns = {1:'Power M6'})
df_sep = df_sep.rename(columns = {1:'Power M6'})
df_oct = df_oct.rename(columns = {1:'Power M6'})
df_nov = df_nov.rename(columns = {1:'Power M6'})
df_dec = df_dec.rename(columns = {1:'Power M6'})
df_jan = df_jan.rename(columns = {1:'Power M6'})
df_feb = df_feb.rename(columns = {1:'Power M6'})
df_mar21 = df_mar21.rename(columns = {1:'Power M6'})

df_mar = df_mar.rename(columns = {2:'Power BIPV'})
df_apr = df_apr.rename(columns = {2:'Power BIPV'})
df_may = df_may.rename(columns = {2:'Power BIPV'})
df_jun = df_jun.rename(columns = {2:'Power BIPV'})
df_jul = df_jul.rename(columns = {2:'Power BIPV'})
df_aug = df_aug.rename(columns = {2:'Power BIPV'})
df_sep = df_sep.rename(columns = {2:'Power BIPV'})
df_oct = df_oct.rename(columns = {2:'Power BIPV'})
df_nov = df_nov.rename(columns = {2:'Power BIPV'})
df_dec = df_dec.rename(columns = {2:'Power BIPV'})
df_jan = df_jan.rename(columns = {2:'Power BIPV'})
df_feb = df_feb.rename(columns = {2:'Power BIPV'})
df_mar21  = df_mar21.rename(columns = {2:'Power BIPV'})

df_mar = df_mar.rename(columns = {3:'Power GREY'})
df_apr = df_apr.rename(columns = {3:'Power GREY'})
df_may = df_may.rename(columns = {3:'Power GREY'})
df_jun = df_jun.rename(columns = {3:'Power GREY'})
```

```
df_jul = df_jul.rename(columns = {3:'Power GREY'})
df_aug = df_aug.rename(columns = {3:'Power GREY'})
df_sep = df_sep.rename(columns = {3:'Power GREY'})
df_oct = df_oct.rename(columns = {3:'Power GREY'})
df_nov = df_nov.rename(columns = {3:'Power GREY'})
df_dec = df_dec.rename(columns = {3:'Power GREY'})
df_jan = df_jan.rename(columns = {3:'Power GREY'})
df_feb = df_feb.rename(columns = {3:'Power GREY'})
df_mar21 = df_mar21.rename(columns = {3:'Power GREY'})

df_mar[' Pmax_M4 '].fillna(df_mar['Power M4'], inplace=True)
df_apr[' Pmax_M4 '].fillna(df_apr['Power M4'], inplace=True)
df_may[' Pmax_M4 '].fillna(df_may['Power M4'], inplace=True)
df_jun['Pmax_M4_corr'].fillna(df_jun['Power M4'], inplace=True)
df_jul['Pmax_M4_corr'].fillna(df_jul['Power M4'], inplace=True)
df_aug[' Pmax_M4 '].fillna(df_aug['Power M4'], inplace=True)
df_sep[' Pmax_M4 '].fillna(df_sep['Power M4'], inplace=True)
df_oct[' Pmax_M4 '].fillna(df_oct['Power M4'], inplace=True)
df_nov[' Pmax_M4 '].fillna(df_nov['Power M4'], inplace=True)
df_dec[' Pmax_M4 '].fillna(df_dec['Power M4'], inplace=True)
df_jan[' Pmax_M4 '].fillna(df_jan['Power M4'], inplace=True)
df_feb[' Pmax_M4 '].fillna(df_feb['Power M4'], inplace=True)
df_mar21[' Pmax_M4 '].fillna(df_mar21['Power M4'], inplace=True)

df_mar[' Pmax_M6'].fillna(df_mar['Power M6'], inplace=True)
df_apr[' Pmax_M6'].fillna(df_apr['Power M6'], inplace=True)
df_may[' Pmax_M6'].fillna(df_may['Power M6'], inplace=True)
df_jun['Pmax_M6_corr'].fillna(df_jun['Power M6'], inplace=True)
df_jul['Pmax_M5_corr'].fillna(df_jul['Power M6'], inplace=True)
df_aug[' Pmax_M6'].fillna(df_aug['Power M6'], inplace=True)
df_sep[' Pmax_M6'].fillna(df_sep['Power M6'], inplace=True)
df_oct[' Pmax_M6'].fillna(df_oct['Power M6'], inplace=True)
df_nov[' Pmax_M6'].fillna(df_nov['Power M6'], inplace=True)
df_dec[' Pmax_M6'].fillna(df_dec['Power M6'], inplace=True)
df_jan[' Pmax_M6'].fillna(df_jan['Power M6'], inplace=True)
df_feb[' Pmax_M6'].fillna(df_feb['Power M6'], inplace=True)
df_mar21[' Pmax_M6'].fillna(df_mar21['Power M6'], inplace=True)

df_mar[' Pmax_M2 '].fillna(df_mar['Power BIPV'], inplace=True)
df_apr[' Pmax_M2 '].fillna(df_apr['Power BIPV'], inplace=True)
df_may[' Pmax_M2 '].fillna(df_may['Power BIPV'], inplace=True)
df_jun['Pmax_M2_corr'].fillna(df_jun['Power BIPV'], inplace=True)
df_jul['Pmax_M2_corr'].fillna(df_jul['Power BIPV'], inplace=True)
df_aug[' Pmax_M2 '].fillna(df_aug['Power BIPV'], inplace=True)
df_sep[' Pmax_M2 '].fillna(df_sep['Power BIPV'], inplace=True)
df_oct[' Pmax_M2 '].fillna(df_oct['Power BIPV'], inplace=True)
df_nov[' Pmax_M2 '].fillna(df_nov['Power BIPV'], inplace=True)
df_dec[' Pmax_M2 '].fillna(df_dec['Power BIPV'], inplace=True)
df_jan[' Pmax_M2 '].fillna(df_jan['Power BIPV'], inplace=True)
df_feb[' Pmax_M2 '].fillna(df_feb['Power BIPV'], inplace=True)
df_mar21[' Pmax_M2 '].fillna(df_mar21['Power BIPV'], inplace=True)

df_mar[' Pmax_M1 '].fillna(df_mar['Power GREY'], inplace=True)
df_apr[' Pmax_M1 '].fillna(df_apr['Power GREY'], inplace=True)
df_may[' Pmax_M1 '].fillna(df_may['Power GREY'], inplace=True)
df_jun['Pmax_M1_corr'].fillna(df_jun['Power GREY'], inplace=True)
df_jul['Pmax_M1_corr'].fillna(df_jul['Power GREY'], inplace=True)
```

```
df_aug[' Pmax_M1 '].fillna(df_aug['Power GREY'], inplace=True)
df_sep[' Pmax_M1 '].fillna(df_sep['Power GREY'], inplace=True)
df_oct[' Pmax_M1 '].fillna(df_oct['Power GREY'], inplace=True)
df_nov[' Pmax_M1 '].fillna(df_nov['Power GREY'], inplace=True)
df_dec[' Pmax_M1 '].fillna(df_dec['Power GREY'], inplace=True)
df_jan[' Pmax_M1 '].fillna(df_jan['Power GREY'], inplace=True)
df_feb[' Pmax_M1 '].fillna(df_feb['Power GREY'], inplace=True)
df_mar21[' Pmax_M1 '].fillna(df_mar21['Power GREY'], inplace=True)

df_mar.to_excel('Mar_pred5.xlsx')
df_apr.to_excel('Apr_pred5.xlsx')
df_may.to_excel('May_pred5.xlsx')
df_jun.to_excel('Jun_pred5.xlsx')
df_jul.to_excel('Jul_pred5.xlsx')
df_aug.to_excel('Aug_pred5.xlsx')
df_sep.to_excel('Sep_pred5.xlsx')
df_oct.to_excel('Oct_pred5.xlsx')
df_nov.to_excel('Nov_pred5.xlsx')
df_dec.to_excel('Dec_pred5.xlsx')
df_jan.to_excel('Jan_pred5.xlsx')
df_feb.to_excel('Feb_pred5.xlsx')
df_mar21.to_excel('Mar21_pred5.xlsx')
```

