UiA University of Agder

Power Prediction of Photovoltaic System using Neural Network Models

CAMILLA LIE

SUPERVISORS

Sathyajith Mathew

Joao Leal

Anne Gerd Imenes

University of Agder, 2019

Faculty of Engineering and Science

Department of Engineering Science

ABSTRACT

This work considers a photovoltaic (PV) system installed on the rooftop of Agder Energi's headquarters located at Kjøita, Kristiansand. The system includes three different types of solar PV modules; Suntech (multi-Si), Sharp (a-Si/µ-Si) and REC (multi-Si), that have a total installed DC capacity of 45 kW_p. The system is grid-connected and instrumented for research and monitoring purposes. Artificial Neural Network (ANN) models were trained to obtain the lowest mean square error (MSE), by testing different configurations using a model-based trial and error approach. The model configurations that gave the lowest (MSE) were used to predict the power production from each of the PV modules using forecasted weather parameters obtained from MEPS (MetCoOp Ensemble Prediction System), with a one-day ahead and two-days ahead forecast horizon.

The input selection of the models was based on both model-free and model-based approach, where the final input selection resulted in global horizontal irradiance, wind speed, air temperature and air mass, with the power (AC) production as output (target). The results indicated that the model configurations of 20 hidden neurons in first hidden layer, and 2 hidden neurons in second hidden layer gave the lowest MSE for all PV modules. Results from the test sets showed that the best model for Suntech gave MSE = 0.0454, Sharp gave MSE = 0.0325 and REC gave MSE = 0.0492. R^2-values between 0.95 and 0.96 were obtained for all three models, indicating good fitting of the predicted values and the targets. Testing the Suntech and REC models with a hold-out set provided slightly less precise predictions compared to the results from the test set, while a higher precision was found for Sharp modules. Testing the model configurations with forecasted weather parameters indicated that the forecast accuracy of the weather will influence the power prediction, and the performance parameters will be accordingly. The one-day ahead forecasts provided MSE equal to 0.2647, 0.2378 and 0.2647, and for the two-days ahead forecast horizon an MSE equal to 0.2996, 0.2252 and 0.2719 for Suntech, Sharp and REC, respectively. An error much higher compared to the test set and hold-out set for the models, which inevitably was expected due to the weather forecast uncertainties.

Based on the findings in this work, it can be concluded that a further optimization of the models will be necessary before obtaining even more precise predictions. However, the models did show good fitting for several days and a potential for using ANN models for power prediction of PV modules.

University of Agder, Grimstad, May 24, 2019

Camilla Lie

Camilla Lie

PREFACE

This thesis is a result of the project in the master course ENE500, which concludes my two year long Renewable Energy Engineering Masters education at the University of Agder (UiA) in Grimstad, Norway. I have previously finished a bachelor's degree in Renewable Energy at the Norwegian University of Science and Technology (NTNU) in Gjøvik, Norway.

The master's thesis corresponds to 30 study credits, which corresponds to a time period between 07.01.2019 and 24.05.2019. This thesis considers power prediction of a photovoltaic (PV) system located in Kjøita, Kristiansand, instrumented with PV modules from three different manufacturers. ANN models based on feedforward neural networks have been used to train three models corresponding to each of the PV modules, in order to predict AC power production. An increase of grid-connected PV systems has been seen each year, introducing a necessity to accurately predict power production from PV systems, as this can influence the stability of the electricity grid.

My interest in solar power started early in my bachelor's course and has continued to grow as I have had several projects related to this subject throughout both the bachelor's and master's degree. Although I did not have any knowledge regarding Artificial Neural Network (ANN) when I started this thesis, I have come a long way in a short amount of time. Learning how to develop an ANN model was time consuming and a lot of trial and error was needed to understand the concept behind it. Having a large dataset with millions of samples was also challenging to handle, especially when combining the data with the ANN models.

I wish to thank my supervisors Sathyajith Mathew, Joao Leal and Anne Gerd Imenes at UiA, for guidance, encouragement and useful inputs regarding my thesis. They provided data from the relevant PV system that allowed me to do this work. I also want to thank Ghali Yakoub for providing historical forecasted data from MEPS (MetCoOp Ensemble Prediction System), which allowed me to test the models with actual forecasted data. Last, I want to thank family and friends for support and encouragement throughout the entire period.

INDIVIDUAL/GROUP MANDATORY DECLARATION

The individual student or group of students is responsible for the use of legal tools, guidelines for using these and rules on source usage. The statement will make the students aware of their responsibilities and the consequences of cheating. Missing statement does not release students from their responsibility.

1.	I/We hereby declare that my/our thesis 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 another exam at another department/university/university college in Norway or abroad; - does not refer to the work of others without it being stated; - does not refer to own previous work without it being stated; - have all the references given in the literature list; - is not a copy, duplicate or copy of another's work or manuscript.	☒
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

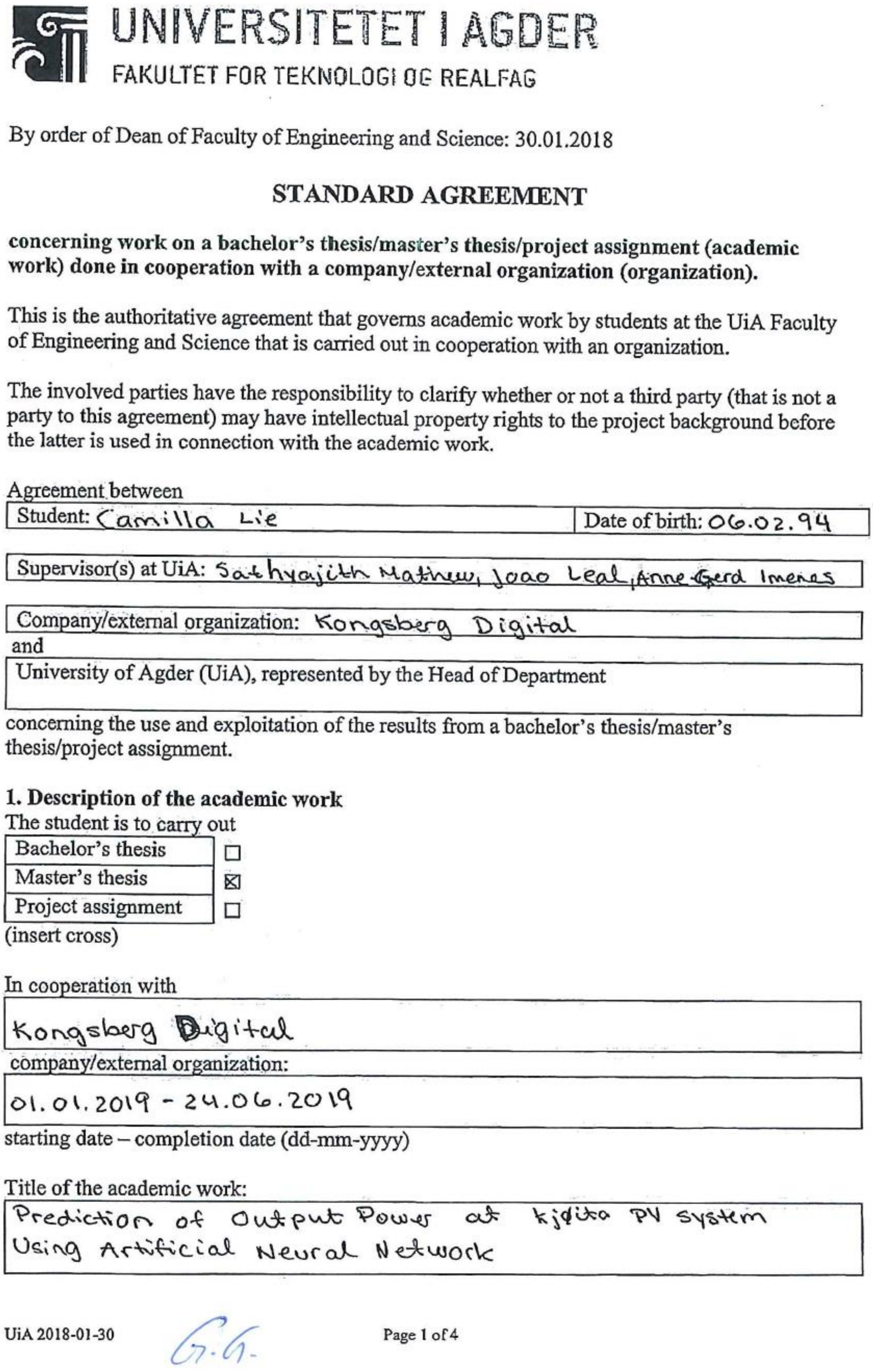
(confidential agreement must be completed)

- 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/Fall. §13)



UNIVERSITETET I AGDER
FAKULTET FOR TEKNOLOGI OG REALFAG

By order of Dean of Faculty of Engineering and Science: 30.01.2018

STANDARD AGREEMENT

concerning work on a bachelor's thesis/master's thesis/project assignment (academic work) done in cooperation with a company/external organization (organization).

This is the authoritative agreement that governs academic work by students at the UiA Faculty of Engineering and Science that is carried out in cooperation with an organization.

The involved parties have the responsibility to clarify whether or not a third party (that is not a party to this agreement) may have intellectual property rights to the project background before the latter is used in connection with the academic work.

Agreement between

Student: Camilla Lie | Date of birth: 06.02.94

Supervisor(s) at UiA: Sathyajith Mathew, Joao Leal, Anne Gerd Imenes

Company/external organization: Kongsberg Digital

and

University of Agder (UiA), represented by the Head of Department

concerning the use and exploitation of the results from a bachelor's thesis/master's thesis/project assignment.

1. Description of the academic work

The student is to carry out

Bachelor's thesis	□
Master's thesis	☒
Project assignment	□

(insert cross)

In cooperation with

Kongsberg Digital

company/external organization:

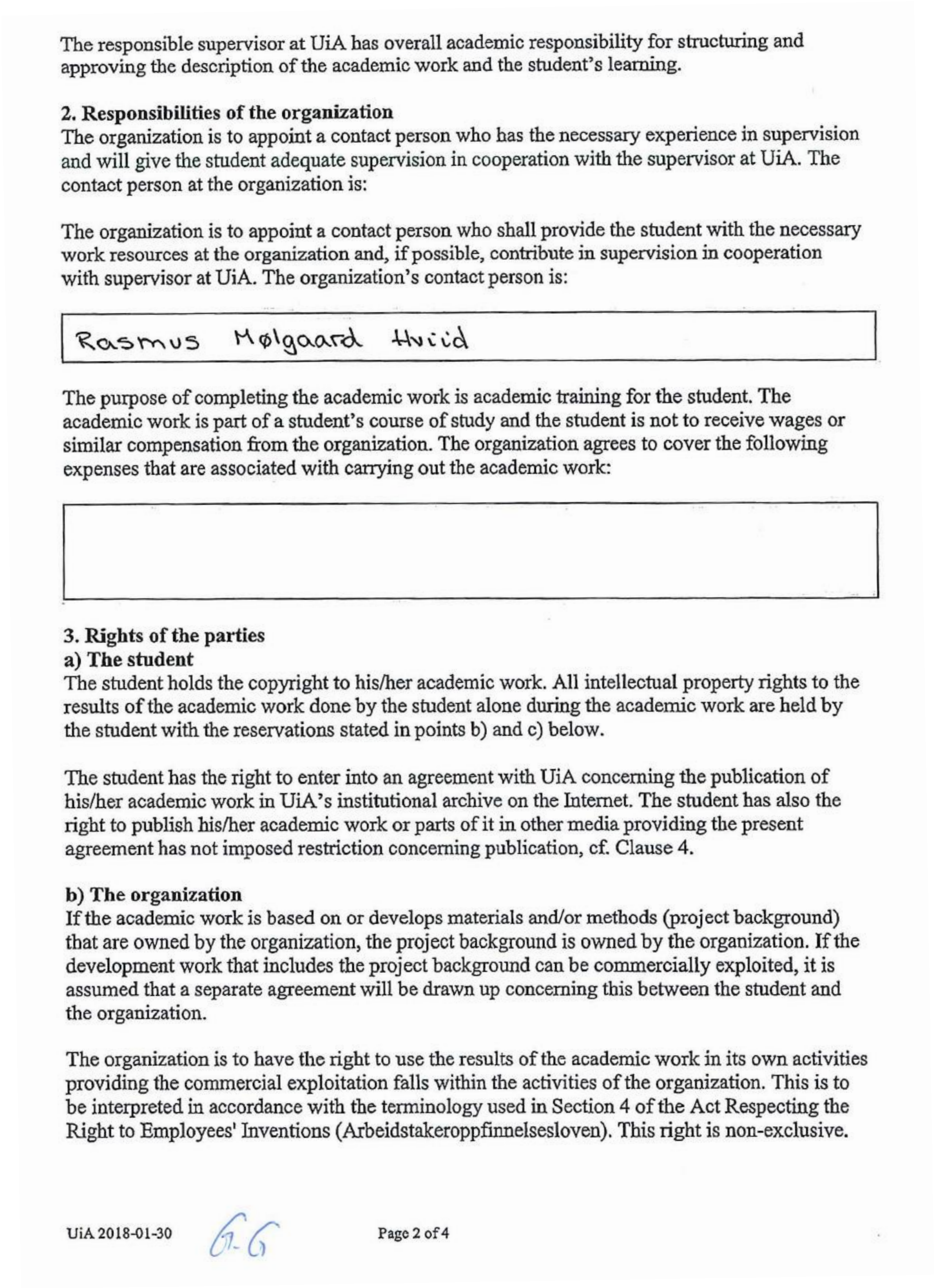
01.01.2019 - 24.06.2019

starting date – completion date (dd-mm-yyyy)

Title of the academic work:

Prediction of Output Power at kjøita PV system
Using Artificial Neural Network

UiA 2018-01-30 G.G. Page 1 of 4



The responsible supervisor at UiA has overall academic responsibility for structuring and approving the description of the academic work and the student's learning.

2. Responsibilities of the organization
The organization is to appoint a contact person who has the necessary experience in supervision and will give the student adequate supervision in cooperation with the supervisor at UiA. The contact person at the organization is:

The organization is to appoint a contact person who shall provide the student with the necessary work resources at the organization and, if possible, contribute in supervision in cooperation with supervisor at UiA. The organization's contact person is:

Rasmus Mølgaard Hviid

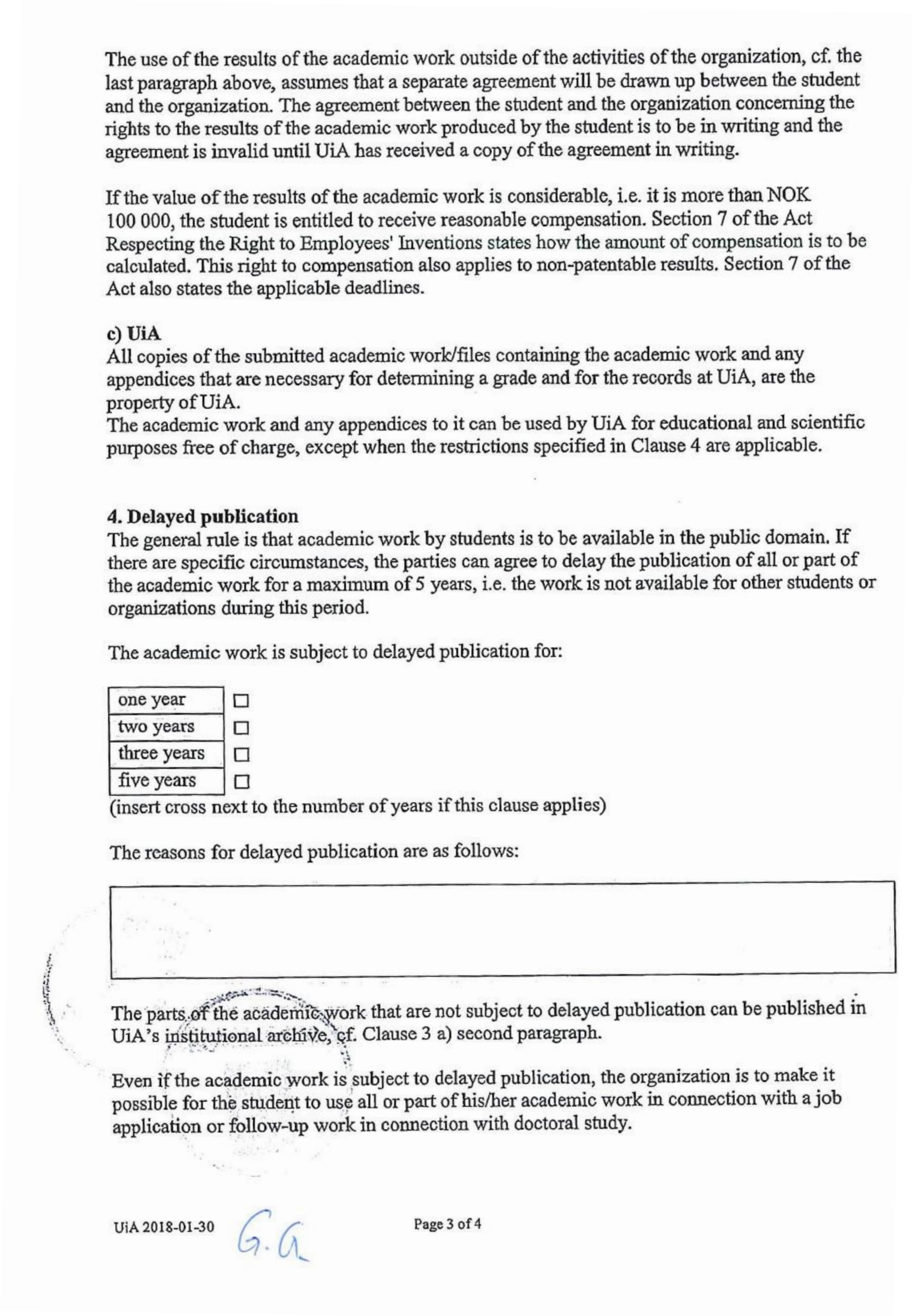
The purpose of completing the academic work is academic training for the student. The academic work is part of a student's course of study and the student is not to receive wages or similar compensation from the organization. The organization agrees to cover the following expenses that are associated with carrying out the academic work:

3. Rights of the parties
a) The student
The student holds the copyright to his/her academic work. All intellectual property rights to the results of the academic work done by the student alone during the academic work are held by the student with the reservations stated in points b) and c) below.