```
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 29 15:27:00 2021
@author: Sigrid
"""
#code to calcualte yield based on output power with predicted values
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

MonoM2_20 = pd.read_excel('Data_from_DB_M2_all.xlsx')
MonoU2_20 = pd.read_excel('Data_from_DB_U2_all.xlsx')
PolyC3 = pd.read_excel('Data_from_DB_C3_all.xlsx')
PolyD3 = pd.read_excel('Data_from_DB_D3_all.xlsx')

#Reference yield for mono west (U2)
Mar_itas = pd.read_excel('mar2020itasMinute.xlsx')
Apr_itas = pd.read_excel('apr2020itasMinute.xlsx')
May_itas = pd.read_excel('mai2020itasMinute.xlsx')
Jun_itas = pd.read_excel('jun2020itasMinute.xlsx')
Jul_itas = pd.read_excel('jul2020itasMinute.xlsx')
Aug_itas = pd.read_excel('aug2020itasMinute.xlsx')
Sep_itas = pd.read_excel('sep2020itasMinute.xlsx')
Oct_itas = pd.read_excel('okt2020itasMinute.xlsx')
Nov_itas = pd.read_excel('nov2020itasMinute.xlsx')
Dec_itas = pd.read_excel('des2020itasMinute.xlsx')
Jan_itas = pd.read_excel('jan2021itasMinute.xlsx')
Feb_itas = pd.read_excel('feb2021itasMinute.xlsx')
Mar21_itas = pd.read_excel('mar2021itasMinute.xlsx')

#getting the ouptu power (Wh)
MarM2_tigo = MonoM2_20.loc[0:21607,['DATETIME', 'PowerDC']]
AprM2_tigo = MonoM2_20.loc[21608:47426,['DATETIME', 'PowerDC']]
MayM2_tigo = MonoM2_20.loc[47427:78738,['DATETIME', 'PowerDC']]
JunM2_tigo = MonoM2_20.loc[78739:110617,['DATETIME', 'PowerDC']]
JulM2_tigo = MonoM2_20.loc[110618:142544,['DATETIME', 'PowerDC']]
AugM2_tigo = MonoM2_20.loc[142545:170624,['DATETIME', 'PowerDC']]
SepM2_tigo = MonoM2_20.loc[170625:193034,['DATETIME', 'PowerDC']]
OctM2_tigo = MonoM2_20.loc[193035:211137,['DATETIME', 'PowerDC']]
NovM2_tigo = MonoM2_20.loc[211138:224040,['DATETIME', 'PowerDC']]
DecM2_tigo = MonoM2_20.loc[224041:231916,['DATETIME', 'PowerDC']]
JanM2_tigo = MonoM2_20.loc[231917:243470,['DATETIME', 'PowerDC']]
FebM2_tigo = MonoM2_20.loc[243471:255902,['DATETIME', 'PowerDC']]
Mar21_M2_tigo = MonoM2_20.loc[255903:277618,['DATETIME', 'PowerDC']]

MarM2_tigo['DATETIME'] = pd.to_datetime(MarM2_tigo['DATETIME'])
AprM2_tigo['DATETIME'] = pd.to_datetime(AprM2_tigo['DATETIME'])
MayM2_tigo['DATETIME'] = pd.to_datetime(MayM2_tigo['DATETIME'])
JunM2_tigo['DATETIME'] = pd.to_datetime(JunM2_tigo['DATETIME'])
JulM2_tigo['DATETIME'] = pd.to_datetime(JulM2_tigo['DATETIME'])
AugM2_tigo['DATETIME'] = pd.to_datetime(AugM2_tigo['DATETIME'])
SepM2_tigo['DATETIME'] = pd.to_datetime(SepM2_tigo['DATETIME'])
OctM2_tigo['DATETIME'] = pd.to_datetime(OctM2_tigo['DATETIME'])
NovM2_tigo['DATETIME'] = pd.to_datetime(NovM2_tigo['DATETIME'])
DecM2_tigo['DATETIME'] = pd.to_datetime(DecM2_tigo['DATETIME'])
JanM2_tigo['DATETIME'] = pd.to_datetime(JanM2_tigo['DATETIME'])
FebM2_tigo['DATETIME'] = pd.to_datetime(FebM2_tigo['DATETIME'])
```

```
Mar21_M2_tigo['DATETIME'] = pd.to_datetime(Mar21_M2_tigo['DATETIME'])

Mar_itas['TIMESTAMP'] = pd.to_datetime(Mar_itas['TIMESTAMP'])
Apr_itas['TIMESTAMP'] = pd.to_datetime(Apr_itas['TIMESTAMP'])
May_itas['TIMESTAMP'] = pd.to_datetime(May_itas['TIMESTAMP'])
Jun_itas['TIMESTAMP'] = pd.to_datetime(Jun_itas['TIMESTAMP'])
Jul_itas['TIMESTAMP'] = pd.to_datetime(Jul_itas['TIMESTAMP'])
Aug_itas['TIMESTAMP'] = pd.to_datetime(Aug_itas['TIMESTAMP'])
Sep_itas['TIMESTAMP'] = pd.to_datetime(Sep_itas['TIMESTAMP'])
Oct_itas['TIMESTAMP'] = pd.to_datetime(Oct_itas['TIMESTAMP'])
Nov_itas['TIMESTAMP'] = pd.to_datetime(Nov_itas['TIMESTAMP'])
Dec_itas['TIMESTAMP'] = pd.to_datetime(Dec_itas['TIMESTAMP'])
Jan_itas['TIMESTAMP'] = pd.to_datetime(Jan_itas['TIMESTAMP'])
Feb_itas['TIMESTAMP'] = pd.to_datetime(Feb_itas['TIMESTAMP'])
Mar21_itas['TIMESTAMP'] = pd.to_datetime(Mar21_itas['TIMESTAMP'])

#Reference yield
MarM2_HG = Mar_itas.loc[:,['TIMESTAMP','POA04_Avg']]
AprM2_HG = Apr_itas.loc[:,['TIMESTAMP','POA04_Avg']]
MayM2_HG = May_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JunM2_HG = Jun_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JulM2_HG = Jul_itas.loc[:,['TIMESTAMP','POA04_Avg']]
AugM2_HG = Aug_itas.loc[:,['TIMESTAMP','POA04_Avg']]
SepM2_HG = Sep_itas.loc[:,['TIMESTAMP','POA04_Avg']]
OctM2_HG = Oct_itas.loc[:,['TIMESTAMP','POA04_Avg']]
NovM2_HG = Nov_itas.loc[:,['TIMESTAMP','POA04_Avg']]
DecM2_HG = Dec_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JanM2_HG = Jan_itas.loc[:,['TIMESTAMP','POA04_Avg']]
FebM2_HG = Feb_itas.loc[:,['TIMESTAMP','POA04_Avg']]
Mar21_M2_HG = Mar21_itas.loc[:,['TIMESTAMP','POA04_Avg']]

Mar = MarM2_HG.merge(MarM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Apr = AprM2_HG.merge(AprM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
May = MayM2_HG.merge(MayM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Jun = JunM2_HG.merge(JunM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Jul = JulM2_HG.merge(JulM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Aug = AugM2_HG.merge(AugM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Sep = SepM2_HG.merge(SepM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Oct = OctM2_HG.merge(OctM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Nov = NovM2_HG.merge(NovM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Dec = DecM2_HG.merge(DecM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Jan = JanM2_HG.merge(JanM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Feb = FebM2_HG.merge(FebM2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Mar21 = Mar21_M2_HG.merge(Mar21_M2_tigo, left_on='TIMESTAMP',
                          right_on='DATETIME')

MarM2_sum = np.sum(Mar)*(1/60)
AprM2_sum = np.sum(Apr)*(1/60)
MayM2_sum = np.sum(May)*(1/60)
JunM2_sum = np.sum(Jun)*(1/60)
JulM2_sum = np.sum(Jul)*(1/60)
AugM2_sum = np.sum(Aug)*(1/60)
SepM2_sum = np.sum(Sep)*(1/60)
OctM2_sum = np.sum(Oct)*(1/60)
NovM2_sum = np.sum(Nov)*(1/60)
DecM2_sum = np.sum(Dec)*(1/60)
JanM2_sum = np.sum(Jan)*(1/60)
```

```
FebM2_sum = np.sum(Feb)*(1/60)
Mar21_M2_sum = np.sum(Mar21)*(1/60)

#calculating the Final yield
MarM2_yield = MarM2_sum['PowerDC']/(315)
AprM2_yield = AprM2_sum['PowerDC']/(315)
MayM2_yield = MayM2_sum['PowerDC']/(315)
JunM2_yield = JunM2_sum['PowerDC']/(315)
JulM2_yield = JulM2_sum['PowerDC']/(315)
AugM2_yield = AugM2_sum['PowerDC']/(315)
SepM2_yield = SepM2_sum['PowerDC']/(315)
OctM2_yield = OctM2_sum['PowerDC']/(315)
NovM2_yield = NovM2_sum['PowerDC']/(315)
DecM2_yield = DecM2_sum['PowerDC']/(315)
JanM2_yield = JanM2_sum['PowerDC']/(315)
FebM2_yield = FebM2_sum['PowerDC']/(315)
Mar21_M2_yield = Mar21_M2_sum['PowerDC']/(315)

YieldM2 = np.array([MarM2_yield, AprM2_yield, MayM2_yield, JunM2_yield,
                    JulM2_yield, AugM2_yield, SepM2_yield, OctM2_yield,
                    NovM2_yield, DecM2_yield, JanM2_yield, FebM2_yield,
                    Mar21_M2_yield])
#------------------------------------------------------------------------------
#Mono west U2
MarU2_tigo = MonoU2_20.loc[0:21538,['DATETIME', 'PowerDC']]
AprU2_tigo = MonoU2_20.loc[21539:47426,['DATETIME', 'PowerDC']]
MayU2_tigo = MonoU2_20.loc[47427:78689,['DATETIME', 'PowerDC']]
JunU2_tigo = MonoU2_20.loc[78690:110565,['DATETIME', 'PowerDC']]
JulU2_tigo = MonoU2_20.loc[110566:142469,['DATETIME', 'PowerDC']]
AugU2_tigo = MonoU2_20.loc[142470:170499,['DATETIME', 'PowerDC']]
SepU2_tigo = MonoU2_20.loc[170500:192866,['DATETIME', 'PowerDC']]
OctU2_tigo = MonoU2_20.loc[192867:210887,['DATETIME', 'PowerDC']]
NovU2_tigo = MonoU2_20.loc[210888:223907,['DATETIME', 'PowerDC']]
DecU2_tigo = MonoU2_20.loc[223908:232838,['DATETIME', 'PowerDC']]
JanU2_tigo = MonoU2_20.loc[232839:244671,['DATETIME', 'PowerDC']]
FebU2_tigo = MonoU2_20.loc[244672:256712,['DATETIME', 'PowerDC']]
Mar21_U2_tigo = MonoU2_20.loc[256713:278618,['DATETIME', 'PowerDC']]

MarU2_tigo['DATETIME'] = pd.to_datetime(MarU2_tigo['DATETIME'])
AprU2_tigo['DATETIME'] = pd.to_datetime(AprU2_tigo['DATETIME'])
MayU2_tigo['DATETIME'] = pd.to_datetime(MayU2_tigo['DATETIME'])
JunU2_tigo['DATETIME'] = pd.to_datetime(JunU2_tigo['DATETIME'])
JulU2_tigo['DATETIME'] = pd.to_datetime(JulU2_tigo['DATETIME'])
AugU2_tigo['DATETIME'] = pd.to_datetime(AugU2_tigo['DATETIME'])
SepU2_tigo['DATETIME'] = pd.to_datetime(SepU2_tigo['DATETIME'])
OctU2_tigo['DATETIME'] = pd.to_datetime(OctU2_tigo['DATETIME'])
NovU2_tigo['DATETIME'] = pd.to_datetime(NovU2_tigo['DATETIME'])
DecU2_tigo['DATETIME'] = pd.to_datetime(DecU2_tigo['DATETIME'])
JanU2_tigo['DATETIME'] = pd.to_datetime(JanU2_tigo['DATETIME'])
FebU2_tigo['DATETIME'] = pd.to_datetime(FebU2_tigo['DATETIME'])
Mar21_U2_tigo['DATETIME'] = pd.to_datetime(Mar21_U2_tigo['DATETIME'])

#Reference yield
MarU2_HG = Mar_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
AprU2_HG = Apr_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
MayU2_HG = May_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
JunU2_HG = Jun_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
```

```
JulU2_HG = Jul_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
AugU2_HG = Aug_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
SepU2_HG = Sep_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
OctU2_HG = Oct_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
NovU2_HG = Nov_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
DecU2_HG = Dec_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
JanU2_HG = Jan_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
FebU2_HG = Feb_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
Mar21_U2_HG = Mar21_itas.loc[:,['TIMESTAMP','PoA05_Avg']]

MarU2 = MarU2_HG.merge(MarU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AprU2 = AprU2_HG.merge(AprU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
MayU2 = MayU2_HG.merge(MayU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JunU2 = JunU2_HG.merge(JunU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JulU2 = JulU2_HG.merge(JulU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AugU2 = AugU2_HG.merge(AugU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
SepU2 = SepU2_HG.merge(SepU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
OctU2 = OctU2_HG.merge(OctU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
NovU2 = NovU2_HG.merge(NovU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
DecU2 = DecU2_HG.merge(DecU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JanU2 = JanU2_HG.merge(JanU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
FebU2 = FebU2_HG.merge(FebU2_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Mar21_U2 = Mar21_U2_HG.merge(Mar21_U2_tigo, left_on='TIMESTAMP',
                             right_on='DATETIME')
MarU2_sum = np.sum(MarU2)*(1/60)
AprU2_sum = np.sum(AprU2)*(1/60)
MayU2_sum = np.sum(MayU2)*(1/60)
JunU2_sum = np.sum(JunU2)*(1/60)
JulU2_sum = np.sum(JulU2)*(1/60)
AugU2_sum = np.sum(AugU2)*(1/60)
SepU2_sum = np.sum(SepU2)*(1/60)
OctU2_sum = np.sum(OctU2)*(1/60)
NovU2_sum = np.sum(NovU2)*(1/60)
DecU2_sum = np.sum(DecU2)*(1/60)
JanU2_sum = np.sum(JanU2)*(1/60)
FebU2_sum = np.sum(FebU2)*(1/60)
Mar21_U2_sum = np.sum(Mar21_U2)*(1/60)

#calculating the Final yield
MarU2_yield = MarU2_sum['PowerDC']/(315)
AprU2_yield = AprU2_sum['PowerDC']/(315)
MayU2_yield = MayU2_sum['PowerDC']/(315)
JunU2_yield = JunU2_sum['PowerDC']/(315)
JulU2_yield = JulU2_sum['PowerDC']/(315)
AugU2_yield = AugU2_sum['PowerDC']/(315)
SepU2_yield = SepU2_sum['PowerDC']/(315)
OctU2_yield = OctU2_sum['PowerDC']/(315)
NovU2_yield = NovU2_sum['PowerDC']/(315)
DecU2_yield = DecU2_sum['PowerDC']/(315)
JanU2_yield = JanU2_sum['PowerDC']/(315)
FebU2_yield = FebU2_sum['PowerDC']/(315)
Mar21_U2_yield = Mar21_U2_sum['PowerDC']/(315)