The student has the right to enter into an agreement with UiA concerning the publication of his/her academic work in UiA's institutional archive on the Internet. The student has also the right to publish his/her academic work or parts of it in other media providing the present agreement has not imposed restriction concerning publication, cf. Clause 4.

b) The organization
If the academic work is based on or develops materials and/or methods (project background) that are owned by the organization, the project background is owned by the organization. If the development work that includes the project background can be commercially exploited, it is assumed that a separate agreement will be drawn up concerning this between the student and the organization.

The organization is to have the right to use the results of the academic work in its own activities providing the commercial exploitation falls within the activities of the organization. This is to be interpreted in accordance with the terminology used in Section 4 of the Act Respecting the Right to Employees' Inventions (Arbeidstakeroppfinnelsesloven). This right is non-exclusive.



The use of the results of the academic work outside of the activities of the organization, cf. the last paragraph above, assumes that a separate agreement will be drawn up between the student and the organization. The agreement between the student and the organization concerning the rights to the results of the academic work produced by the student is to be in writing and the agreement is invalid until UiA has received a copy of the agreement in writing.

If the value of the results of the academic work is considerable, i.e. it is more than NOK 100 000, the student is entitled to receive reasonable compensation. Section 7 of the Act Respecting the Right to Employees' Inventions states how the amount of compensation is to be calculated. This right to compensation also applies to non-patentable results. Section 7 of the Act also states the applicable deadlines.

c) UiA
All copies of the submitted academic work/files containing the academic work and any appendices that are necessary for determining a grade and for the records at UiA, are the property of UiA.
The academic work and any appendices to it can be used by UiA for educational and scientific purposes free of charge, except when the restrictions specified in Clause 4 are applicable.

4. Delayed publication
The general rule is that academic work by students is to be available in the public domain. If there are specific circumstances, the parties can agree to delay the publication of all or part of the academic work for a maximum of 5 years, i.e. the work is not available for other students or organizations during this period.

The academic work is subject to delayed publication for:

one year	□
two years	□
three years	□
five years	□

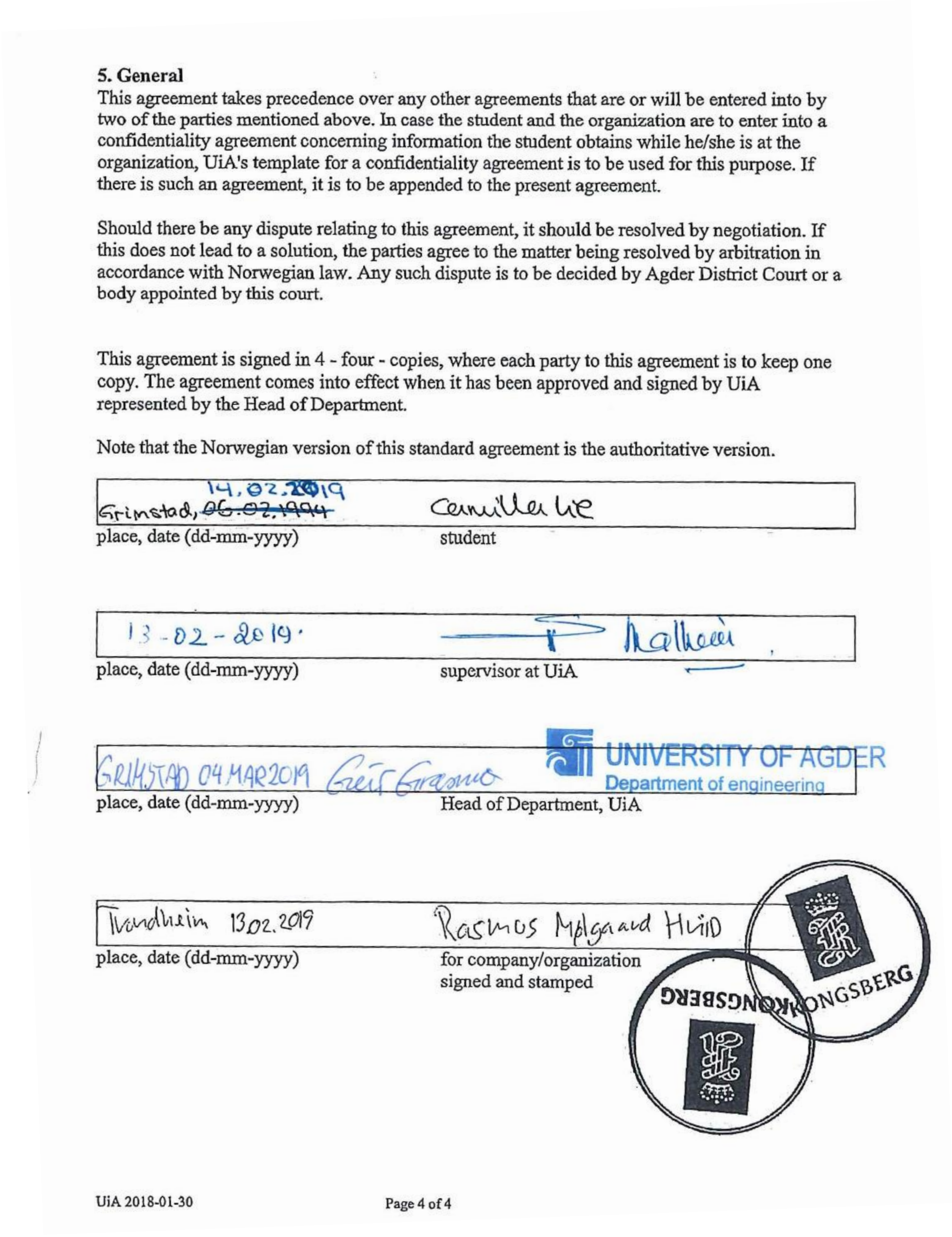
(insert cross next to the number of years if this clause applies)

The reasons for delayed publication are as follows:

The parts of the academic work that are not subject to delayed publication can be published in UiA's institutional archive, cf. Clause 3 a) second paragraph.

Even if the academic work is subject to delayed publication, the organization is to make it possible for the student to use all or part of his/her academic work in connection with a job application or follow-up work in connection with doctoral study.

UiA 2018-01-30 G.G Page 3 of 4



5. General

This agreement takes precedence over any other agreements that are or will be entered into by two of the parties mentioned above. In case the student and the organization are to enter into a confidentiality agreement concerning information the student obtains while he/she is at the organization, UiA's template for a confidentiality agreement is to be used for this purpose. If there is such an agreement, it is to be appended to the present agreement.

Should there be any dispute relating to this agreement, it should be resolved by negotiation. If this does not lead to a solution, the parties agree to the matter being resolved by arbitration in accordance with Norwegian law. Any such dispute is to be decided by Agder District Court or a body appointed by this court.

This agreement is signed in 4 - four - copies, where each party to this agreement is to keep one copy. The agreement comes into effect when it has been approved and signed by UiA represented by the Head of Department.

Note that the Norwegian version of this standard agreement is the authoritative version.

Grimstad, ~~06.02.1994~~ 14.02.2019 Camilla Lie
place, date (dd-mm-yyyy) student

13-02-2019.
place, date (dd-mm-yyyy) supervisor at UiA

GRIMSTAD 04 MAR 2019 Geir Grasmo
UNIVERSITY OF AGDER
Department of engineering
place, date (dd-mm-yyyy) Head of Department, UiA

Trondheim 13.02.2019 Rasmus Mølgaard Huid
place, date (dd-mm-yyyy) for company/organization
signed and stamped

KONGSBERG
KONGSBERG

UiA 2018-01-30 Page 4 of 4

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Symbols

A	Input variable
α	Alpha
D	Day of year
Δ	Delta
ε	Error
G_{SC}	Solar constant
H	Altitude
I	Identity matrix
I_{SC}	Short circuit current
je	Jacobian error
jj	Levenberg-Marquardt
jX	Jacobian matrix
K	Kelvin
mu	Adaptive value
N	Number of variables
n	Model input
Ω	Omega
Φ	Phi
P_{mp}	Maximum power
s	Second
Θ	Theta
t	Output variable
T	Temperature
V_{OC}	Open circuit voltage
W	Weights
Z	Scaled score

ABBREVIATIONS

AC	Alternating current
AI	Artificial intelligence
AM	Air mass
AMI	Average mutual information
ANN	Artificial neural network
AT	Air temperature
CV	Coefficient of variation
DC	Direct current
DHI	Diffuse horizontal irradiance
ECMWF	European Centre for Medium-Range Weather Forecasts
EU	European Union
FFNN	Feedforward neural network
GHI	Global horizontal irradiance
GTI	Global tilted irradiance
LR	Linear regression
MAE	Mean absolute error
MARE	Mean absolute relative error
MBE	Mean bias error
MEPS	MetCoOp Ensemble Prediction System
ML	Machine learning
MSE	Mean square error
NN	Neural network
PCA	Principal component analysis
PV	Photovoltaic
RNN	Recurrent neural network
SLAF	Scaled Lagged Average Forecasting
TEK	The building code
UiA	University of Agder
UTC	Coordinated Universal Time
WS	Wind speed

1. INTRODUCTION

As an increase of grid-connected photovoltaic (PV) systems can be seen over the last few last years, having accurate forecasts for the power production fed into the grid has become more of an important issue. In 2018, approximately 23.5 MW$_p$ of PV system capacity was installed in Norway, which was an increase of 29% compared to 2017. Globally, a total capacity of PV installed at the end of 2017 was approximately 400 GW$_p$, accounting for about 2% of the total power production worldwide. For the last years, the amount of installed PV capacity in Norway has increased from 9 MW$_p$ to 45 MW$_p$ (at the end of 2017) and are expected to continue to rise for upcoming years. The reason for an increase of PV systems is primarily because of the reduction of the investment cost, which decreased with 10-20% from 2015 to 2017, but also factors such as incentives, TEK-regulations (regulations on technical requirements for building works) and EU directives have played a role. As this increase have been expected to continue for years ahead, the grid-connected PV systems will lead to higher changes in the electricity grid and can create instabilities due to sudden changes in weather [1].

For grid operators to be able to handle sudden changes of power in the grid, accurate predictions of output power from PV systems can contribute to reveal important information to regulate the electricity grid more efficiently. Variation in solar irradiance due to weather fluctuations causes variations in the power production from PV systems and as the use of large-scale grid connected PV system also is increasing, it will be important to strengthen the prediction of PV system output power. Artificial neural network (ANN) has the advantage to overcome limitations of traditional methods to solve complex problems that are difficult to model and analyze, and is therefore viewed as a convenient method to forecast solar radiation intensity and power output of PV systems. ANNs have become a widely used method in the forecasting field, with the aim to find the most accurate approach [2].

ANNs are non-linear computational models that can be used for a variety of problems, including classification, speech recognition, clustering, prediction and forecasting. Recent research has shown that ANNs have powerful pattern classification and recognition capabilities as they are able to learn and generalize from experience. They learn from examples and even with unknown or hard to describe underlying relationships, ANNs can capture these subtle functional relationships. Having enough data or observations allow for solving problems whose solutions require knowledge that is difficult to specify. The ANNs ability to generalize allows for data containing noisy information to be correctly inferred and are capable of performing non-linear modeling without a priori knowledge about an input-output variable relationship. Because of this, ANNs are seen as a more general and flexible modeling tool for forecasting [3].

1.1. PROBLEM DEFINITION

This work considers Artificial Neural Network (ANN) models using input variables from historical measured data from a PV system, to predict power production at any time of the year. Three different types of PV modules are being studied, whereas also three different ANN models will be trained and tested, each model corresponding to each of the PV modules. Having an insight to which configurations that give the best performance parameters, i.e. lowest mean square error (MSE), these models will be used with historical forecasted weather data from a forecasting service.

Based on this, the following goal was established:

- Potential of using ANN models for PV power prediction and their application to power forecast based on MetCoOp Ensemble Prediction System (MEPS).

Further on, it was also established three sub goals:

- Determine the relevant input parameters that should be used in the ANN models.
- Find the configuration (number of hidden neurons and layers) that provides the best model performance.
- Evaluate how the ANN models perform with a hold-out dataset and forecasted weather parameters coming from MEPS.

The three sub goals listed above will thus contribute to reach the main goal of this study. In the methodology section (section 3) the methodology for achieving these goals is presented.

1.2. Limitations and Assumptions

To be able to achieve the goal of this work, several limitations and assumptions were made along the way. The main limitations and assumptions made in this work will be presented in this subsection, while a more explanatory description will be stated in the methodology section in this work.

As there was little previous knowledge regarding certain aspects of this work, the process or reaching the goal was mainly based on a trial and error approach. Data obtained from the PV system was recorded on minute-basis, but for an Artificial Neural Network (ANN) model to work with forecasted weather data obtained from MEPS (MetCoOp Ensemble Prediction System), which was obtained on hourly basis, it was decided to also use the on-site data with hourly values. As the data contained samples with missing data and/or data that could have been compromised, it was desirable to filter out this data. This will be considered data-manipulation, which again can affect the models' performance. As there was a maintenance log for the system, this was used to filter out data that had been compromised because of maintenance, software updates or similar. Additional data-manipulation was not performed in this work. Due to this, if any compromised data not stated in the maintenance log, could at some extent influence the ANN model performance and careful considerations regarding results from the data samples must be taken.

An ANN model was chosen to be used as a prediction model for power production. This was chosen based on the models' ability of generalization and literature review that supports the use of this type of model for non-linear prediction systems. How ANNs work will be explained throughout the report and a literature review will also be presented.

For the ANN model to be used with forecasted weather parameters, it was necessary to find the optimal model configurations. For this, a model-based trial and error approach was used. To limit the extent of the model testing, seven tests with a variation of hidden layers and hidden neurons in the network were considered. It was created three different ANN models, one for each of the PV modules. The seven tests included a variation with a low number and a relatively high number of neurons in one hidden layer, where also some tests included a second hidden layer. As mentioned above, there was no previous experience from training such models and a random composition of hidden neurons and layers were chosen. Before the seven tests could be performed, certain configurations were tested to see which would provide a model with better performance. The mean square error (MSE) was used as the performance parameter to decide which model configuration that indicated the most precise model. Other performance parameters were also considered and discussed, but the choice of the most optimal model out of the seven tests were chosen from the lowest MSE value. This parameter is the default performance parameter in MATLAB's toolbox, which was used for developing and testing the ANN models.

A preliminary report related to this thesis done by Lie [4], characterized parameters from the PV system at Kjøita, Kristiansand. Due to lack of information regarding the measured parameters on-site, the results would be considered uncertain because of vague pre-processing of the raw data. It was, however, uncovered some weaknesses in the data, i.e. the DC power production was lower than the AC power production. This was considered a large deviation, and the DC power will for this work be considered non-valid due to sensors uncertainties. Thus, the AC power will be used.

If other limitations or assumptions were made throughout this work in order to reach the goal, this will be specified in the methodology section.

1.3. Thesis Structure

The structure of this thesis involves a theoretical background that will include a brief literature review related to topics in this thesis. The literature review will give an indication of previously done research related to this work and give an understanding of main contributors that is important to consider when developing ANN models for PV systems. The theoretical background will cover aspects that is considered important to have insight about before reading the methodology, results and discussion sections.