YieldU2 = np.array([MarU2_yield, AprU2_yield, MayU2_yield,
                    JunU2_yield, JulU2_yield, AugU2_yield, SepU2_yield,
                    OctU2_yield, NovU2_yield, DecU2_yield, JanU2_yield,
                    FebU2_yield, Mar21_U2_yield])
```

```
#-------------------------------------------------------------------------------
#Poly C3
MarC3_tigo = PolyC3.loc[0:21430,['DATETIME', 'PowerDC']]
AprC3_tigo = PolyC3.loc[21431:47323,['DATETIME', 'PowerDC']]
MayC3_tigo = PolyC3.loc[47324:78651,['DATETIME', 'PowerDC']]
JunC3_tigo = PolyC3.loc[78652:110625,['DATETIME', 'PowerDC']]
JulC3_tigo = PolyC3.loc[110626:142593,['DATETIME', 'PowerDC']]
AugC3_tigo = PolyC3.loc[142594:170701,['DATETIME', 'PowerDC']]
SepC3_tigo = PolyC3.loc[170702:193169,['DATETIME', 'PowerDC']]
OctC3_tigo = PolyC3.loc[193170:211345,['DATETIME', 'PowerDC']]
NovC3_tigo = PolyC3.loc[211346:224445,['DATETIME', 'PowerDC']]
DecC3_tigo = PolyC3.loc[224446:233339,['DATETIME', 'PowerDC']]
JanC3_tigo = PolyC3.loc[233340:245061,['DATETIME', 'PowerDC']]
FebC3_tigo = PolyC3.loc[245062:256770,['DATETIME', 'PowerDC']]
Mar21_C3_tigo = PolyC3.loc[256771:278618,['DATETIME', 'PowerDC']]

MarC3_tigo['DATETIME'] = pd.to_datetime(MarC3_tigo['DATETIME'])
AprC3_tigo['DATETIME'] = pd.to_datetime(AprC3_tigo['DATETIME'])
MayC3_tigo['DATETIME'] = pd.to_datetime(MayC3_tigo['DATETIME'])
JunC3_tigo['DATETIME'] = pd.to_datetime(JunC3_tigo['DATETIME'])
JulC3_tigo['DATETIME'] = pd.to_datetime(JulC3_tigo['DATETIME'])
AugC3_tigo['DATETIME'] = pd.to_datetime(AugC3_tigo['DATETIME'])
SepC3_tigo['DATETIME'] = pd.to_datetime(SepC3_tigo['DATETIME'])
OctC3_tigo['DATETIME'] = pd.to_datetime(OctC3_tigo['DATETIME'])
NovC3_tigo['DATETIME'] = pd.to_datetime(NovC3_tigo['DATETIME'])
DecC3_tigo['DATETIME'] = pd.to_datetime(DecC3_tigo['DATETIME'])
JanC3_tigo['DATETIME'] = pd.to_datetime(JanC3_tigo['DATETIME'])
FebC3_tigo['DATETIME'] = pd.to_datetime(FebC3_tigo['DATETIME'])
Mar21_C3_tigo['DATETIME'] = pd.to_datetime(Mar21_C3_tigo['DATETIME'])

MarC3_HG = Mar_itas.loc[:,['TIMESTAMP','POA04_Avg']]
AprC3_HG = Apr_itas.loc[:,['TIMESTAMP','POA04_Avg']]
MayC3_HG = May_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JunC3_HG = Jun_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JulC3_HG = Jul_itas.loc[:,['TIMESTAMP','POA04_Avg']]
AugC3_HG = Aug_itas.loc[:,['TIMESTAMP','POA04_Avg']]
SepC3_HG = Sep_itas.loc[:,['TIMESTAMP','POA04_Avg']]
OctC3_HG = Oct_itas.loc[:,['TIMESTAMP','POA04_Avg']]
NovC3_HG = Nov_itas.loc[:,['TIMESTAMP','POA04_Avg']]
DecC3_HG = Dec_itas.loc[:,['TIMESTAMP','POA04_Avg']]
JanC3_HG = Jan_itas.loc[:,['TIMESTAMP','POA04_Avg']]
FebC3_HG = Feb_itas.loc[:,['TIMESTAMP','POA04_Avg']]
Mar21_C3_HG = Mar21_itas.loc[:,['TIMESTAMP','POA04_Avg']]

MarC3 = MarC3_HG.merge(MarC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AprC3 = AprC3_HG.merge(AprC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
MayC3 = MayC3_HG.merge(MayC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JunC3 = JunC3_HG.merge(JunC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JulC3 = JulC3_HG.merge(JulC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AugC3 = AugC3_HG.merge(AugC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
SepC3 = SepC3_HG.merge(SepC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
OctC3 = OctC3_HG.merge(OctC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
NovC3 = NovC3_HG.merge(NovC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
DecC3 = DecC3_HG.merge(DecC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JanC3 = JanC3_HG.merge(JanC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
FebC3 = FebC3_HG.merge(FebC3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Mar21_C3 = Mar21_C3_HG.merge(Mar21_C3_tigo, left_on='TIMESTAMP',
                             right_on='DATETIME')

MarC3_sum = np.sum(MarC3)*(1/60)
AprC3_sum = np.sum(AprC3)*(1/60)
MayC3_sum = np.sum(MayC3)*(1/60)
JunC3_sum = np.sum(JunC3)*(1/60)
JulC3_sum = np.sum(JulC3)*(1/60)
AugC3_sum = np.sum(AugC3)*(1/60)
SepC3_sum = np.sum(SepC3)*(1/60)
OctC3_sum = np.sum(OctC3)*(1/60)
NovC3_sum = np.sum(NovC3)*(1/60)
DecC3_sum = np.sum(DecC3)*(1/60)
JanC3_sum = np.sum(JanC3)*(1/60)
FebC3_sum = np.sum(FebC3)*(1/60)
Mar21_C3_sum = np.sum(Mar21_C3)*(1/60)

MarC3_yield = MarC3_sum['PowerDC']/(270)
AprC3_yield = AprC3_sum['PowerDC']/(270)
MayC3_yield = MayC3_sum['PowerDC']/(270)
JunC3_yield = JunC3_sum['PowerDC']/(270)
JulC3_yield = JulC3_sum['PowerDC']/(270)
AugC3_yield = AugC3_sum['PowerDC']/(270)
SepC3_yield = SepC3_sum['PowerDC']/(270)
OctC3_yield = OctC3_sum['PowerDC']/(270)
NovC3_yield = NovC3_sum['PowerDC']/(270)
DecC3_yield = DecC3_sum['PowerDC']/(270)
JanC3_yield = JanC3_sum['PowerDC']/(270)
FebC3_yield = FebC3_sum['PowerDC']/(270)
Mar21_C3_yield = Mar21_C3_sum['PowerDC']/(270)

YieldC3 = np.array([MarC3_yield, AprC3_yield, MayC3_yield, JunC3_yield,
                    JulC3_yield, AugC3_yield, SepC3_yield, OctC3_yield,
                    NovC3_yield, DecC3_yield, JanC3_yield, FebC3_yield,
                    Mar21_C3_yield])
#------------------------------------------------------------------------------
#Calculations for poly west D3
MarD3_tigo = PolyD3.loc[0:21334,['DATETIME', 'PowerDC']]
AprD3_tigo = PolyD3.loc[21335:46796,['DATETIME', 'PowerDC']]
MayD3_tigo = PolyD3.loc[46797:77546,['DATETIME', 'PowerDC']]
JunD3_tigo = PolyD3.loc[77547:108859,['DATETIME', 'PowerDC']]
JulD3_tigo = PolyD3.loc[108860:140267,['DATETIME', 'PowerDC']]
AugD3_tigo = PolyD3.loc[140268:167841,['DATETIME', 'PowerDC']]
SepD3_tigo = PolyD3.loc[167842:189924,['DATETIME', 'PowerDC']]
OctD3_tigo = PolyD3.loc[189925:207751,['DATETIME', 'PowerDC']]
NovD3_tigo = PolyD3.loc[207752:220502,['DATETIME', 'PowerDC']]
DecD3_tigo = PolyD3.loc[220503:229097,['DATETIME', 'PowerDC']]
JanD3_tigo = PolyD3.loc[229098:240187,['DATETIME', 'PowerDC']]
FebD3_tigo = PolyD3.loc[240188:251643,['DATETIME', 'PowerDC']]
Mar21_D3_tigo = PolyD3.loc[251644:278618,['DATETIME', 'PowerDC']]