The methodology section will have information based on the theoretical background and a further description regarding assumptions and the progress in this work will be presented. Important decisions such as input selection, data division (dividing data into subsets for ANN model training purposes), model structure and performance will be presented, completed with a discussion of the method choice for this work.

The result and discussion section will present and illustrate the objective of this work together with a discussion regarding the results. The results will present the steps from obtaining the raw data, to the results from the final testing of the different ANN models where a discussion regarding performance parameters will be of importance. Finally, a conclusion will be presented followed by suggestions for further work.

2. THEORETICAL BACKGROUND

2.1. LITERATURE REVIEW

A review of photovoltaic power forecasting by Antonanzas et al. [5] from 2016, assesses different techniques and approaches to improve the accuracy and reduce uncertainty in prediction models. The review is considering the latest advancements and with it follows a summary of the main techniques used to issue the predictions along with a discussion regarding the benefits of the different techniques. The review concludes that ANNs are the most commonly used machine learning techniques among solar power forecasting as they have proven useful in a wide variety of situations and with many inputs variables. The next most used techniques are the support vector machines that use supervised modeling method. They are strong when it comes to generalization capacity and have a great ability to deal with non-linear problems. As the most common model of PV forecasting is based on ANNs, the parameters and the ANN method approach can vary. Reviewing several publications, frequent variables used in models were the global horizontal irradiance (GHI), temperature (ambient) and power (historical data) (e.g. in [6, 7]).

Omar et al. [8] used artificial neural networks (ANNs) ensembles to predict power output with input variables global horizontal irradiance, wind speed, air temperature, pressure, humidity cloud cover and time of year and day. Different numbers of hidden layers and hidden neurons were altered to see the difference in errors from the trained models. The results from this study showed that averaging the output forecasts from an ensemble of similar configuration networks were likely to perform better regarding a day-ahead forecasting than a single network of the same configurations.

Kudo et al. [6] suggested the use of normalized solar radiation when training an ANN network for solar power based on weather parameters. The weather varies for different seasons, and the use of only one season for a model would require large amount of data, it was suggested that the normalized radiation could give the model better performance. The normalized radiation was obtained by dividing the solar radiation with the extraterrestrial radiation.

A study by Liu et al. [9] was done to see the correlation between output power from a PV system with solar irradiance and air temperature. The output power indicated a linear correlation with the solar irradiance intensity, while the air temperature gave neither positive nor negative linear correlation, meaning the power output has a non-linear correlation with air temperature. The air temperature was therefore considered an important parameter that will influence the power output. As this study used several different input variables for solar radiation intensity and temperatures, the optimal architecture of the network was found to be 28-20-11, with results indicating good performance of forecasts of daily power output of the relevant PV system. Another study, done by Oudjana et al. [10], defines three models to train a feedforward neural network, involving different input variables. A model with only air temperature as input showed a large forecast error due to low correlation between air temperature and power output. A second model included only solar irradiance and provided an acceptable error. A third model included both air temperature and irradiance, which consequently also provided the best model of the three, with the lowest error.

Kaldellis et al. [11] discuss the temperature and wind speed impact on the efficiency of a PV installation. The findings show that the photovoltaic conversion process depend on the solar irradiance and the module temperature, where the module temperature is affected by the solar irradiance, air temperature and ventilation (wind speed), as well as the technical characteristics of the PV panels. The study highlights the importance of the wind speed impact and that it plays a fundamental role in determining the modules' temperature and accordingly their efficiency. The study concludes that determination of the wind's effect on the thermal loss mechanisms of PV panels are close, but not identical to those in PV simulations and literature.

Defining a network depends on several configurations. In a study done by Sulaiman et al. [12], investigate the performance of a multi-layer feedforward neural network (MLFNN) technique compared with a linear regression (LR) model. For the neural network technique, a configuration with Levenberg-Marquardt activation function, 18 hidden neurons in one layer using *logsig* as activation function and *purelin* as activation function in output layer, was chosen as the best training parameters that provided the lowest error. The results show that the classes MFLNN had superior prediction performance compared to the LR during training and testing.

2.2. Solar Radiation

The solar radiation from the sun's surface (extraterrestrial radiation) is relatively constant throughout the year, often indicated as the solar constant (1367 W/m^2). But as is reaches the earth's surface, it has been affected by the earth's atmosphere due to absorption and scattering. On clear, cloud-free days, the maximum radiation reaching the earth's surface occurs as the sun is directly overhead, because of shorter pathlength through the atmosphere. Figure 1 illustrates the spectrum of solar radiation, where the black line represents the idealized radiation from the sun which can be seen as a blackbody with temperature at approximately 5778 K, the yellow area represents the radiation outside the atmosphere and the red area represents the radiation at sea level.

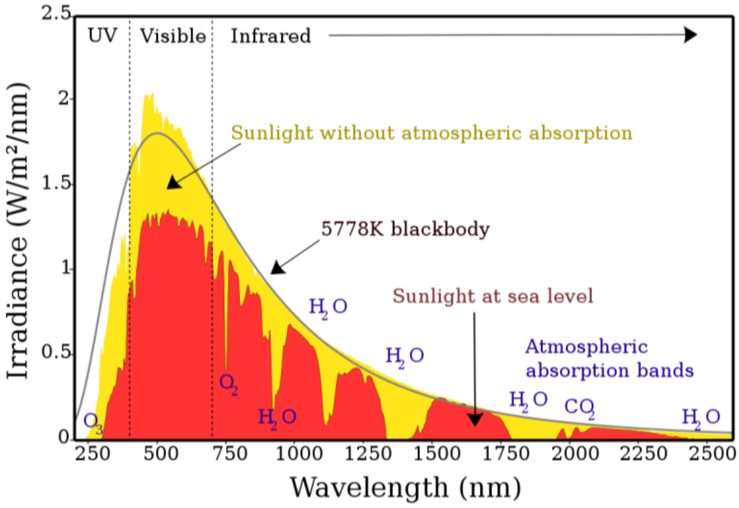


Figure 1: Solar radiation spectrum outside the atmosphere and at the earth's surface [13]

The red curve is varying due to absorption bands occurring in the atmosphere from various gases. Due to scattering of radiation in the atmosphere, the solar radiation reaching the earth's surface is not constant. This scattering is occurring due to molecules, aerosols and dust particles and scattered light is known as diffuse radiation. Diffuse radiation is typically 10% of the global radiation when the sun is directly overhead on a clear day. Figure 2 illustrates the direct and diffuse radiation reaching the earth's surface.

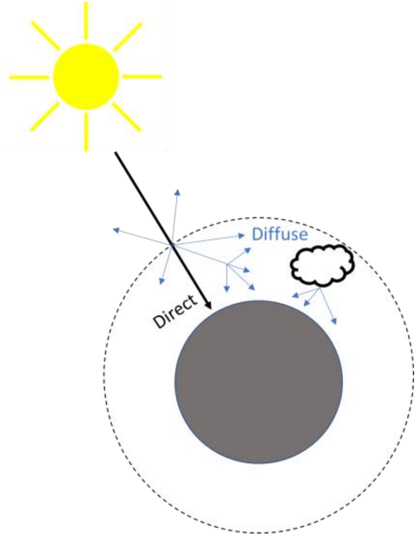


Figure 2: Direct and diffuse solar irradiance

- **Global Horizontal Irradiance:** the sum of the direct and diffuse irradiance emitted onto a horizontal surface [14, pp. 18].
- **Diffuse Horizontal Irradiance: r**adiation that is scattered or diffused by particles in the atmosphere or reflected by the earth's surface. It is measured on a horizontal surface where the direct sunlight is being blocked by a ball or disc by using e.g. a pyranometer [14, pp. 18].
- **Global Tilted Irradiance:** Measuring the irradiance in plane with a tilted PV module gives the global tilted irradiance, which corresponds to the same amount of irradiance that the PV modules receive [14, pp. 21].

2.3. Air Mass (AM)

The pathlength the sunlight travels through the atmosphere to reach the earth's surface is referred to as the air mass. The air mass varies throughout the day, depending on the location of the sun relative to the earth. Equation (1) can be used to calculate the air mass based on the assumption of a homogenous, non-refractive atmosphere. This equation introduces an error of approximately 10% close to the horizon (when the sun's elevation angle is 10° above the horizon). To account for this error, equation (2) can be used, which takes the variation of atmospheric density into account.

$$AM = \frac{1}{cos(\theta_Z)} \tag{1}$$

$$AM = \frac{e^{-0.0001184*h}}{cos(\theta_Z) + 0.5057 * (96.080 + \theta_Z)^{-1.634}} \tag{2}$$

where h is the site altitude [m] and Θ_Z is the zenith angle [°]. This angle is a good approximation down to ≈ 10°. As the sun is directly overhead, the air mass is at its lowest, meaning more sunlight reaches through the atmosphere. As the location of the sun moves towards the horizon, the zenith angle, Θ_z, becomes higher, hence the air mass increases. Figure 3 shows the amount of atmosphere (air mass) the radiation from the sun must pass through to reach the earth's surface and how it depends on the sun's position on the sky.

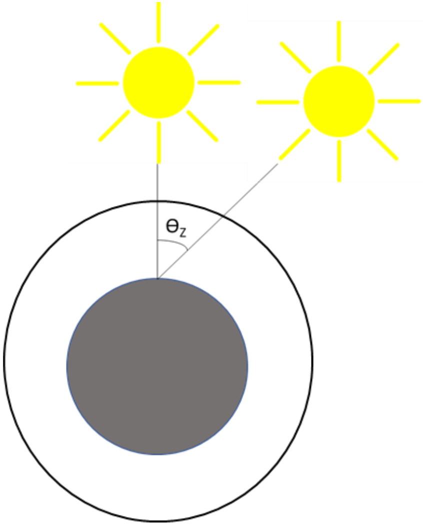


Figure 3: Air mass

The zenith angle can be calculated using equation (3).

$$\cos(\theta_Z) = \cos(\phi)\cos(\delta)\cos(\omega) + \sin(\phi)\sin(\delta) \tag{3}$$

where ϕ is the latitude angle, δ is the solar declination angle and ω is the hour angle.

The declination angle (δ) can be calculated by using equation (4).

$$\delta = 23.45° * \sin\left(360° * \left(\frac{284 + D}{365}\right)\right) \tag{4}$$

where *D* is the day of the year. To calculate the hour angle (ω), equation (5) can be used.

$$\sin(\mathrm{w}) = \frac{\sin(\alpha)\ - \sin(\delta)\sin(\phi)}{\cos(\delta)\cos(\phi)} \tag{5}$$

where α is the altitude angle.

To get the altitude angle, α, equation (6) can be used.

$$\sin(\alpha) = \sin(\delta)\sin(\phi) + \cos(\delta)\cos(\omega)\cos(\phi) \tag{6}$$

2.4. Photovoltaic System

Individual solar cells electrically connected together is called a PV module, and are done so to increase their power output. The cells are packed so that they are protected from the environment and to protect the used from electric shock. The effects that are considered the most important in PV modules or arrays are [15]:

- losses due to mismatched solar cells interconnections
- module temperature
- PV modules failure modes

The top surface material of the cells should be impervious to water, good impact resistance, stable under prolonged UV exposure and low thermal resistivity. In addition, reflection from the front surface should be low. To reduce this reflection, applying anti-reflection coating to the top surface can be done, but an issue with the robustness of these coatings are to be found. A technique that can be used instead, is to "roughen" or texture the surface. However, this technique is more likely to attach dust and dirt to the top layer of the modules, thus less "self-cleaning", where the advantage of less reflection is outweighed by losses from top surface soiling [15].

A representation of how the different terms from PV cells to arrays can be seen in Figure 4.

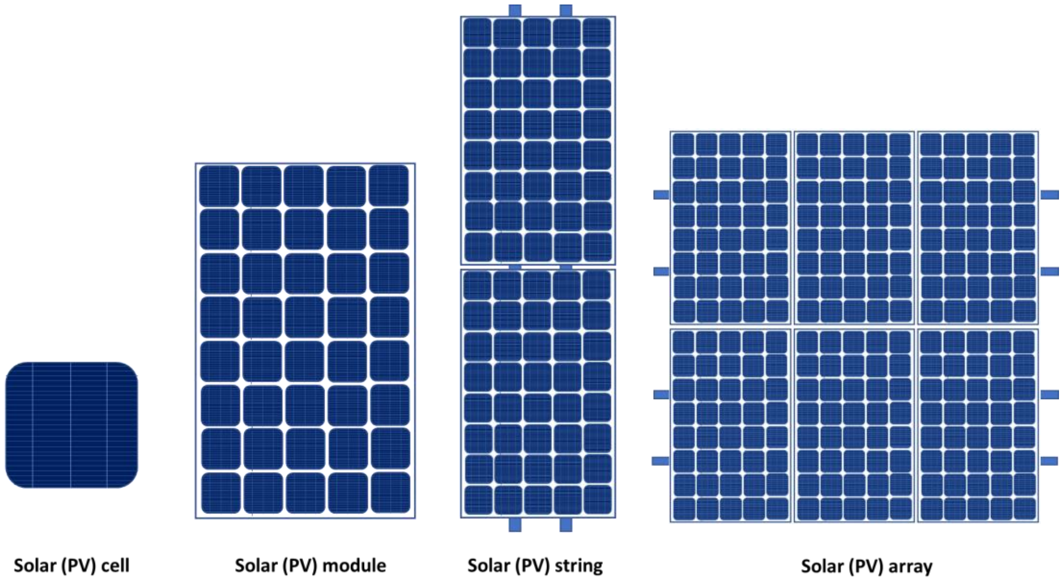


Figure 4: From solar cell to solar array

Several PV cells make up a PV module, several modules connected in series is referred to as a PV string and several PV modules (strings) make up a PV array. By having several PV arrays connected togheter and into a house or the electricity grid, make up a PV system. A simplified grid-connected PV-system is illustrated in Figure 5.

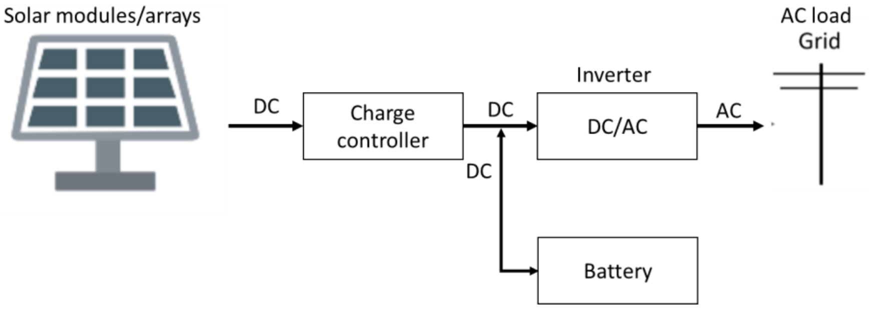


Figure 5: Simplified schematic of grid-connected PV system

Figure 5 illustrates how solar (PV) modules or arrays are connected to a charge controller that again is connected to a DC/AC inverter and/or batteries. Solar cells produce direct current (DC), which needs to be converted into alternating current (AC) if it is to be fed to the electricity grid. DC energy can be stored in a battery or used in e.g. home appliances. The charge controller with MPPT (maximum power point tracker) or PWM (pulse width modulator) is typically used, and is needed if a battery is connected to the system. The charge controller determines if the power generated from the solar modules is needed to charge the batteries, used in home appliances or fed to the grid. The controller is preventing the battery from overcharging and makes sure the electricity stored in the battery does not to get back to the solar modules during no-production periods [14, pp. 99-105].

2.5. Solar Cell Materials

According to Basore's [16] definition, there are three different material structures divided by crystal grain size; crystalline, multi-/polycrystalline and amorphous. The three different material structures are illustrated in Figure 6 and represents the arrangements of the atoms for each material. The following descriptions of the materials are based on Wenham et al. [14, pp. 31-33].

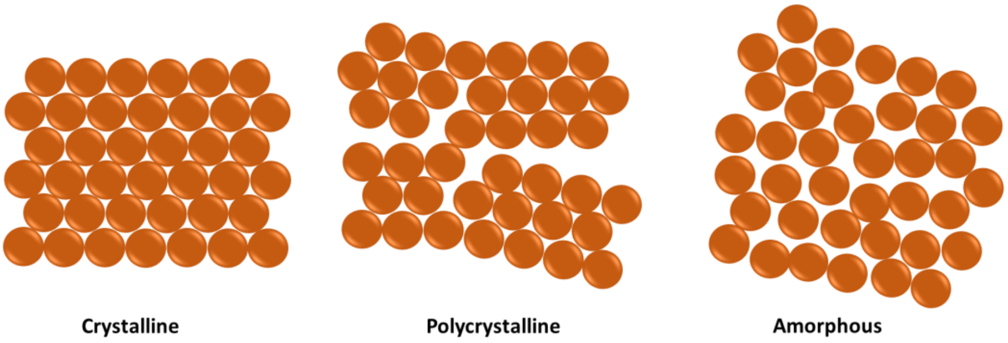


Figure 6: Material structures

2.5.1. Crystalline Silicon

Crystalline silicon includes atoms arranged in a regular pattern with an ordered crystal structure. The manufacturing process for this technology is a careful and time-consuming process, making it the most expensive type of silicon. The crystalline silicone usually has the highest quality but because of the cost, multi/polycrystalline and amorphous silicon are increasingly being used.