MarD3_tigo['DATETIME'] = pd.to_datetime(MarD3_tigo['DATETIME'])
AprD3_tigo['DATETIME'] = pd.to_datetime(AprD3_tigo['DATETIME'])
MayD3_tigo['DATETIME'] = pd.to_datetime(MayD3_tigo['DATETIME'])
JunD3_tigo['DATETIME'] = pd.to_datetime(JunD3_tigo['DATETIME'])
JulD3_tigo['DATETIME'] = pd.to_datetime(JulD3_tigo['DATETIME'])
AugD3_tigo['DATETIME'] = pd.to_datetime(AugD3_tigo['DATETIME'])
SepD3_tigo['DATETIME'] = pd.to_datetime(SepD3_tigo['DATETIME'])
```

```
OctD3_tigo['DATETIME'] = pd.to_datetime(OctD3_tigo['DATETIME'])
NovD3_tigo['DATETIME'] = pd.to_datetime(NovD3_tigo['DATETIME'])
DecD3_tigo['DATETIME'] = pd.to_datetime(DecD3_tigo['DATETIME'])
JanD3_tigo['DATETIME'] = pd.to_datetime(JanD3_tigo['DATETIME'])
FebD3_tigo['DATETIME'] = pd.to_datetime(FebD3_tigo['DATETIME'])
Mar21_D3_tigo['DATETIME'] = pd.to_datetime(Mar21_D3_tigo['DATETIME'])

MarD3_HG = Mar_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
AprD3_HG = Apr_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
MayD3_HG = May_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
JunD3_HG = Jun_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
JulD3_HG = Jul_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
AugD3_HG = Aug_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
SepD3_HG = Sep_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
OctD3_HG = Oct_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
NovD3_HG = Nov_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
DecD3_HG = Dec_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
JanD3_HG = Jan_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
FebD3_HG = Feb_itas.loc[:,['TIMESTAMP','PoA05_Avg']]
Mar21_D3_HG = Mar21_itas.loc[:,['TIMESTAMP','PoA05_Avg']]

MarD3 = MarD3_HG.merge(MarD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AprD3 = AprD3_HG.merge(AprD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
MayD3 = MayD3_HG.merge(MayD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JunD3 = JunD3_HG.merge(JunD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JulD3 = JulD3_HG.merge(JulD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
AugD3 = AugD3_HG.merge(AugD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
SepD3 = SepD3_HG.merge(SepD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
OctD3 = OctD3_HG.merge(OctD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
NovD3 = NovD3_HG.merge(NovD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
DecD3 = DecD3_HG.merge(DecD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
JanD3 = JanD3_HG.merge(JanD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
FebD3 = FebD3_HG.merge(FebD3_tigo, left_on='TIMESTAMP', right_on='DATETIME')
Mar21_D3 = Mar21_D3_HG.merge(Mar21_D3_tigo, left_on='TIMESTAMP',
                             right_on='DATETIME')
MarD3_sum = np.sum(MarD3)*(1/60)
AprD3_sum = np.sum(AprD3)*(1/60)
MayD3_sum = np.sum(MayD3)*(1/60)
JunD3_sum = np.sum(JunD3)*(1/60)
JulD3_sum = np.sum(JulD3)*(1/60)
AugD3_sum = np.sum(AugD3)*(1/60)
SepD3_sum = np.sum(SepD3)*(1/60)
OctD3_sum = np.sum(OctD3)*(1/60)
NovD3_sum = np.sum(NovD3)*(1/60)
DecD3_sum = np.sum(DecD3)*(1/60)
JanD3_sum = np.sum(JanD3)*(1/60)
FebD3_sum = np.sum(FebD3)*(1/60)
Mar21_D3_sum = np.sum(Mar21_D3)*(1/60)

MarD3_yield = MarD3_sum['PowerDC']/(270)
AprD3_yield = AprD3_sum['PowerDC']/(270)
MayD3_yield = MayD3_sum['PowerDC']/(270)
JunD3_yield = JunD3_sum['PowerDC']/(270)
JulD3_yield = JulD3_sum['PowerDC']/(270)
AugD3_yield = AugD3_sum['PowerDC']/(270)
SepD3_yield = SepD3_sum['PowerDC']/(270)
OctD3_yield = OctD3_sum['PowerDC']/(270)
```

```
NovD3_yield = NovD3_sum['PowerDC']/(270)
DecD3_yield = DecD3_sum['PowerDC']/(270)
JanD3_yield = JanD3_sum['PowerDC']/(270)
FebD3_yield = FebD3_sum['PowerDC']/(270)
Mar21_D3_yield = Mar21_D3_sum['PowerDC']/(270)

YieldD3 = np.array([MarD3_yield, AprD3_yield, MayD3_yield, JunD3_yield,
                    JulD3_yield, AugD3_yield, SepD3_yield, OctD3_yield,
                    NovD3_yield, DecD3_yield, JanD3_yield, FebD3_yield,
                    Mar21_D3_yield])
#------------------------------------------------------------------------------
#45deg
df_mar = pd.read_excel('Mar_pred5.xlsx')
df_apr = pd.read_excel('Apr_pred5.xlsx')
df_may = pd.read_excel('May_pred5.xlsx')
df_jun = pd.read_excel('Jun_pred5.xlsx')
df_jul = pd.read_excel('Jul_pred5.xlsx')
df_aug = pd.read_excel('Aug_pred5.xlsx')
df_sep = pd.read_excel('Sep_pred5.xlsx')
df_oct = pd.read_excel('Oct_pred5.xlsx')
df_nov = pd.read_excel('Nov_pred5.xlsx')
df_dec = pd.read_excel('Dec_pred5.xlsx')
df_jan = pd.read_excel('Jan_pred5.xlsx')
df_feb = pd.read_excel('Feb_pred5.xlsx')
df_mar21= pd.read_excel('Mar21_pred5.xlsx')

df_mar['DateAndTime '] = pd.to_datetime(df_mar['DateAndTime '])
df_apr['DateAndTime '] = pd.to_datetime(df_apr['DateAndTime '])
df_may['DateAndTime '] = pd.to_datetime(df_may['DateAndTime '])
df_jun['DateAndTime '] = pd.to_datetime(df_jun['DateAndTime '])
df_jul['DateAndTime '] = pd.to_datetime(df_jul['DateAndTime '])
df_aug['DateAndTime '] = pd.to_datetime(df_aug['DateAndTime '])
df_sep['DateAndTime '] = pd.to_datetime(df_sep['DateAndTime '])
df_oct['DateAndTime '] = pd.to_datetime(df_oct['DateAndTime '])
df_nov['DateAndTime '] = pd.to_datetime(df_nov['DateAndTime '])
df_dec['DateAndTime '] = pd.to_datetime(df_dec['DateAndTime '])
df_jan['DateAndTime '] = pd.to_datetime(df_jan['DateAndTime '])
df_feb['DateAndTime '] = pd.to_datetime(df_feb['DateAndTime '])
df_mar21['DateAndTime '] = pd.to_datetime(df_mar21['DateAndTime '])