2.5.2. Multi/polycrystalline Silicon (multi-Si)

Multicrystalline or polycrystalline materials require a grain size on the order of a few millimeters to avoid significant recombination losses. The production technique for this material is less critical than for single crystal material, resulting in a cheaper type of material. This material has a lower quality than the crystalline material due to the grain boundaries blocking carrier flow, which results in a greater recombination loss.

2.5.3. Amorphous Silicon (a-Si/µ-Si)

In this type of material, there is no long-range order in the atom arrangement structure, which will result in a more challenging way to obtain reasonable current flows in a PV cell configuration. The band gap for amorphous silicon is 1.7 eV, while in crystalline silicone has a 1.1 eV band gap, and the absorption coefficient of a-Si is much higher than that of crystalline silicon. a-Si is commonly called "thin film", due to having very thin films of semiconductor that are deposited onto glass or other low-cost substrates and are often used in calculators and watches. The material is cheaper to produce than the crystalline material but in return the efficiency and lifetime is lower.

2.6. Solar Cell Output

Photovoltaic (PV) cells convert sunlight directly into electricity with the use of semiconductors. According to Wenham et al. [14, pp. 53] single-junction solar cells have a theoretical efficiency limit around 30%. Conventional solar cells are made of crystalline silicone because of high efficiency. These cell types can have an efficiency up to 24-25% under laboratory conditions and state-of-the-art technology, while commercially, mass-produced cells have an efficiency typically around 13-19% [14, pp. 53].

The output of a PV module depends on several different weather conditions and operational state of the cells. Weather conditions such as irradiance and cell temperature have proven to influence the output power, where also variables like wind velocity, humidity and pressure influence the irradiance and cell temperature. The module temperature has the effect on the cell output in such a way that an increase of temperature, results in a decrease in cell output. The effect of temperature on the maximum power output (P_{mp}) for silicon is represented with equation (7), given by Wenham et al. [14, pp. 47].

$$\frac{1}{P_{mp}}\frac{dP_{mp}}{dT} = -(0.004{\sim}0.005)°C^{-1} \tag{7}$$

Where P_{mp} is the maximum power and T is the cell temperature.

Figure 7 represents the typical effect of increasing temperature of a solar cell.

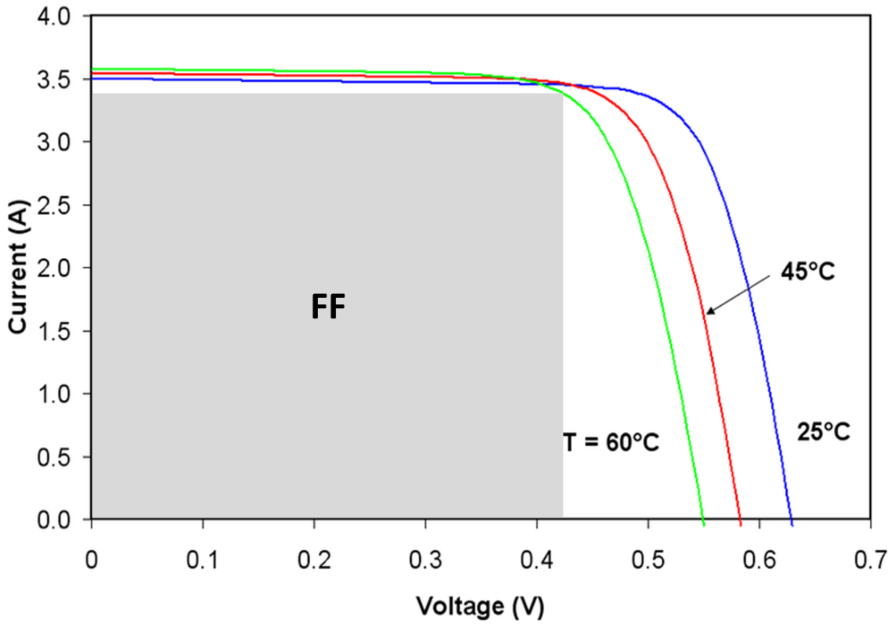


Figure 7: The effect of temperature on the I-V characteristics of a solar cell [17]

As seen from Figure 7, an increase of temperature has the effect of increasing the I_{SC} and decreasing the V_{OC}, ultimately reduce the fill factor (FF), hence the reduce the cell output. The fill factor is the ratio of the maximum power (P_{mp}) from the solar cell to the product of the open circuit voltage (V_{OC}) and short circuit current (I_{SC}) [14, pp. 44], see equation (8).

$$FF = \frac{P_{mp}}{V_{OC} * I_{SC}} \tag{8}$$

Other aspects that can affect the cell output are the resistances (parasitic series and shunt). Contributors to the series resistance can be the bulk resistance on the semiconductor material, metallic contacts and interconnections. The shunt resistance is due to impurities near the junction and p-n junction non-idealities. Aspects such as faults/failures, dirt, shading and degradation will also have a negative effect on the output of a solar cell. For even more information of the behavior of solar cells, Wenham et al. [14] can be reviewed.

2.7. ARTIFICIAL INTELLIGENCE (AI)

Understanding intelligence and building intelligent systems are definite goals of artificial intelligence (AI). The definition, however, is widely discussed and agreeing on an accurate description is challenging. Many have tried to come up with a reasonable definition that covers all aspects, but weaknesses can still be found in the definitions. Elain Rich came up with the definition;

"Artificial Intelligence is the study of how to make computers do things at which, at the moment, people are better." [18]

This definition characterizes what AI researches have been doing for the last 50 years and will even be up to date in year 2050, as stated by Ertel [19, pp. 2].

AI is a broad branch of computer science, with the goal of a system that can function intelligently and independently. It can widely be divided into two main areas; symbolic learning and statistical learning [20, pp. 31]. The symbolic learning includes computer visions and robotics, where image processing is used. Machine learning uses pattern recognition for either statistical learning or deep learning. Statistical learning deals with techniques for speech recognition and natural language processing. There are different types of deep learning in machines, which are essentially different techniques to replicate what the human brain does. For the machine learning area, it is essential with data for the machine to be able to learn [19]. To see an overview of how the three areas are connected, see Figure 8, that shows that deep learning is part of machine learning, that again is a branch of artificial intelligence.

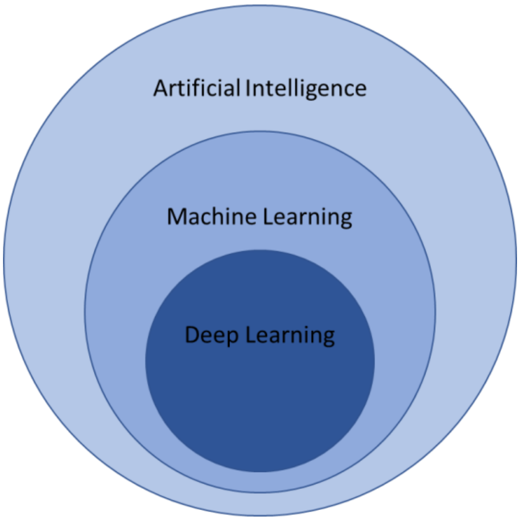


Figure 8: Subsets of Artificial Intelligence (AI)

2.8. Machine Learning

As seen in Figure 8, machine learning (ML) is part of artificial intelligence. By using example data or past experience, ML can be used to optimize a performance criterion using computational algorithms. Such algorithms turn empirical data into usable models [21]. A significant interest in machine learning is to develop efficient algorithms for designing models, to do analyses and predictions, where big data has become of particular importance. Big data refers to large amounts of data with high dimensions, which will need algorithms to be computationally efficient and robust in their performance as the datasets can include noise and missing values [22].

Machine learning can either be shallow learning or deep learning, depending on their credit assignment paths or learning ability. These paths are possibly learnable chains of causal links between actions and effects. Usually, machine learning is divided into two main methods; supervised and unsupervised learning. Whether a model uses supervised or unsupervised learning, depends on the available data. For the supervised model the data used for training is labeled, while for unsupervised learning, the data is unlabeled. Labeled data refers to a dataset that includes attributes (sorted data within categories), observations of each attribute and features (targets). As unsupervised models do not have labelled data, they will have to understand the data by for example using clustering techniques. Figure 9 provides an overview of machine learning methods [23].

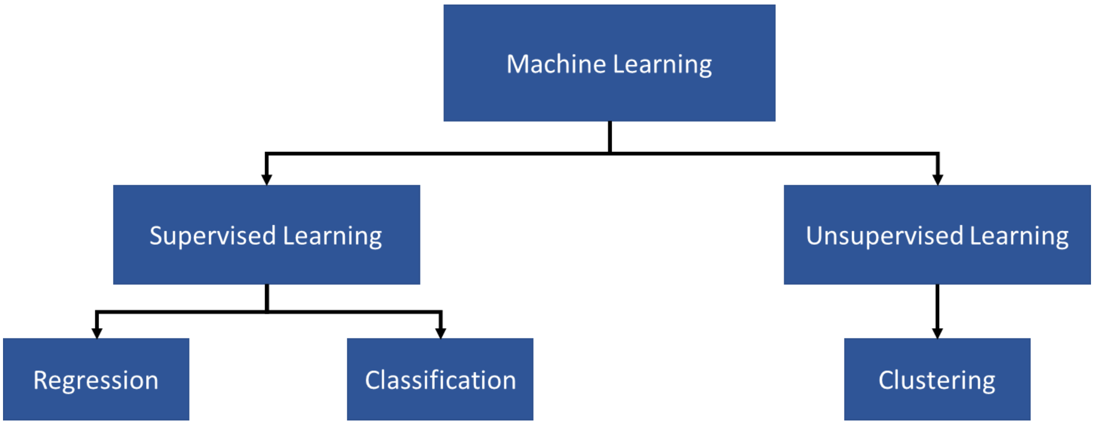


Figure 9: Machine learning scheme

Supervised learning establishes a learning process as it continuously adjusts the predictive model by comparing the predicted results with the actual results, until the predicted results of the model reach an expected performance. For a supervised model, both the input and output data is known, but the relationship between the variables may be unknown [24, pp. 5-9]. A common goal with supervised neural network training is to find weights that yield lower values of error that is being calculated from the model. Within supervised learning there are two methods, classification and regression [22].

Unsupervised learning has only knowledge about the input data, so the data will be grouped or interpreted based only on input data. It encodes raw data in a form that is more convenient for subsequent goal-directed learning and can be connected to topics of regularization and compression [24]. Clustering is a multivariate statistical technique aiming to find pattern in any shape in a large dataset by collecting similar individuals of homogeneous classes [25].

2.8.1. Classification

Techniques that uses classification predicts categorical responses by classifying the input data into categories [23]. A classification task involves assigning an unknown pattern to a number of known classes. By having a rule that fits the past data and which can also fit new data, predictions can be made for novel instances [21].

2.8.2. Regression

A regression task can be considered a curve fitting problem, where the output variable is a non-discrete variable that takes values in an interval in the real axis or in a region in the complex numbers plane. Having a data set including training points of y_i and x_i, where i can represent any real number higher than 1, it is possible to estimate a function f, whose graph fits the data [22]. If a new data point with an unknown output value occurs, the function from the trained model can be used to predict its output value. Figure 10 illustrates a simple linear regression example where a new point is being predicted with the function that has been designed to fit the available training data set.

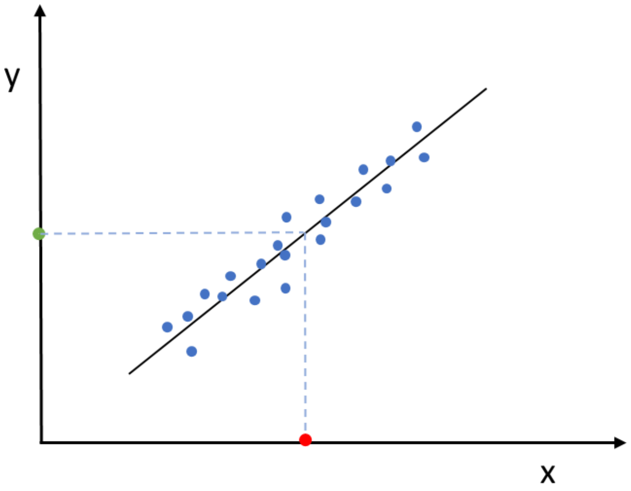


Figure 10: Linear regression example

The blue points in Figure 10 represents the training data and the red points on the x-axis represents a new point to the data set. The green point is the new output value $\hat{y}$, predicted from the trained model with an equation equal to (9).

$$\hat{y} = f(x) \tag{9}$$

where $\hat{y}$ is the output values, and *f(x)* is a function derived from a trained model [22, pp. 3-4].

2.9. ARTIFICIAL NEURAL NETWORKS (ANNs)

ANNs consist of different layers connected to each other and work on the structure and functions of a human brain. They can perform complex algorithms to train a neural network by learning from huge volumes of data.

A standard neural network (NN) consists of many neurons, which is a connected processor that produce a sequence of real-valued activations. Input neurons activate through sensors perceiving information from the environment, other neurons get activated by weighting connections from previous neurons. Depending on the connection of the neurons, the wanted output, or the desired behavior of the network, long causal chains of computational stages may be required [24].

2.9.1. Neural Network Terms

Neuron (node/perceptron): A neuron is the basic unit of a neural network. It receives a certain amount of inputs and a bias value, where the input value is multiplied with a weight value. Each neuron is connected to another neuron, which each connection has an associated weight value.

Transfer function (activation function): The transfer function (also called activation function) defines the output of a neuron in terms of the induced local field. Commonly used transfer functions are represented in Figure 11. The functions are determined whether the local field is above or below zero, calculating a layer's output from its net input.

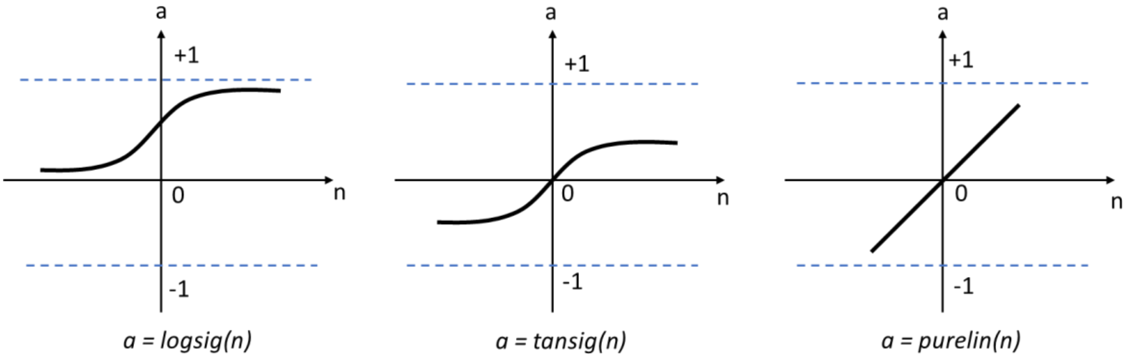


Figure 11: Transfer functions. a = output, n = input

The *logsig*-function is a Log-Sigmoid transfer function that generates output between 0 and 1 as the neuron's input goes from zero to positive infinity. The *tansig*-function is Tan-Sigmoid transfer function, which takes both negative and positive input values and the *purelin*-function is a linear transfer function. The sigmoid functions are most commonly used in pattern recognition, while the linear function is used for function fitting problems [26].

Backpropagation: The backpropagation algorithm is used to calculate a networks performance with respect to weights and bias variables. The algorithm is often used on a validation set of a neural network model (this will be explained in more detail in following subsections) and can be used to stop the network training if the errors calculated by the backpropagation algorithm fails to improve. The predicted value is compared to the target value, and a loss function is used to calculate the error between the two points. The average of the loss functions of the entire training set is called the cost function and is measured by for example the mean square error (MSE). A commonly used backpropagation is the Levenberg-Marquardt *jj*, which uses the Jacobian *jX,* and can be calculated based on:

$$jj = jX * jX$$

$$je = jX * \varepsilon$$

$$dX = \frac{-(jj + I * mu)}{je} \tag{10}$$

where *jX* is the Jacobian matrix, *ε* is all the errors, *I* is the identity matrix and *mu* is the adaptive value [27].

Weights: Weights will be assigned to each neuron where the goal of training a model is to update the weight values to decrease the loss (error). The weights represent the strengths of connection between neurons and decide how much the input will influence the output. If the weight has a value of zero, this input will not influence the output at all. An increasing positive weight value will increase the output, while a negative weight will decrease the output. How fast the network learns is based on the learning rate, which is the rate at which the weights are being updated. For each new weight update using the backpropagation, which is the same as one iteration (epoch), the learning rate will decide the new weight update.

Epoch: Also called iteration, is the term used for each weight-update. For each epoch, the weight assigned to the connections are updated with the use of backpropagation. The backpropagation algorithm will decide the weight update and based on wanted outcome of the model (i.e. performance of the model) and continue to iterate until the goal of the model performance is reached.

2.9.2. Feedforward Neural Network (FFNN)

A FFNN is structured with layers where information flows from the input layer, through hidden layer(s) to the output layer, without any loops or cycles in the network. However, depending on the number of layers of perceptrons, the network can be without a hidden layer, describing a single layer perceptron.

1. **Single Layer Perceptron**

A single layer perceptron contains only an input layer and an output layer and is the simplest form of neural network. A representation of the perceptron can be seen in Figure 12.

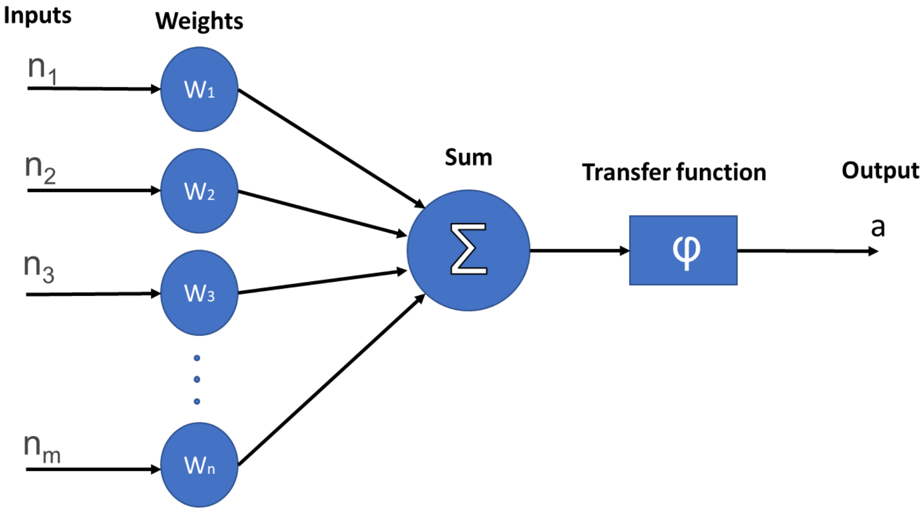


Figure 12: Nonlinear model of a perceptron (neuron)

The inputs in the perceptron (neuron) may come from the environment or be the outputs from other perceptrons. Each input is associated with a connecting weight, which are being summed and defined by an transfer function, before obtaining the output *a* [21, pp. 271].

2. **Multilayer Perceptron**

This class of ANN has one or more hidden layers with computation neurons. It is called a hidden layer because it is not seen directly either on the input side or the output side. The purpose of the hidden layers is to intervene in a useful manner between the external input and output of the model, by computing its own activation value. How many layers the network has determines the width of the model [28, pp. 165]. Each neuron in the hidden layers receives input from other units and computes a value that is being forwarded to the next layer. Figure 13 represents a network with two hidden layers.

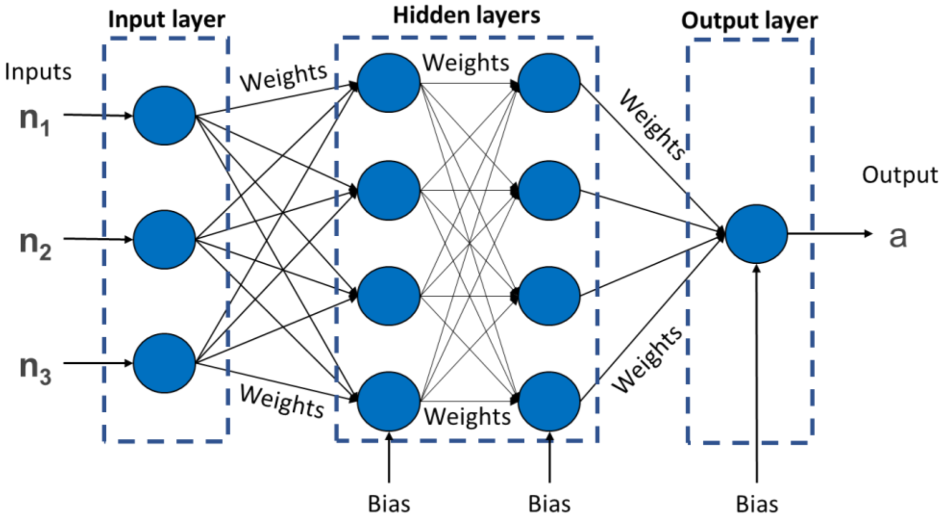


Figure 13: Multilayer feedforward neural network with two hidden layers

All inputs are each connected to the first hidden layer, but not to each other. The input then passes through the transfer function of the perceptrons (neuron) and are passed on to the next layer, until it reaches the output layer. The model learns by updating the weights (backpropagation). This algorithm helps the model to learn and update the weights in order to increase the efficiencies [21, pp. 279]. The network illustrated in Figure 13 can be referred to as a 3-4-4-1 network, as there are 3 inputs, 4 hidden neurons in first hidden layer and 4 in second, and 1 output.

In practice, it is more common to only use one hidden layer of neurons as a network with more than one layer makes the network quite complicated to analyze. But if a hidden layer contains many hidden neurons, it may be sensible to go to multiple hidden layers [21, pp. 281].

2.9.3. Recurrent Neural Network (RNN)

If a network includes feedback connections, it is called recurrent neural network (RNN). This type of network has self-connections or connections to neurons in the previous layers, in contrast to a feedforward network which only sends information in one direction. RNN acts as a short-term memory, allowing the network to remember what happened in the past [21, pp. 305-306]. The difference between a recurrent network and a feedforward network is illustrated with Figure 14.

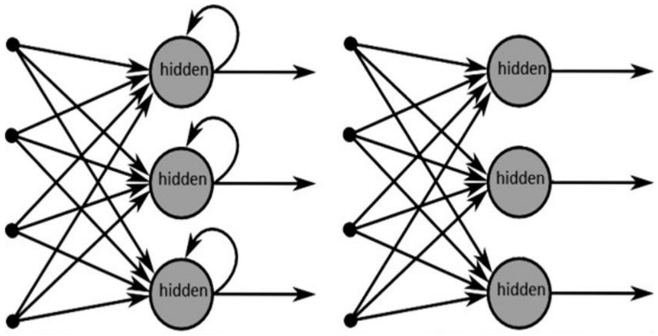


Figure 14: Recurrent (left) and feedforward (right) neural network [29]

2.10. ANN Model Development Process

There are several steps that can be followed in the process of developing an artificial neural network (ANN). Depending on the available data and what is desired to achieve from the model, the method can vary. In the following subsections, a brief description of the steps in developing an ANN model will be presented, based on Figure 15. The four first steps in Figure 15 can be considered the data-preprocessing part, and the three last can be considered dealing with the ANN models. The descriptions have been based on Maier et al. [30].

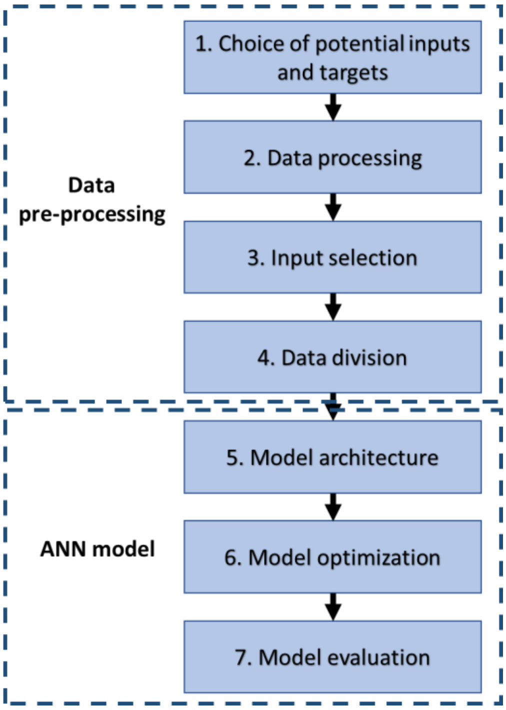


Figure 15: Steps in ANN model development process

According to Maier et al. [30], all ANN prediction models take the term in equation (11):

$$t = f(a, W) + \varepsilon \tag{11}$$

where *t* is the model output (target), *f(a,W)* is the functional relationship between model outputs, inputs and parameters, a is the model inputs, *W* is the connection weights and *ε* is the model error. *f(a,W)* will be governed by the model architecture and geometry (e.g. number of hidden layers and neurons, type of transfer function) and the parameters need to be defined.

2.10.1. Choice of Potential Inputs and Targets

Depending on available parameters/data, the input selection possibilities will be restricted. If a model is to be based on weather parameters, it is important to identify what parameters and how much data is available. The target (output) of the model is to be based on the input variables.

2.10.2. Data Processing

Having the input selection determined, processing of the data should be performed. This step is important to be certain that the training of the model is based on correct, unbiased data. This step includes gathering data and processing it before using it in an ANN model. It can also be necessary to scale the data that will be used as input variables, to avoid problems with assigning weights for data on different scales. This can be done by standardizing the data, allowing an input selection without extreme differences in maximum and minimum values.

2.10.3. Input Selection

This can be a model-free approach, which uses statistical measures of significant, or model-based approach, that choose the input variables based on the performance of the model. The first approach can use methods for dimensionality reduction, which can exclude variables providing redundant information because the variable is closely related to another input variable. This can increase the likelihood of overfitting (overtraining) [21, pp. 120]. The latter approach will involve training and evaluating the model several times to observe what provides the best performance of a trained model.

2.10.4. Data Division

Data division involves dividing the data samples into sub-sets for the training process. A data division usually involves three sets; a training, validation and test set. The training set will be used to train the model, or in other words, assign the connection weights. The validation set can be used as a stopping criterion (also called early stopping), so the model will stop altering the weights as a performance indicator have failed to improve. A test set will be used as an unseen data set for the model to test new data to observe the performance of the model. How this division is decided can be evaluated with the use of statistical properties, or by using random division. The latter case can make it difficult to compare the performance of different model configurations, as it is uncertain if the model is able to capture the relationship and generalize it, depending on different pattern in the validation set.

2.10.5. Model Architecture

The most popular architecture is the feedforward multilayer perceptron model, due to its complexity and ability to generalize data. To estimate the relationship between input and output variables, the model architecture has a significant impact.

2.10.6. Model Optimization and Evaluation

The optimization of the model can be divided into three main aspects:

I. Model structure selection
II. Model calibration
III. Performance evaluation

The model structure, along with the model architecture, defines the functional form (*f(a,W)* from equation (11)) of the relationship between the inputs and output(s) of the model. The optimal network structure can be obtained, generally by finding a balance between generalization ability and network complexity. Having too low complexity of the model, e.g. network size and number of free parameters, the network might have problems capturing the desired input-output relationship. However, if a too complex network structure is used, the generalization ability and processing speed might decrease, resulting in a model that is more difficult to calibrate and less transparent.

The calibration of the model has the aim to find a set of connection weights that represent the desired input-output relationship in a best way possible. Overfitting can be of an issue, but where optimal generalization ability is achieved when a suitable error measure between actual and predicted outputs in a test set is minimized. This applies if the data in the training and testing set are representative of the modelling domain.

Overfitting is a phenomenon when a model learns too many input-output examples, resulting in a network finding a feature (i.e. noisy data) in the training data, memorizing this feature that is not true of the underlying function that is to be modeled. Thus, a model will lose its ability to generalize between similar input-output relationship. Figure 16 illustrates the generalization ability of a model based on overfitting and underfitting [31, pp. 164-165].

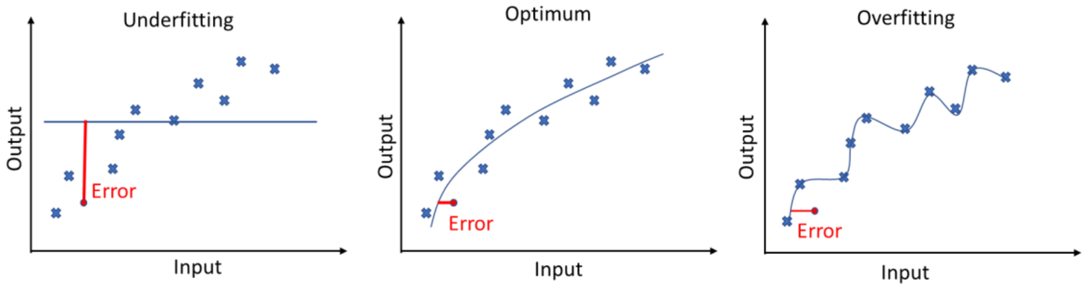


Figure 16: Overfitting and underfitting versus good generalization (optimum)

The performance evaluation of the model calibration can be assessed by evaluating one or more criteria. Maier et al. [30] suggests that if overfitting is a concern, the test data should be used for evaluating the model.

3. Methodology

This section explains the progress followed in this work. The case study for this work will first be presented, along with a brief description of the instrumentation at the system. For the Artificial Neural Network (ANN) models to be based on valid datapoints, various pre-processing techniques was used on the relevant variables. A feedforward neural network with backpropagation in MATLAB have been chosen to create a model for each of the PV-module types located at Kjøita, Kristiansand. As there are several weather parameters measured at the location, it will be relevant to investigate which of the variables are of importance to the model and if there are any lagging values related to these variables, worth considering. The method for creating a model will be a model-based approach for establishing an optimal number of hidden neurons and hidden layers, depending on performance parameters. The models are to be tested with forecasted weather parameters to observe the performance of the models by also including possible errors from weather forecast deviations. This will be elaborated in more details in the upcoming subsections.

3.1. Case Study – Kjøita PV System

The data was obtained from a research project at the headquarters of Agder Energi in Kristiansand (PV system owner) performed in collaboration with several partners including the University of Agder [32]. The PV modules are installed on top of Agder Energi's headquarter located in Kjøita, Kristiansand, south in Norway (lat. 58.154, long. 8.001 and altitude 20 m). The PV system consists of two different PV technologies with three different PV cells suppliers. The total system is illustrated in Figure 17 and key information regarding the PV modules relevant for this work can be found in Table 1.

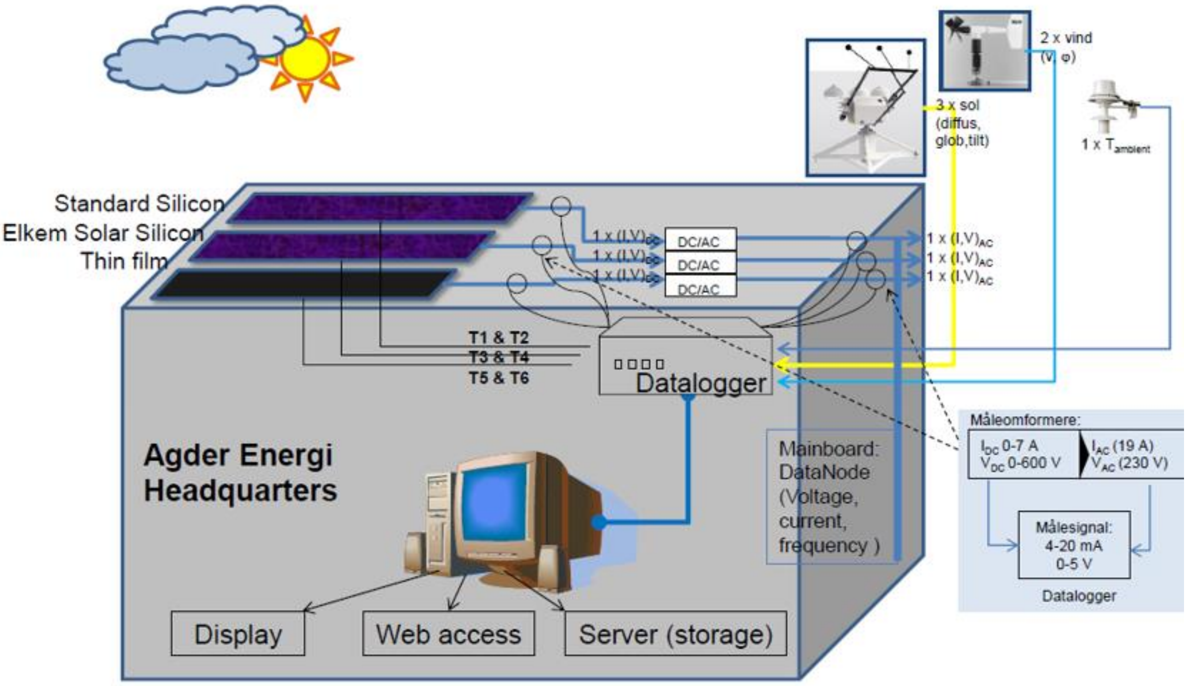


Figure 17: Instrumentation and data collection overview at Kjøita, Kristiansand (appendix A.1)

Table 1: Key information regarding PV system at Kjøita

PV module type	PV technology	Efficiency (η_{STC})	Installed power (kW_p)	Area (m^2)	No. modules
Suntech STP225-20/Wd	Poly-Si	13.6%	5.40	1.65	24
Sharp NA-F135 (G5)	a-Si	9.5%	2.43	1.42	18
REC 225PE	Poly-Si	13.6%	5.40	1.65	24

The three PV module types are connected to three different inverters, logging data separately. For the Suntech (of type Elkem Solar Silicone) and REC modules (of the traditional Siemens-process type), the inverters have two strings with 12 modules connected in each, while Sharp has 9 modules connected in each. At the same time as the inverter is logging data, sensors are also measuring the DC and AC current and voltage (see Figure 17). An overview of the parameters measured at Kjøita PV system can be found in Table 2 and an illustration of the PV modules on the roof of Agder Energy's headquarter can be found in Figure 18. Technical data for the PV modules can be found in appendix A.2.