Mono_mar = df_mar.loc[:,['DateAndTime ','Power M4']]
Mono_apr = df_apr.loc[:,['DateAndTime ','Power M4']]
Mono_may = df_may.loc[:,['DateAndTime ','Power M4']]
Mono_jun = df_jun.loc[:,['DateAndTime ','Power M4']]
Mono_jul = df_jul.loc[:,['DateAndTime ','Power M4']]
Mono_aug = df_aug.loc[:,['DateAndTime ','Power M4']]
Mono_sep = df_sep.loc[:,['DateAndTime ','Power M4']]
Mono_oct = df_oct.loc[:,['DateAndTime ','Power M4']]
Mono_nov = df_nov.loc[:,['DateAndTime ','Power M4']]
Mono_dec = df_dec.loc[:,['DateAndTime ','Power M4']]
Mono_jan = df_jan.loc[:,['DateAndTime ','Power M4']]
Mono_feb = df_feb.loc[:,['DateAndTime ','Power M4']]
Mono_mar21 = df_mar21.loc[:,['DateAndTime ','Power M4']]

Poly_mar = df_mar.loc[:,['DateAndTime ','Power M6']]
Poly_apr = df_apr.loc[:,['DateAndTime ','Power M6']]
Poly_may = df_may.loc[:,['DateAndTime ','Power M6']]
```

```
Poly_jun = df_jun.loc[:,['DateAndTime ','Power M6']]
Poly_jul = df_jul.loc[:,['DateAndTime ','Power M6']]
Poly_aug = df_aug.loc[:,['DateAndTime ','Power M6']]
Poly_sep = df_sep.loc[:,['DateAndTime ','Power M6']]
Poly_oct = df_oct.loc[:,['DateAndTime ','Power M6']]
Poly_nov = df_nov.loc[:,['DateAndTime ','Power M6']]
Poly_dec = df_dec.loc[:,['DateAndTime ','Power M6']]
Poly_jan = df_jan.loc[:,['DateAndTime ','Power M6']]
Poly_feb = df_feb.loc[:,['DateAndTime ','Power M6']]
Poly_mar21 = df_mar21.loc[:,['DateAndTime ','Power M6']]

Mono_mar_sum = np.sum(Mono_mar)*(1/60)
Mono_apr_sum = np.sum(Mono_apr)*(1/60)
Mono_may_sum = np.sum(Mono_may)*(1/60)
Mono_jun_sum = np.sum(Mono_jun)*(1/60)
Mono_jul_sum = np.sum(Mono_jul)*(1/60)
Mono_aug_sum = np.sum(Mono_aug)*(1/60)
Mono_sep_sum = np.sum(Mono_sep)*(1/60)
Mono_oct_sum = np.sum(Mono_oct)*(1/60)
Mono_nov_sum = np.sum(Mono_nov)*(1/60)
Mono_dec_sum = np.sum(Mono_dec)*(1/60)
Mono_jan_sum = np.sum(Mono_jan)*(1/60)
Mono_feb_sum = np.sum(Mono_feb)*(1/60)
Mono_mar21_sum = np.sum(Mono_mar21)*(1/60)

Poly_mar_sum = np.sum(Poly_mar)*(1/60)
Poly_apr_sum = np.sum(Poly_apr)*(1/60)
Poly_may_sum = np.sum(Poly_may)*(1/60)
Poly_jun_sum = np.sum(Poly_jun)*(1/60)
Poly_jul_sum = np.sum(Poly_jul)*(1/60)
Poly_aug_sum = np.sum(Poly_aug)*(1/60)
Poly_sep_sum = np.sum(Poly_sep)*(1/60)
Poly_oct_sum = np.sum(Poly_oct)*(1/60)
Poly_nov_sum = np.sum(Poly_nov)*(1/60)
Poly_dec_sum = np.sum(Poly_dec)*(1/60)
Poly_jan_sum = np.sum(Poly_jan)*(1/60)
Poly_feb_sum = np.sum(Poly_feb)*(1/60)
Poly_mar21_sum = np.sum(Poly_mar21)*(1/60)

Mono_mar_yield = Mono_mar_sum['Power M4']/315
Mono_apr_yield = Mono_apr_sum['Power M4']/315
Mono_may_yield = Mono_may_sum['Power M4']/315
Mono_jun_yield = Mono_jun_sum['Power M4']/315
Mono_jul_yield = Mono_jul_sum['Power M4']/315
Mono_aug_yield = Mono_aug_sum['Power M4']/315
Mono_sep_yield = Mono_sep_sum['Power M4']/315
Mono_oct_yield = Mono_oct_sum['Power M4']/315
Mono_nov_yield = Mono_nov_sum['Power M4']/315
Mono_dec_yield = Mono_dec_sum['Power M4']/315
Mono_jan_yield = Mono_jan_sum['Power M4']/315
Mono_feb_yield = Mono_feb_sum['Power M4']/315
Mono_mar21_yield = Mono_mar21_sum['Power M4']/315

Poly_mar_yield = Poly_mar_sum['Power M6']/270
Poly_apr_yield = Poly_apr_sum['Power M6']/270
Poly_may_yield = Poly_may_sum['Power M6']/270
Poly_jun_yield = Poly_jun_sum['Power M6']/270
```

```
Poly_jul_yield = Poly_jul_sum['Power M6']/270
Poly_aug_yield = Poly_aug_sum['Power M6']/270
Poly_sep_yield = Poly_sep_sum['Power M6']/270
Poly_oct_yield = Poly_oct_sum['Power M6']/270
Poly_nov_yield = Poly_nov_sum['Power M6']/270
Poly_dec_yield = Poly_dec_sum['Power M6']/270
Poly_jan_yield = Poly_jan_sum['Power M6']/270
Poly_feb_yield = Poly_feb_sum['Power M6']/270
Poly_mar21_yield = Poly_mar21_sum['Power M6']/270

Yield_mono = np.array([Mono_mar_yield, Mono_apr_yield, Mono_may_yield,
                       Mono_jun_yield, Mono_jul_yield, Mono_aug_yield,
                       Mono_sep_yield, Mono_oct_yield, Mono_nov_yield,
                       Mono_dec_yield, Mono_jan_yield, Mono_feb_yield,
                       Mono_mar21_yield])
Yield_poly = np.array([Poly_mar_yield, Poly_apr_yield, Poly_may_yield,
                       Poly_jun_yield, Poly_jul_yield, Poly_aug_yield,
                       Poly_sep_yield, Poly_oct_yield, Poly_nov_yield,
                       Poly_dec_yield, Poly_jan_yield, Poly_feb_yield,
                       Poly_mar21_yield])
#_------------------------------------------------------------------------------
#BIPV
BIPV_mar = df_mar.loc[:,['DateAndTime ','Power BIPV']]
BIPV_apr = df_apr.loc[:,['DateAndTime ','Power BIPV']]
BIPV_may = df_may.loc[:,['DateAndTime ','Power BIPV']]
BIPV_jun = df_jun.loc[:,['DateAndTime ','Power BIPV']]
BIPV_jul = df_jul.loc[:,['DateAndTime ','Power BIPV']]
BIPV_aug = df_aug.loc[:,['DateAndTime ','Power BIPV']]
BIPV_sep = df_sep.loc[:,['DateAndTime ','Power BIPV']]
BIPV_oct = df_oct.loc[:,['DateAndTime ','Power BIPV']]
BIPV_nov = df_nov.loc[:,['DateAndTime ','Power BIPV']]
BIPV_dec = df_dec.loc[:,['DateAndTime ','Power BIPV']]
BIPV_jan = df_jan.loc[:,['DateAndTime ','Power BIPV']]
BIPV_feb = df_feb.loc[:,['DateAndTime ','Power BIPV']]
BIPV_mar21 = df_mar21.loc[:,['DateAndTime ','Power BIPV']]