Table 2: Parameters logged at Kjøita PV system

Parameter	Unit
Global horizontal irradiance	W/m^2
Diffuse horizontal irradiance	W/m^2
Global tilted irradiance	W/m^2
Wind speed	m/s
Wind direction	-
Air temperature	°C
Module temperature (mid- and end-of-array)	°C
Current (AC and DC)	A
Voltage (AC and DC)	V

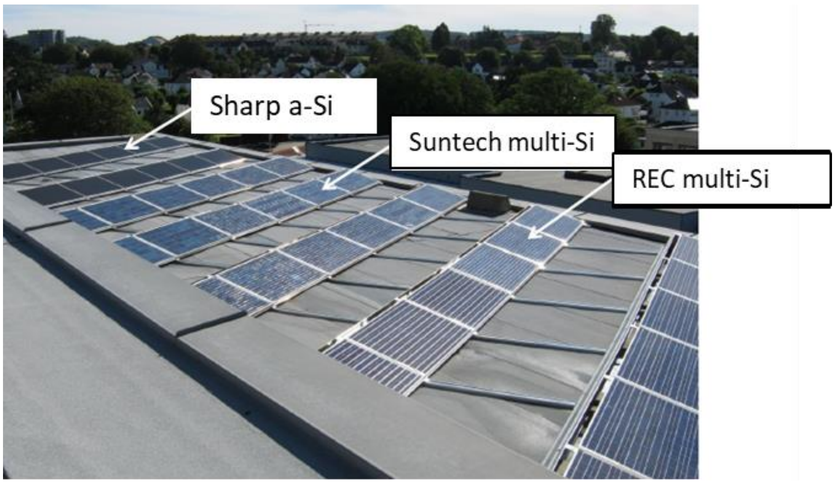


Figure 18: PV system on rooftop of Agder Energi's headquarter, Kjøita (appendix A.1)

The inverters installed are of type Eltek Theia HE-t series where a capacity of 4.6 kW$_p$ was installed for the m-Si modules Suntech and REC (with a maximum efficiency of 97.3%) and a capacity of 2.9 kW$_p$ for the Sharp thin film modules (maximum efficiency of 97%). See appendix A.3 for datasheet. A Campbell Scientific Datalogger CR1000 with AM416 Multiplexer is used to record the measurements in the datalogging system. The irradiation sensors used are 3 x Kipp&Zonen CMP 11 secondary standard pyranometers with CVF3 ventilation units, measuring global tilted irradiance (GTI). A Kipp&Zonen SOLYS2 two-axis tracker with shading ball assembly measure the global horizontal irradiance (GHI) and diffuse horizontal irradiance (DHI). Temperature sensors are mounted on the back of the modules. The sensors are six Campbell Scientific 110 PV surface temperature probes. There are installed 2 temperature sensors for each type of PV modules, one on the module at the end of the array and one on the module in the middle of the array. To measure the ambient temperature, wind speed and direction, an R.M. Young 5103 instrument from Campbell Scientific is installed. The DC and AC voltage and current for the three PV arrays are measured with transducers from LEM and Phoenix.

3.1.1. Solar Irradiance and Power Production Relationship On-Site

Power production from PV systems have proven to be linearly correlated with solar irradiance. Module temperature have also been proven to affect the efficiency of a solar cell, where parameters such as solar irradiance, air temperature and wind speed thus have an effect of the module temperature [9, 11]. Figure 19 illustrates the relationship between solar irradiance and power production for each of the PV modules for two random days.

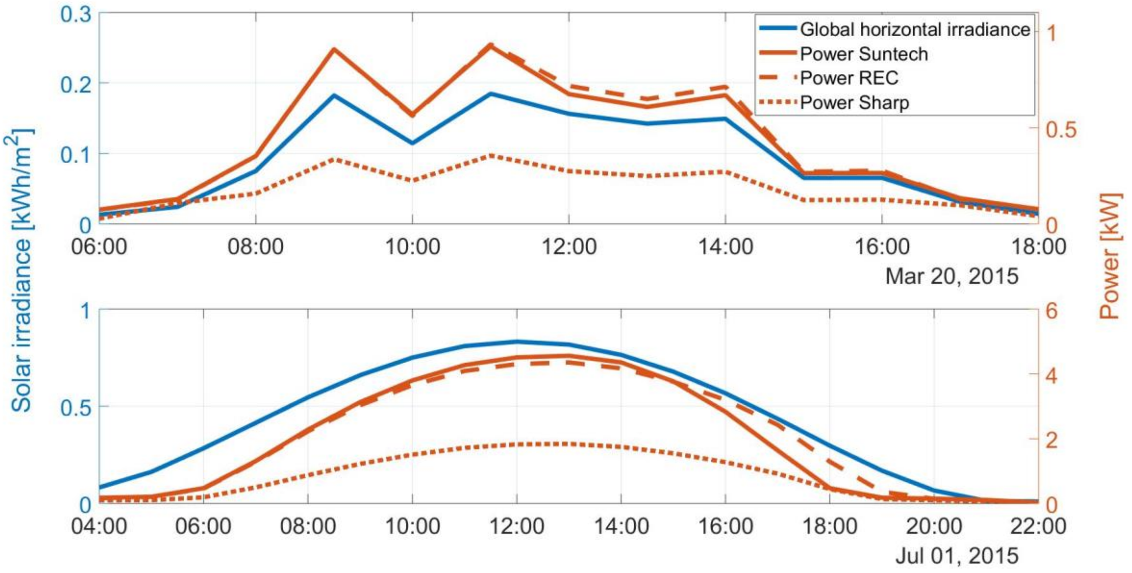


Figure 19: Relationship between solar irradiance and power production

The same trends (the behavior of irradiance related to the power) can be observed for all the PV modules and the solar irradiance. An increase of solar irradiance appears to result in an increase in power, and opposite, a decrease of irradiance corresponds to a decrease of power production. The power production from Sharp PV modules are naturally lower compared to Suntech and REC, due to lower installed power and number of modules installed.

3.2. FORECASTED WEATHER PARAMETERS

The forecasted weather information is gathered from MEPS (MetCoOp Ensemble Prediction System), which is an atmospheric ensemble model that covers Scandinavia and the Nordic seas. MEPS is the cooperation around numerical weather prediction between Sweden, Norway and Finland, established in 2011. The forecasts are updated every 6 hours daily, with three-hourly cycling for data assimilation and a horizontal resolution between 1 and 2.5 km and a forecast horizon of 67 hours. Boundary perturbations are generated from European Centre for Medium-Range Weather Forecasts (ECMWF) using the SLAF (Scaled Lagged Average Forecasting) method, which is a scaling method of the forecasts. The data is offered by the Norwegian meteorological institute and freely available to the public for use. For more detailed information regarding the forecasted weather information, it is recommended to review [33].

Data was only available from November 2016 as earlier forecasts were no longer saved in the database. The forecasted weather parameters available from MEPS that could be used for this purpose are presented in Table 3.

Table 3: Forecasted weather parameters from MEPS

Parameters	Unit
Accumulated downward surface shortwave radiation	J/m^2
Wind speed (10 m, x-direction)	m/s
Air temperature (10 m)	K

The accumulated downward surface shortwave radiation was processed in such a way that it was on hourly basis and provided the global horizontal irradiance [W/m^2]. The wind speed and air temperature were predicted at 10 m above the ground, whereas Kjøita PV system is mounted 20 m over the ground (roof of the building). Obstacles such as trees and buildings can cause turbulence and therefore vary by height above ground level. The wind speed is averaged over one hour, which was considered to provide a decent estimation of the wind speed at 20 m. The forecast update at 06:00 UTC for relevant days was obtained, thus a two-day ahead forecast horizon could be used. Difference in the time zone (Norwegian time is thus UTC+2/+1 hour(s), depending on time of year) was considered when using the forecasted data. The forecasted weather data along with the actual measured weather data can be found in appendix A.4.

3.3. DATA AND MODEL STRUCTURE

To meet the goal of this work, the software MATLAB was the main tool used for this purpose. MATLAB has implemented different machine learning toolboxes that provide additional algorithms for training a network, such as a feedforward neural network. An overview of the toolboxes used for machine learning and deep learning in MATLAB, can be reviewed in [23].

Figure 20 presents a flowchart of the workflow pursued in this work. Before ANN models could be implemented, the dataset had to be pre-processed, including filtering of non-valid data recordings from the data logging center. Another part of the data pre-processing is the division of the data into training, validation and testing set, as well as standardizing the data to better fit in an ANN model.

After the pre-processing, implementation and optimization of the models were performed, before the trained models were tested with historic on-site data and historic forecasted weather data. These steps will be described in more details in the following subsections.

Data pre-processing
Model architecture
Model testing

Figure 20: Flowchart of work approach

3.4. Data Pre-Processing

Before any pre-processing of the raw data, there were 2 878 626 timesteps (samples) for 25 different variables. To prepare the data for a neural network model, different approaches were performed to filter out data with low validity or known biases in the dataset. The methods used for data pre-processing will now be explained and is illustrated in Figure 21.

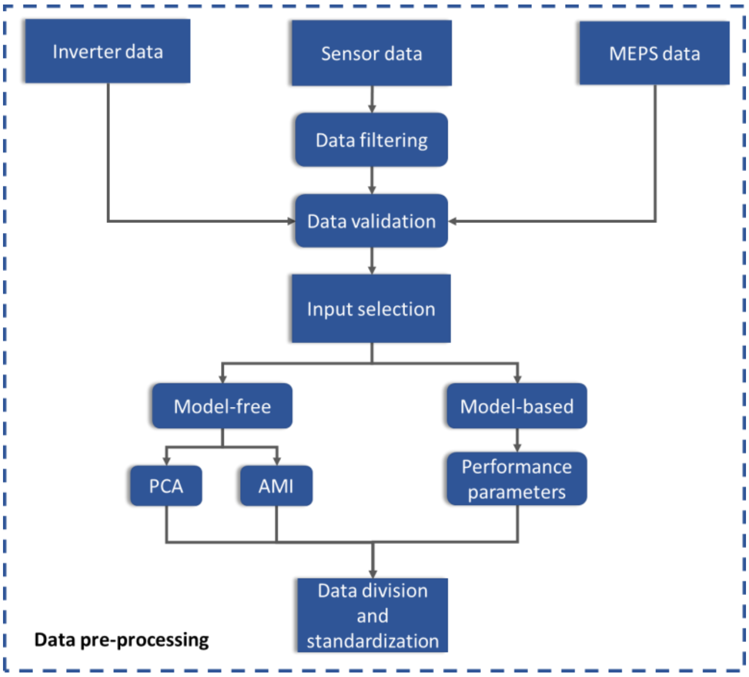


Figure 21: Pre-processing of dataset

Figure 21 illustrates there are three different datasets used in this work; inverter data, sensor data and MEPS data. The inverter data was obtained on a monthly basis measured by the inverters on-site, the sensor data are measured on minute-basis by AC and DC sensors on-site and the MEPS data is the forecasted weather parameters, on hourly basis. The sensor data went through a filtering process, which will be explained below, and the data were validated with one another. The inverter data was considered valid, allowing the sensor data to be validated on a monthly basis. The forecasted data from MEPS were compared with the filtered, validated data. Parameters from these datasets were used for input selection which is divided into two parts; model-free and model-based approach. The model-free approach involves principal component analysis (PCA) and average mutual information (AMI), which will be explained in following subsections. The model-based approach will be based on the ANN model performances and from this include additional input variables if necessary. As the input selection has been decided, the relevant data samples will be divided into different sets for ANN training purposes and standardized.

3.4.1. Data Filtering

The data filtering was performed in two parts;

Filtering of times without sunlight

For the ANN models to only consider datapoints where the PV modules produce power from sunlight, samples before sunrise and after sunset were removed from the dataset. This process was performed by using one of MATLAB's functions, created by Mahoot and can be found in [34], to calculate times of sunrise and sunset each day of the year at the exact location. The longitude, latitude and time zone were considered in this process. The MATLAB code for removal of times without sunlight in PV system dataset can be found in appendix A.5.

Filtering using maintenance log

For a PV system to operate at its best, service and maintenance is crucial to avoid additional losses. For certain periods of time, the system has various reasons for not operating or the sensors did not give valid measurements. The maintenance log was therefore thoroughly reviewed and datapoints that were not considered valid filtered out from the dataset. Not valid datapoints could be times where sensors were calibrated, a software update of the inverter was necessary or e.g. cleaning the module surface which could influence the performance during this period. All comments in the maintenance log were double-checked with a visualization of the logged data, and at times where vague comments in the log presented, it was taken a decision on whether to keeping the sample in the data set or not. As it was desired to keep as much data as possible, but not keep data that could influence the precision of an ANN model, all decisions were carefully considered. In case of doubt, data was removed from the set. See appendix A.6 for the MATLAB script created for filtering of data.

3.4.2. Data Validation

The sensor calibration of the voltage, V_{OC}, and current, $I_{SC,}$ (AC and DC) was reported to have had some complications, which resulted in data samples that were uncertain. In a preliminary study with characterization of the relevant PV system done by Lie [4], the DC power was measured as lower than the AC power, which should not be the case due to losses. The DC measurements were not considered further in this work, as the sensors were not measuring valid values. The AC power calculated from I_{AC} and V_{AC} were compared to the logged data from the inverter. Because the sensor-data had some issues with the calibration of the sensors, it was desired to check the validity of the values. To do this, recorded data from the inverters measured on monthly basis was available for cross-checking.

3.4.3. Input Selection

The input variables that were used in the ANN model were chosen based on three main aspects:

1. Available on-site and forecasted weather parameters
2. Results from model-free approaches; principal component analysis (PCA) and average mutual information (AMI)
3. Model-based analysis comparing performance from training ANN models

Also, information obtained from literature review have been used to select input variables. Liu [9] evaluated the irradiance and air temperature's effect on power output and Kaldellis [11] evaluated the air temperature and wind speed's effect on the output. Both indicating correlation between solar irradiance and power, and an effect from temperature. The module temperature would be affected by other parameters, such as the solar irradiance, wind speed and technical characteristics of relevant PV panels.

Input selection based on number 1 and 2 listed above was chosen before the actual training and testing of the ANN models. After observing some of the prediction results based on these two aspects, it was decided to use a model-based approach to include another input parameter, the air mass. If the performance parameters would improve by including air mass, this would also be part of the input selection of the models.

Because there are three different types of PV modules installed on the roof, three different models will be trained. Suntech and REC are of the same material and have the same installed capacity but is from two different manufacturers. The Sharp modules, however, is of thin film material and have a lower installed capacity. The input selection for the three modules will be the same and based on the three criteria listed above.

As the forecasted parameters included the air temperature, it was decided to use the air temperature measured on-site to train models. Based on Kaldellis. [11], the wind speed had an effect on the module temperature of the cells, and could have an impact on the modules behavior. It was thus decided to use air temperature along with wind speed, opposed to module temperature as this would need additional calculations, which could introduce errors the models would be trained on.

The initial assumption of which parameters that should be included in an ANN model is presented in Table 4.

Table 4: Initial assumption of input and output parameters of ANN models

Inputs	Output
GHI, WS, T	P_{AC}

In Table 4, *GHI* is the global horizontal irradiance, *T* is the air temperature and P_{AC} is the AC power production from the relevant PV modules.

3.4.4. Air Mass (Model-Based Approach)

The air mass was calculated as it could be used as an input variable in the ANN models. The air mass will give an indication of the sun's location on the sky, which will vary with time. As the irradiance for a horizontal surface (GHI) was available from the weather station parameters, the air mass can contribute with information regarding the spectral variations. Meaning different conditions can give the same GHI-value and by including air mass this will provide information regarding the sun's position.

To calculate the most accurate values for air mass, equation (2) from subsection 2.3 in the theoretical background was used. For the early morning hours and late evening hours, the sun's zenith angle is high, meaning the sun is close to the horizon. For these calculations, a zenith angle between 0 and 90° was used, where 90° is at the horizon and 0° is right overhead.

As there are uncertainties related to air mass calculations at times when the sun is close to the horizon, the equation did at some early and late hours indicate a zenith angle higher than 90°. This error was corrected by scaling the zenith angle, allowing the value to become just below 90°. This will only occur at times when few measurements were included within the relevant hour, because of the time of sunrise or sunset and the power production is significantly low.

This calculated air mass is supposed to give the model an indication to where the sun is on the sky relative to the earth surface, but as the models were trained it was noticed some outlier predictions where the air mass was suspected to interfere in a negative way. Because of this, the models were also tested by including a safety margin related to the air mass. This would make the highest possible zenith angle equal to 89° and used to avoid having high air mass values. See appendix A.7 for MATLAB script including calculations.

3.4.5. Principal Component Analysis (Model-Free Approach)

Having a dataset with several variables, it could be difficult to evaluate the importance of each variable. Principal component analysis (PCA) is a method that seeks to maximize the variance and ensure that the variables are independent of one another. This can eventually yield a better test or give more stable estimates of regression coefficients [35]. As mentioned in subsection 2.10 (ANN Model Development Process) regarding input selection, having more input variables than necessary can increase the likelihood of overfitting.

PCA seeks to maximize the variance with an unsupervised method and can be used when several measurements are made on each individual or object in one or more samples. The principal components are the eigenvector of the covariance matrix of the input sample with the largest eigenvalue [21, pp. 120-123].

Calculation of principal components can follow these steps:

1. Standardization of data
2. Compute covariance matrix
3. Compute eigenvectors and eigenvalues of the covariance matrix
4. Evaluate the principal components' explained variance

As the PCA has as many principal components as there are variables in the dataset, the principal components will present the dimension in which the variance of the entire dataset is explained. The first component will explain most of the variance and the second will explain the second most and so on. The last principal component, which explains the least of the variance in the dataset, will thus be evaluated if it is necessary to include in final input selection.

Figure 22 illustrates the two principal components of a dataset including two variables.

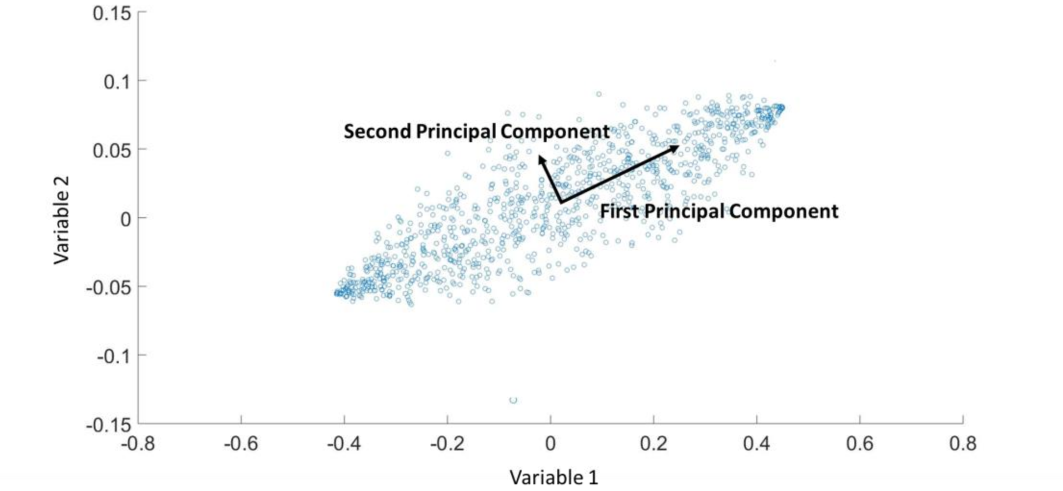


Figure 22: Principal components of a two-dimensional dataset

In MATLAB, PCA can be calculated with the following procedure (equation (12)) , which follows the four steps above [36].

$$[coeff, score, latent, tsquared, explained] = pca(X) \tag{12}$$

where:

coeff = *p*-by-*p* matrix where each column contains coefficients for principal component

score = matrix with principal component scores, rows = observations, columns = components

latent = eigenvalues of the covariance matrix (variances) of *X*

tsquared = Hotelling's T-Squared Static, sum of squares of the standardized scores of each observation

explained = Total variance explained by each principal component, in percentage

X = Matrix with dataset, rows = observations and columns = components

See appendix A.8 for script created in MATLAB to calculate PCA.

3.4.6. Average Mutual Information (Model-Free Approach)

Given knowledge of one variable (*X*), mutual information quantifies the dependence of another random variable (*Y*) [37]. These variables must be sampled simultaneously, and the mutual information can intuitively be information of how much one variable says about another. One theorem of the mutual information states that the mutual information between two variables is zero if they are statistically independent [38]. The AMI uses the variable's joint density function *p(x,y)* relative to their marginals, *p(x)* and *p(y),* to be defined as the negative of the entropy where entropy is a function which attempts to characterize the "unpredictability" of a random variable [37, p. 15, 38].

The mutual information can be calculated by using equation (13).

$$I(X,Y) = \sum_{X,Y} p(x,y) log \frac{p(x,y)}{p(x)p(y)} \tag{13}$$

where *p(x)* and *p(y)* are the marginal distributions of *X* and *Y*, and *p(x,y)* is the joint density function of *X* and *Y*.

AMI does not provide a finite value as a correlation coefficient does (usually an absolute value between 0 and 1). The mutual information provides a value that can range from 0, meaning complete independence, to infinity for completely correlated and continuous pair.

To calculate the AMI for the different input variables relative to AC power output from the dataset, some processing of the data samples had to be done. Because the AMI calculations are based on a continuous dataset, whereas data from Kjøita was discontinuous because of the pre-processing (e.g. filtering out hours without sunlight present), some alteration of the data samples were needed before implementing the AMI function. Variables that were used for AMI calculations were the solar irradiance (GHI), ambient temperature and the wind speed, that were compared with the power (AC). If these inputs were to have any lagging values, meaning that they are influenced by previous measurements, the AMI can reveal this, and the lagging values should be included as input in the ANN model.

The AMI calculations were based on Thomas et al. [37]. This calculation requires continuous samples, where some adaptation of the data used in the calculations would have to be done. See appendix A.9 for MATLAB script used for calculations related to AMI.

3.4.7. Data Standardization

For the input and output variables presented in Table 4 to be suitable to an ANN model, they were standardized. The standardization method used had the properties such that the mean would be approximately equal to zero and the standard deviation approximately equal to one for each of the features, see equation (14).

$$Z = \frac{x - \bar{x}}{STD} \tag{14}$$

where *Z* is the scaled score, $\bar{x}$ is the mean and *STD* is the standard deviation associated to *x*.

3.4.8. Data Division

The data was divided into training, validation and testing set with a 70% training, 15% validation and 15% testing. It was desirable to see whether a random data division would give a fair representation of the entire dataset, meaning that no matter how the dataset was divided, approximately the same coefficient of variation was achieved. To calculate this coefficient, equation (15) was used.

$$Coefficient\ of\ variation = \frac{STD}{\bar{x}} \tag{15}$$

where *STD* is the standard deviation and $\bar{x}$ is the mean.

These coefficients can be compared by testing different randomizations of the data samples (i.e. shuffling the lines). All lines in the data sets were randomized until a low deviation between the coefficients was reached. To evaluate a suitable data division, the absolute relative error between the coefficients of variation was found by using the absolute relative error equation, see equation (16).

$$Absolute\ relative\ error = \left|\frac{\Delta x}{x}\right| = \left|\frac{x_0 - x}{x}\right| \tag{16}$$

where *Δx* is the error, *x* is the true value and x_0 is the inferred value.

It was decided that an absolute relative error lower than 3% would be appropriate and the lines were shuffled until this criterion was reached. How the data division was performed in MATLAB can be seen in appendix A.10, presenting an example from using the Suntech PV modules. This procedure was done for each of the PV modules.

3.5. Model Architecture

An ANN (feedforward neural network) in MATLAB has these main aspects to consider when developing a model:

- Transfer function
- Training algorithm (backpropagation algorithm)
- Number of hidden neurons
- Number of hidden layers
- Number of epochs
- Weight initializer
- Validation check

For a model to be created and fit the purpose of this work, a method with trial and error was used. As there are several configurations that are included in creating an ANN model, it was decided to keep some constant and vary others to see which would provide the best model based on lowest mean square error (MSE). See Table 5 for an overview of constant configurations.

Table 5: Constant parameter configurations for model training

Parameter	Configuration
Network	Feedforward
Data division (training-validation-testing)	70-15-15
Training algorithm	Levenberg-Marquardt
Transfer function (hidden layer(s))	*Tansig*
Transfer function (output layer)	*Tansig*
Weight Initializer	Random
Maximum epochs	1000
Validation check (MSE)	1000
Runs	50

Each test was run 50 times, to account for variation of the initialization weights (the weights the models starts with to train a model will differ as they have random initialization). Other configurations than in Table 5 were tested, but as the result did not give valid output results or provided low performance parameters, these configurations were discarded. These configurations included input selection, different transfer functions and validation checks as well as number of hidden neurons and layers. Results regarding this will be briefly presented in the results and discussion section.

Beside the constant configurations of the model, two configurations were changed to find the optimal model based on lowest mean square error (model-based approach):

I. Hidden layers
II. Hidden neurons

To be able to decide these two configurations, it was created 7 different tests. The tests can be seen in Table 6 and were decided based on a trial and error method and to include a high number of neurons and low number of neurons.

Table 6: List of tests for optimal training configuration

Test number	Neurons in hidden layer 1	Neurons in hidden layer 2
1	8	-
2	10	-
3	20	-
4	30	-
5	50	-
6	10	2
7	20	2

As there was no previous experience with testing a feedforward neural network, the configurations were chosen based on results from learning how to use the MATLAB toolbox, and a variation between a low and large number of hidden neurons to see how the model performance changes.

Two additional tests were also performed but discarded as they did not appear to give good performance results based on a few training rounds. Tests that were discarded:

I. 4 neurons in one hidden layer
II. 120 neurons in one hidden layer

Results from this will be commented in the results and discussion section of this thesis.

To find the model that gives the lowest mean square error, all configurations except those listed in Table 6, were kept constant.

3.5.1. Model Optimization (Early Stopping)

For the model to avoid overfitting (overtraining) the network, it was needed to stop the training before this would occur. The training was therefore provided with a validation set, which would indicate when the network started to overfit, by stopping at a mean square error (MSE) value which would not improve after several new iterations. To find the best model, the validation check was set to 1000, as shown in Table 5. MATLAB's default configuration for this validation check was at 6, but with some experience by training the model several times, it was decided to use a high validation check to be able to obtain the lowest MSE.

3.5.2. Training Algorithm (Backpropagation)

The backpropagation algorithm chosen was based on MATLAB's recommendations with respect to training time and memory requirements. As a model can be trained using different training algorithm, it was for this work decided to work with and optimize the model based on one training algorithm. The algorithm chosen was the Levenberg-Marquardt algorithm, as this is often the fastest backpropagation algorithm in the Deep Learning Toolbox and highly recommended as a first-choice supervised algorithm, according to documentation provided by MathWorks [27].

The Levenberg-Marquardt algorithm is implemented in the feedforward toolbox and is well suited for a network where the mean squared error (MSE) is the performance index. The algorithm uses small steps to update the weights and with each epoch checks the error. If the error has increased or decreased, the steps for updating the weights are changed accordingly to eventually obtain convergence where minimum error is found [39].

3.5.3. Transfer Function

The transfer function used for hidden layers and output layer was *tansig* (hyperbolic tangent Sigmoid). This function deals with both negative and positive values, as this was necessary due to standardized input and output data. The default function in MATLAB was the *purelin*, which was first tested to observe the model performance. It was discovered that by using this function, the model could make predictions that would get non-valid results such as negative power production. The *logsig* function was also tested, but also provided non-valid results. Results related to this will be briefly presented in the results and discussion section.

The equation for the *tansig*-transfer function is shown in equation (17), which describes the input/output relation.

$$a = \frac{e^n - e^{-n}}{e^n + e^{-n}} \tag{17}$$

where *a* is the output and *n* is the input.

3.5.4. Model Evaluation

To select a model, the mean squared error (MSE) would be compared for the seven tests (see Table 6) performed for each of the PV modules. In order to evaluate the performance of these tests, metrics mean square error (MSE), R-squared (R^2), mean absolute relative error (MARE), mean absolute error (MAE) and mean bias error (MBE) will be included. All performance parameters will be calculated by using the standardized data.

The network performance for a feedforward neural network is evaluated by calculating the mean square error (MSE), which in MATLAB uses the function *immse*, see equation (18).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2 = immse(t_i, a_i) \tag{18}$$

where *N* is number of outputs, t_i is the target outputs and a_i is the network outputs [27].

R^2 represents the statistical measure of how the variance of one dependent variable is explained by variables in a regression model and is calculated with equation (19).

$$R^2 = 1 - \sum_i \frac{(a_i - t_i)^2}{(t_i - \bar{t}_i)^2} \tag{19}$$

where $\bar{t_\imath}$ is the mean target value, a_i and t_i as in equation (18).

Mean absolute relative error (MARE) is calculated with equation (20), mean absolute error (MAE) with equation (21) and the mean bias error (MBE) with equation (22).

$$MARE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{t_i - a_i}{t_i}\right| \tag{20}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}(|t_i - a_i|) \tag{21}$$

$$MBE = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i) \tag{22}$$

where *N*, t_i and a_i is as in equation (18).

3.6. Testing of the ANN Models

After having the seven tests with the evaluation parameters described in subsection 3.5.4 Model Evaluation, it will be possible to decide which of the model configurations (number of hidden layer and hidden neurons) that provide models with lowest mean square error (MSE). This model configuration will then be used with new data sets (a hold-out set and weather forecast data) and observed how the models perform.

For the final part, it was desired to test the models with historical forecasted weather parameters. Historical data from 24. April-30. April 2017 was obtained, which allowed for a two-days ahead forecast horizon for the 26.-30. April and a one-day ahead forecast horizon for 25.-30. April (as there was not obtained data from two days ahead of 25. April). As the ANN model should have been optimized as much as possible with the previously stated criteria, the prediction output of the model should mostly be influenced by the forecast error. A two day ahead forecast is expected to have a higher error compared to the one day ahead forecasts, due to more uncertainties in weather conditions.

Different MATLAB scripts with ANN models were created for each of the PV modules, to see an example of how this code was developed, see appendix A.22, which uses data from Suntech PV modules.

3.7. Choice of Method

There are several ways to find a well performing model and its model configurations. Several methods can be time consuming, which is why a model-based method have been the main focus in this work, with the goal to obtain the lowest mean square error (MSE) by comparing the models. Recommendations from previously done research and MATLAB's neural network toolbox guide have also been an influencer to decisions that have been made throughout the process (e.g. training algorithm).

Throughout the testing of the different model configurations, certain aspects were discovered that would compromise the model performance by reducing its performance. As most of these aspects have been discussed throughout this subsection, the main attributes to deciding ANN model configurations used in this work are listed below, along with parameters tested.

I. Input selection
   1. Without air mass
   2. With scaled air mass
   3. **With maximum air mass (≤ 89°)**

II. Transfer function
   1. *purelin*
   2. *logsig*
   3. ***tansig***

III. Validation checks
   1. 6
   2. **1000**

IV. Number of tests
   1. **Seven**
   2. Nine (including 4 and 120 hidden neurons in one hidden layer)

The highlighted configurations were those that provided a model performance of satisfaction, and results related to this will be presented in results and discussion section.

Other limitations were also needed to be able to achieve results in this work, whereas some could be part of reducing the model performance. One aspect that was not included in this work, that could give large deviations on certain days, is the precipitation. During times with heavy snowfall where the snow remains on the PV modules, a large deviation will occur if the sun starts shining as there will be no production. This should be considered when evaluating the model performances, if large errors occur.

Grid search is a method that goes through several hyperparameters, testing several different parameters at the time. This method has not been applied in this work as the only changing parameters were the hidden layers and neurons. It was thus decided to test with certain compositions of these parameters and evaluate the models based on this.

Underfitting and overfitting can be of an issue if the model is being overtrained. High epochs can cause this, but as a validation set has been applied in the training phase, this was considered a low risk.

Based on this, a method that uses model-free approaches, such as for the principal component analysis and the average mutual information, was performed before the network training. And a model-based approach for the testing of the hidden layers and neurons, as well as including one extra input parameter (air mass).