GREY_mar = df_mar.loc[:,['DateAndTime ','Power GREY']]
GREY_apr = df_apr.loc[:,['DateAndTime ','Power GREY']]
GREY_may = df_may.loc[:,['DateAndTime ','Power GREY']]
GREY_jun = df_jun.loc[:,['DateAndTime ','Power GREY']]
GREY_jul = df_jul.loc[:,['DateAndTime ','Power GREY']]
GREY_aug = df_aug.loc[:,['DateAndTime ','Power GREY']]
GREY_sep = df_sep.loc[:,['DateAndTime ','Power GREY']]
GREY_oct = df_oct.loc[:,['DateAndTime ','Power GREY']]
GREY_nov = df_nov.loc[:,['DateAndTime ','Power GREY']]
GREY_dec = df_dec.loc[:,['DateAndTime ','Power GREY']]
GREY_jan = df_jan.loc[:,['DateAndTime ','Power GREY']]
GREY_feb = df_feb.loc[:,['DateAndTime ','Power GREY']]
GREY_mar21 = df_mar21.loc[:,['DateAndTime ','Power GREY']]

BIPV_mar_sum = np.sum(BIPV_mar)*(1/60)
BIPV_apr_sum = np.sum(BIPV_apr)*(1/60)
BIPV_may_sum = np.sum(BIPV_may)*(1/60)
BIPV_jun_sum = np.sum(BIPV_jun)*(1/60)
BIPV_jul_sum = np.sum(BIPV_jul)*(1/60)
BIPV_aug_sum = np.sum(BIPV_aug)*(1/60)
BIPV_sep_sum = np.sum(BIPV_sep)*(1/60)
```

```
BIPV_oct_sum = np.sum(BIPV_oct)*(1/60)
BIPV_nov_sum = np.sum(BIPV_nov)*(1/60)
BIPV_dec_sum = np.sum(BIPV_dec)*(1/60)
BIPV_jan_sum = np.sum(BIPV_jan)*(1/60)
BIPV_feb_sum = np.sum(BIPV_feb)*(1/60)
BIPV_mar21_sum = np.sum(BIPV_mar21)*(1/60)

GREY_mar_sum = np.sum(GREY_mar)*(1/60)
GREY_apr_sum = np.sum(GREY_apr)*(1/60)
GREY_may_sum = np.sum(GREY_may)*(1/60)
GREY_jun_sum = np.sum(GREY_jun)*(1/60)
GREY_jul_sum = np.sum(GREY_jul)*(1/60)
GREY_aug_sum = np.sum(GREY_aug)*(1/60)
GREY_sep_sum = np.sum(GREY_sep)*(1/60)
GREY_oct_sum = np.sum(GREY_oct)*(1/60)
GREY_nov_sum = np.sum(GREY_nov)*(1/60)
GREY_dec_sum = np.sum(GREY_dec)*(1/60)
GREY_jan_sum = np.sum(GREY_jan)*(1/60)
GREY_feb_sum = np.sum(GREY_feb)*(1/60)
GREY_mar21_sum = np.sum(GREY_mar21)*(1/60)

BIPV_mar_yield = BIPV_mar_sum['Power BIPV']/218.1
BIPV_apr_yield = BIPV_apr_sum['Power BIPV']/218.1
BIPV_may_yield = BIPV_may_sum['Power BIPV']/218.1
BIPV_jun_yield = BIPV_jun_sum['Power BIPV']/218.1
BIPV_jul_yield = BIPV_jul_sum['Power BIPV']/218.1
BIPV_aug_yield = BIPV_aug_sum['Power BIPV']/218.1
BIPV_sep_yield = BIPV_sep_sum['Power BIPV']/218.1
BIPV_oct_yield = BIPV_oct_sum['Power BIPV']/218.1
BIPV_nov_yield = BIPV_nov_sum['Power BIPV']/218.1
BIPV_dec_yield = BIPV_dec_sum['Power BIPV']/218.1
BIPV_jan_yield = BIPV_jan_sum['Power BIPV']/218.1
BIPV_feb_yield = BIPV_feb_sum['Power BIPV']/218.1
BIPV_mar21_yield = BIPV_mar21_sum['Power BIPV']/218.1

GREY_mar_yield = GREY_mar_sum['Power GREY']/193
GREY_apr_yield = GREY_apr_sum['Power GREY']/193
GREY_may_yield = GREY_may_sum['Power GREY']/193
GREY_jun_yield = GREY_jun_sum['Power GREY']/193
GREY_jul_yield = GREY_jul_sum['Power GREY']/193
GREY_aug_yield = GREY_aug_sum['Power GREY']/193
GREY_sep_yield = GREY_sep_sum['Power GREY']/193
GREY_oct_yield = GREY_oct_sum['Power GREY']/193
GREY_nov_yield = GREY_nov_sum['Power GREY']/193
GREY_dec_yield = GREY_dec_sum['Power GREY']/193
GREY_jan_yield = GREY_jan_sum['Power GREY']/193
GREY_feb_yield = GREY_feb_sum['Power GREY']/193
GREY_mar21_yield = GREY_mar21_sum['Power GREY']/193

Yield_BIPV = np.array([BIPV_mar_yield, BIPV_apr_yield, BIPV_may_yield,
                        BIPV_jun_yield, BIPV_jul_yield, BIPV_aug_yield,
                        BIPV_sep_yield, BIPV_oct_yield, BIPV_nov_yield,
                        BIPV_dec_yield, BIPV_jan_yield, BIPV_feb_yield,
                        BIPV_mar21_yield])
Yield_GREY = np.array([GREY_mar_yield, GREY_apr_yield, GREY_may_yield,
                       GREY_jun_yield, GREY_jul_yield, GREY_aug_yield,
                       GREY_sep_yield, GREY_oct_yield, GREY_nov_yield,
                       GREY_dec_yield, GREY_jan_yield, GREY_feb_yield,
                       GREY_mar21_yield])
#------------------------------------------------------------------------------
Months=np.array(['Mar 20', 'Apr 20','May 20', 'Jun 20', 'Jul 20',
                 'Aug 20', 'Sep 20', 'Oct 20',
                 'Nov 20', 'Dec 20', 'Jan 21', 'Feb 21', 'Mar 21'])
plt.plot(Months, YieldM2, marker='o', color='grey', label='Mono 10deg East')
plt.plot(Months, YieldU2, marker='o', color='orange', label='Mono 10deg West')
plt.plot(Months, YieldC3, marker='o', color='b', label='Poly 10deg East')
plt.plot(Months, YieldD3,  marker='o', color='#ff7f24',label='Poly 10deg West')
plt.plot(Months, Yield_mono, marker='o', color='green', label='45deg mono')
plt.plot(Months, Yield_poly,  marker='o', color='red', label='45deg poly')
plt.plot(Months, Yield_BIPV, marker='o', color='#ee3a8c', label='BIPV')
plt.plot(Months, Yield_GREY,  marker='o', color='k', label='BIPV GREY')
plt.ylabel('Spesific Yield [kWh/kWp]', fontsize=18)
plt.yticks(fontsize=18)
plt.grid(axis='y')
plt.xticks(rotation=45, fontsize=16)
plt.legend(fontsize=16)
plt.show()
```

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Mar 24 10:48:13 2021
@author: Sigrid
"""
#code to plot annual yield for all eight PV modules over a duration of
#12 months
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#Mono
#import M2 data
MonoM2 = pd.read_excel('Data_from_DB_M2_all.xlsx')
Mono_M2 = MonoM2.loc[0:255902,['PowerDC']]
Wh_mono_e = np.sum(Mono_M2)*(1/60)
Yield_mono_e = Wh_mono_e/315

MonoU2 = pd.read_excel('Data_from_DB_U2_all.xlsx')
Mono_U2 = MonoU2.loc[0:256712,['PowerDC']]
Wh_mono_w = np.sum(Mono_U2)*(1/60)
Yield_mono_w = Wh_mono_w/315

#Poly
PolyC3 = pd.read_excel('Data_from_DB_C3_all.xlsx')
Poly_C3 = PolyC3.loc[0:256770,['PowerDC']]
Wh_poly_e = np.sum(Poly_C3)*(1/60)
Yield_poly_e = Wh_poly_e/270

PolyD3 = pd.read_excel('Data_from_DB_D3_all.xlsx')
Poly_D3 = PolyD3.loc[0:251643,['PowerDC']]
Wh_poly_w = np.sum(Poly_D3)*(1/60)
Yield_poly_w = Wh_poly_w/270
#------------------------------------------------------------------------------
#45deg
Mar = pd.read_excel('Filtered_Mar.xlsx')
Apr = pd.read_excel('Filtered_Apr.xlsx')
May = pd.read_excel('Filtered_May.xlsx')
Jun = pd.read_excel('Filtered_Jun.xlsx')
Jul = pd.read_excel('Filtered_Jul.xlsx')
Aug = pd.read_excel('Filtered_Aug.xlsx')
Sep = pd.read_excel('Filtered_Sep.xlsx')
Oct = pd.read_excel('Filtered_Oct.xlsx')
Nov = pd.read_excel('Filtered_Nov.xlsx')
Dec = pd.read_excel('Filtered_Dec.xlsx')
Jan = pd.read_excel('Filtered_Jan.xlsx')
Feb = pd.read_excel('Filtered_Feb.xlsx')

Mar = Mar.dropna()
Apr = Apr.dropna()
May = May.dropna()
Jun = Jun.dropna()
Jul = Jul.dropna()
Aug = Aug.dropna()
Sep = Sep.dropna()
Oct = Oct.dropna()
Nov = Nov.dropna()
```

```
Dec = Dec.dropna()
Jan = Jan.dropna()
Feb = Feb.dropna()

Mono_mar = Mar.loc[:,[' Pmax_M4 ']]
Mono_apr = Apr.loc[:,[' Pmax_M4 ']]
Mono_may = May.loc[:,[' Pmax_M4 ']]
Mono_jun = Jun.loc[:,[' Pmax_M4 ']]
Mono_jul = Jul.loc[:,[' Pmax_M4 ']]
Mono_aug = Aug.loc[:,[' Pmax_M4 ']]
Mono_sep = Sep.loc[:,[' Pmax_M4 ']]
Mono_oct = Oct.loc[:,[' Pmax_M4 ']]
Mono_nov = Nov.loc[:,[' Pmax_M4 ']]
Mono_dec = Dec.loc[:,[' Pmax_M4 ']]
Mono_jan = Jan.loc[:,[' Pmax_M4 ']]
Mono_feb = Feb.loc[:,[' Pmax_M4 ']]

Mono45 = pd.concat([Mono_mar, Mono_apr, Mono_may, Mono_jun, Mono_jul,
                    Mono_aug, Mono_sep, Mono_oct, Mono_nov, Mono_dec,
                    Mono_jan, Mono_feb])

Poly_mar = Mar.loc[:,[' Pmax_M6']]
Poly_apr = Apr.loc[:,[' Pmax_M6']]
Poly_may = May.loc[:,[' Pmax_M6']]
Poly_jun = Jun.loc[:,[' Pmax_M6']]
Poly_jul = Jul.loc[:,[' Pmax_M6']]
Poly_aug = Aug.loc[:,[' Pmax_M6']]
Poly_sep = Sep.loc[:,[' Pmax_M6']]
Poly_oct = Oct.loc[:,[' Pmax_M6']]
Poly_nov = Nov.loc[:,[' Pmax_M6']]
Poly_dec = Dec.loc[:,[' Pmax_M6']]
Poly_jan = Jan.loc[:,[' Pmax_M6']]
Poly_feb = Feb.loc[:,[' Pmax_M6']]

Poly45 = pd.concat([Poly_mar, Poly_apr, Poly_may, Poly_jun, Poly_jul,
                    Poly_aug, Poly_sep, Poly_oct, Poly_nov, Poly_dec,
                    Poly_jan, Poly_feb])

Mono45_sum = np.sum(Mono45)*(1/60)
Poly45_sum = np.sum(Poly45)*(1/60)

Yield45_mono = Mono45_sum/315
Yield45_poly = Poly45_sum/270

#BIPV
BIPV_mar = Mar.loc[:,[' Pmax_M2 ']]
BIPV_apr = Apr.loc[:,[' Pmax_M2 ']]
BIPV_may = May.loc[:,[' Pmax_M2 ']]
BIPV_jun = Jun.loc[:,[' Pmax_M2 ']]
BIPV_jul = Jul.loc[:,[' Pmax_M2 ']]
BIPV_aug = Aug.loc[:,[' Pmax_M2 ']]
BIPV_sep = Sep.loc[:,[' Pmax_M2 ']]
BIPV_oct = Oct.loc[:,[' Pmax_M2 ']]
BIPV_nov = Nov.loc[:,[' Pmax_M2 ']]
BIPV_dec = Dec.loc[:,[' Pmax_M2 ']]
BIPV_jan = Jan.loc[:,[' Pmax_M2 ']]
BIPV_feb = Feb.loc[:,[' Pmax_M2 ']]
```

```
BIPV = pd.concat([BIPV_mar, BIPV_apr, BIPV_may, BIPV_jun, BIPV_jul,
                  BIPV_aug, BIPV_sep, BIPV_oct, BIPV_nov, BIPV_dec,
                  BIPV_jan, BIPV_feb])

BIPV_sum = np.sum(BIPV)*(1/60)
Yield_BIPV = BIPV_sum/218.1

#Collecting form grey BIPV
GREY_mar = Mar.loc[:,[' Pmax_M1 ']]
GREY_apr = Apr.loc[:,[' Pmax_M1 ']]
GREY_may = May.loc[:,[' Pmax_M1 ']]
GREY_jun = Jun.loc[:,[' Pmax_M1 ']]
GREY_jul = Jul.loc[:,[' Pmax_M1 ']]
GREY_aug = Aug.loc[:,[' Pmax_M1 ']]
GREY_sep = Sep.loc[:,[' Pmax_M1 ']]
GREY_oct = Oct.loc[:,[' Pmax_M1 ']]
GREY_nov = Nov.loc[:,[' Pmax_M1 ']]
GREY_dec = Dec.loc[:,[' Pmax_M1 ']]
GREY_jan = Jan.loc[:,[' Pmax_M1 ']]
GREY_feb = Feb.loc[:,[' Pmax_M1 ']]

GREY = pd.concat([GREY_mar, GREY_apr, GREY_may, GREY_jun, GREY_jul,
                  GREY_aug, GREY_sep, GREY_oct, GREY_nov, GREY_dec,
                  GREY_jan, GREY_feb])

GREY_sum = np.sum(GREY)*(1/60)

Yield_GREY = GREY_sum/193
#------------------------------------------------------------------------------
barWidth=0.4
Name = np.array(['Mono 10deg E', 'Mono 10deg W', 'Poly 10deg E', 'Poly 10deg W', '45deg Mono',
                 '45deg Poly', 'BIPV', 'BIPV GREY'])

Yield = pd.concat([Yield_mono_e,  Yield_mono_w, Yield_poly_e,
                   Yield_poly_w, Yield45_mono, Yield45_poly,
                   Yield_BIPV, Yield_GREY])

fig, ax = plt.subplots()
bar_plot = plt.bar(Name, Yield, width=barWidth, color='r')
plt.ylabel('Spesific Yield [kWh/kWp]', fontsize=18)
plt.yticks(fontsize=18)
plt.xticks( fontsize=16)
plt.ylim(ymax=1300, ymin=0)
plt.grid(axis='y')
ax.set_axisbelow(True)

bar_label = [920.10, 962.70, 969.10, 984.03, 988.11, 1049.93,
             761.99, 759.80]

def autolabel(rects):
    for idx,rect in enumerate(bar_plot):
        height = rect.get_height()
        ax.text(rect.get_x() + rect.get_width()/2., 1.05*height,
                bar_label[idx],
                ha='center', va='bottom', rotation=0, fontsize=12, color='r')
```

```
autolabel(bar_plot)
plt.show()
```