4. Results and Discussion

Based on the progress described in section 3 Methodology, results from this will now be presented. The input selection along with the ANN model specifications will firstly be presented. Results related to the training of the models will then be presented, followed by results from the models being tested with the forecasted weather data obtained from MEPS (MetCoOp Ensemble Prediction System). The model performances will be discussed along the way. The performance parameters are all calculated based on the standardized data. All model structures (number of hidden layers and neurons) will be referred to without including number of input variables, as this is constant for all (e.g. 5-5-1, meaning 5 hidden neurons in first and second layer, and 1 in the output layer).

4.1. Input Selection

The input selection for the ANN models were, as mention in section 3 Methodology, based on available data on-site and from MEPS' weather forecasts, model-free approach and model-based approach. The input selection from the model-free and models-based approaches with the available data will in the following subsections be presented.

4.1.1. Principal Component Analysis (Model-Free Approach)

The principal component analysis (PCA) were to find the dimensions where most of the variance in the dataset was explained. Depending on the last principal component and its percentage of variance explained, it will be assessed if the dimension (variable) is necessary to include in the input selection.

Figure 23 shows the principal component analysis based on three variables, global horizontal irradiance, wind speed and air temperature.

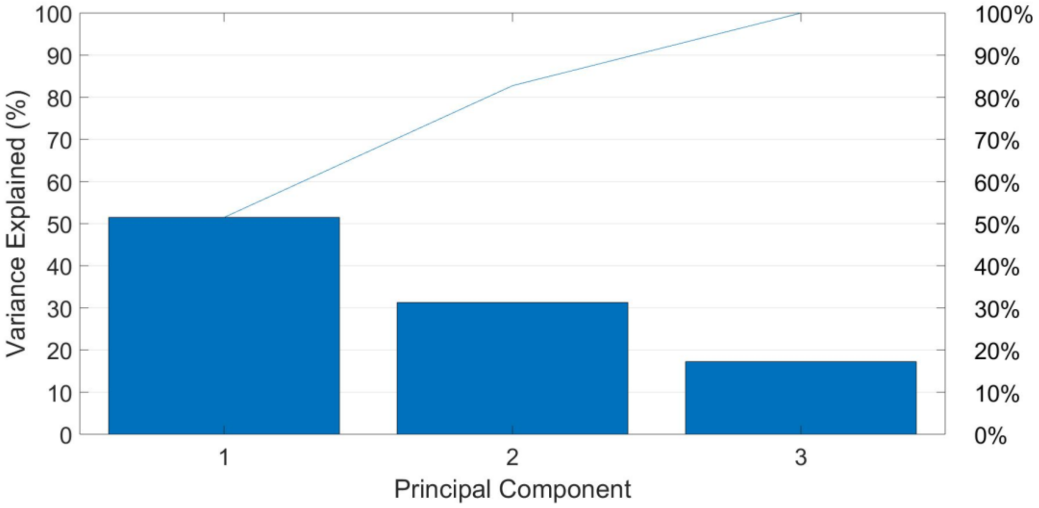


Figure 23: Pareto-chart of principal components

Figure 23 indicates that in order to explain the variances of the dataset, all three dimensions (variables) should be included. As the last component explains approximately 18% of the variance, this should also be included even though it explains the least. Nevertheless, this does not directly mean that these three variables explain the output power, it means that to explain the variance of the dataset provided to the principal component analysis, all three dimensions must be included.

4.1.2. Average Mutual Information (Model-Free Approach)

The average mutual information can reveal if there is a non-linear correlation between the input parameters and the output, with regard to time lag. Figure 24 presents the AMI between the global horizontal irradiance, air temperature and wind speed, relative to the AC power production. To calculate this, 500 days was used in order to obtain the AMI.

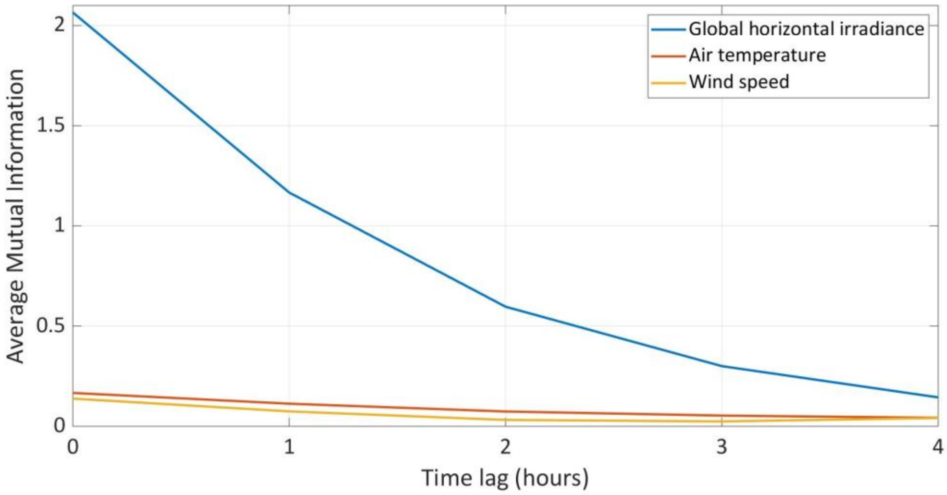


Figure 24: Average mutual information with a 4-hour time lag

From Figure 24, it can be observed a higher non-linear correlation between the solar irradiance (GHI), and the power output compared to the air temperature and wind speed. It does not appear to be any lagging values on hourly basis as all graphs drop after zero time lag (at the time of measurement). This indicates that the past hours (up to 4 hours back in time), the variables do not influence the power and should not be considered in an ANN model, as the AMI value drops for each time lag.

4.1.3. Air Mass (Model-Based Approach)

As the air mass could be of interest due to training a model based on several seasons, this was included to the input selection to observe the effect. It was noticed a distinguishable pattern during morning hours from the test results of the trained models, with a predicted power higher than the actual power when air mass was not included. Figure 25 illustrates the effect of including air mass as input variable for two days in April 2017, illustrated with predicted power using a feedforward model for Suntech PV modules (from a test set).

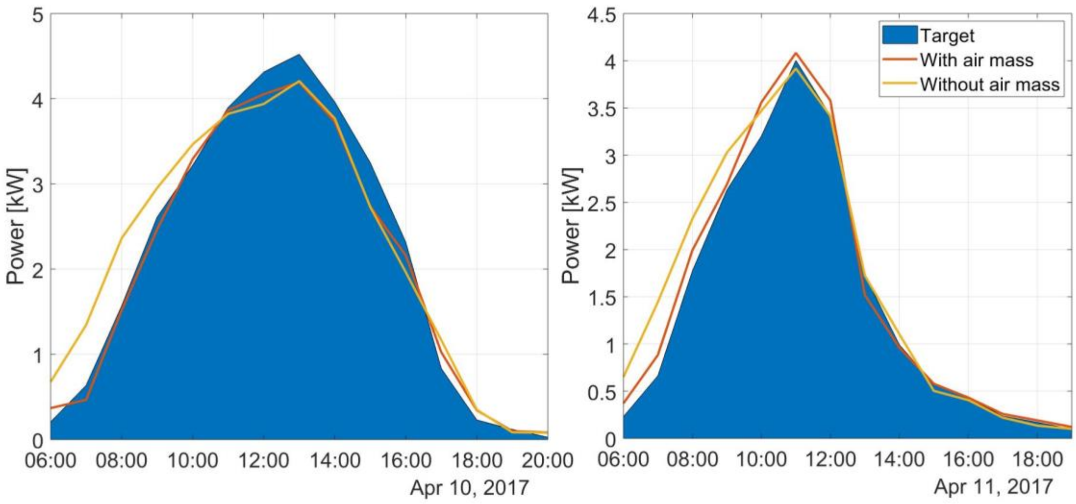


Figure 25: Including air mass to input selection, red and yellow line represent power predictions

Two days in April were used to demonstrate how the air mass affected the predicted output power. It is clear that the air mass contributes to a prediction output closer to the target. The predicted output without air mass as input appeared to have a higher prediction from sunrise to around 11:00 in the morning, which was a pattern that seem to occur on several occasions. As the predictions seem to improve, or in other words have better fitting to the target power, the air mass was decided to be included as an input variable for training models.

Two different approaches were used to calculate the air mass, as explained in subsection 3.4.4, due to high values with a scaling method.

4.1.4. Data Validation

After the pre-processing of the data (described in subsection 3.4 Data Pre-Processing), the data that could be used in training ANN models were validated. As sensor data could at some occasions have unreliable measured values, this data was compared to the monthly energy data measured by the inverters which was considered having valid values.

Figure 26 presents the two dataset (sensor data and inverter data) and how they can compare on a monthly basis for the Suntech, Sharp and REC PV modules, respectively.

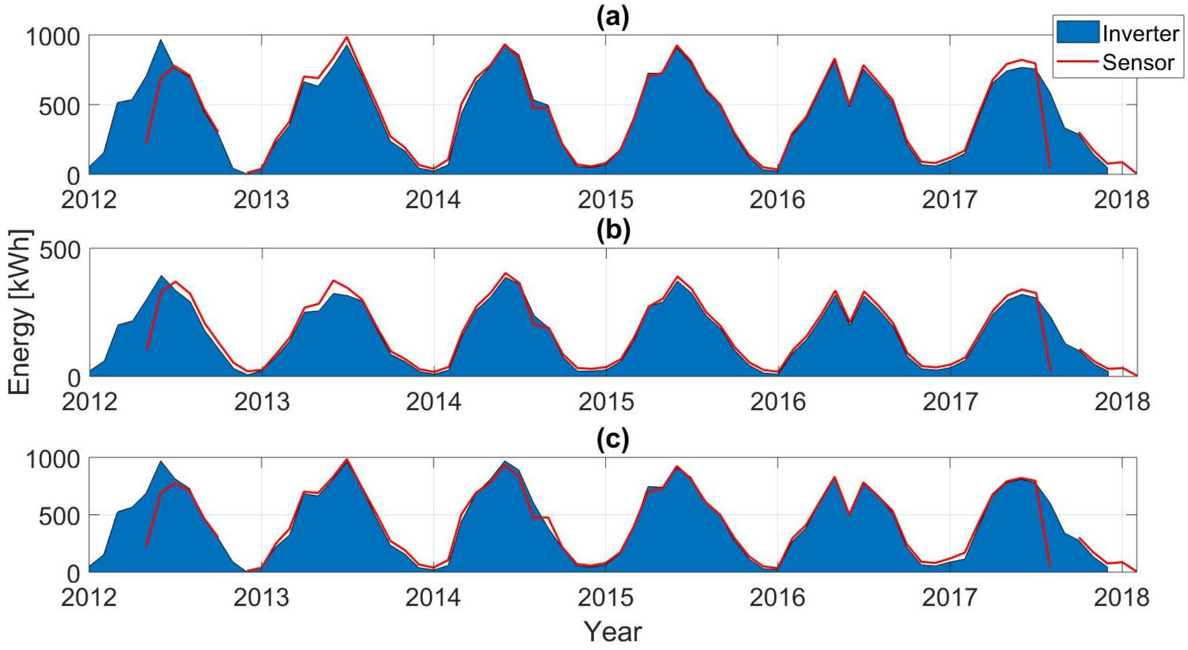


Figure 26: Comparing monthly energy production for (a) Suntech, (b) Sharp and (c) REC

As seen in Figure 26, certain months seem to have more deviations than others. Years 2014, 2015 and 2016 had monthly sensor values that were comparable to the inverter data and were considered valid. The months March and April 2017 also had low deviations and could be used for additional testing purposes of the models. Other years than those mentioned, appeared to have more deviations and were discarded for the rest of this work. During winter months (December to February, typically), irradiation- and hence power output- are low, and any uncertainties and/or offsets or noise in the sensor data will be relatively larger compared to the inverter data. Keeping this data may affect the model and should thus be carefully considered when evaluating a model performance.

For testing of the model with unseen data for continuous days, the months March and April 2017 were suggested to be used as test data (hold-out data), to illustrate how the models behave with typical values for different days. As these months have little variation from the monthly inverter data (see Figure 26) this was considered valid for testing purposes. This data will be referred to as hold-out set, as it is not part of the dataset related to the training of the models. Figure 27 presents the hourly power production for all three types of PV modules for the relevant months.

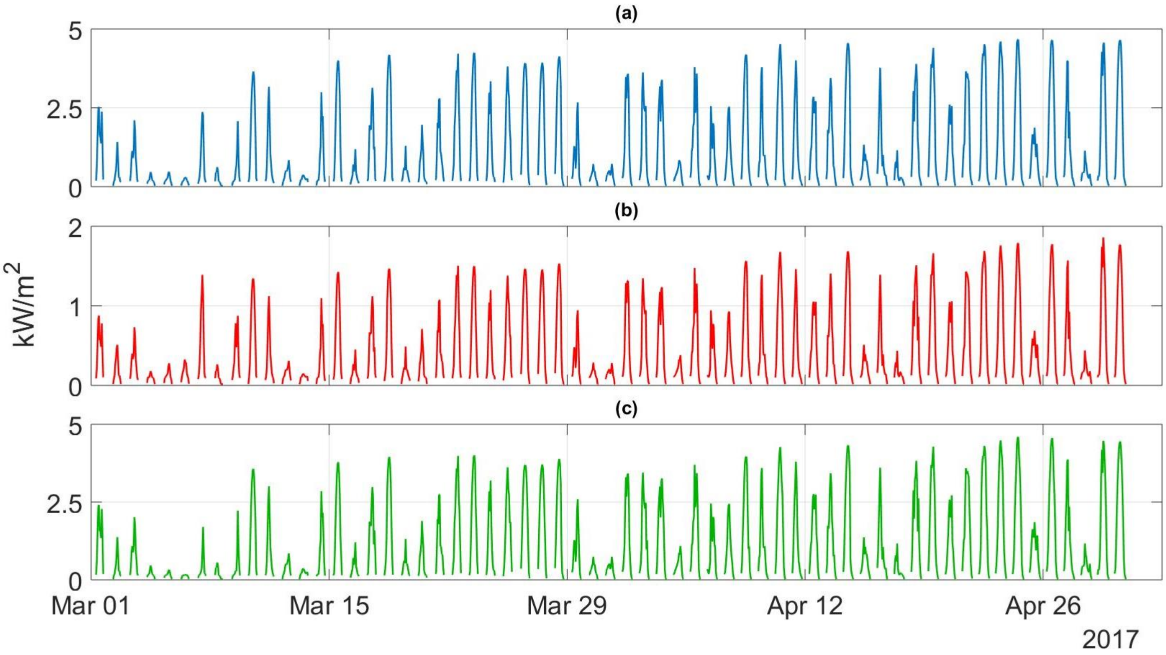


Figure 27: Suntech (a), Sharp (b) and REC (c) power production on hourly basis from hold-out set

The comparing of the three PV modules in Figure 27, similar patterns can be observed, which also allows for data validation. For Sharp, Figure 27(b), it can be observed a lower power production compared to Suntech and REC, due to the installed capacity and number of modules, referring to Table 1.

Table 7 presents data samples used for training, validation and test set in the ANN models and hold-out data that will present results from the trained model based on continuous days, along with number of samples for each of the PV modules.

Table 7: Data used for model training and testing

Data	Time	Samples
Data for ANN model training	01. January 2014 - 31. December 2016	13 275 (Suntech) 13 303 (Sharp) 13 290 (REC)
Hold-out data for testing	01. March - 30. April 2017	864 (all)

The number of samples in Table 7 for ANN model training differ for each of the PV modules. The reason for this is that there are some missing values (related to power production for each PV module), in the datasets which were not included.

4.1.5. Final Input Selection

By including the air mass as an input variable to the model, a distinguishable offset in the power prediction in the morning appeared to disappear (Figure 25), and the air mass was thus included for further model training. From the PCA analysis, all three variables (global horizontal irradiance, air temperature and wind speed) were required to explain the variance of the full dataset. The AMI indicated that there was no time lag for any of the weather parameters on hourly basis and no lagging values were thus included. Based on these matters, the input and output variables of the models would be as presented in Table 8.

Table 8: Final inputs and output for ANN models

Inputs	Output
$GHI, WS, T, \boldsymbol{AM}$	P_{AC}

In Table 8, *GHI* is the global horizontal irradiance, *T* is the air temperature, *AM* is the air mass and P_{AC} is the AC power output.

The inputs and output values are on hourly basis, and a graphic representation of all the variables used in training, validation and testing can be seen in appendix A.11.

In Figure 58 (a) in appendix A.11, the air mass has high peaks at times when the sun is close to the horizon, especially during summer (June-August). In Figure 58(b), the peaks have been cut-off and it is easier to see the shape of the air mass during daytime at summer and winter months. These two different ways of calculating the air mass will be tested as input variables in the ANN models to see if it can improve the model further by evaluating the performance parameters.

4.2. Data Division

After having decided on the input selection, a division of the data into three separate sets was performed. These sets were divided in such a way that the training set was 70%, and validation 15% and test set 15% of the total data samples, after the filtering and validation of the data. The data was divided so that the coefficient of variation (CV) would not exceed an absolute relative difference of 3% when comparing the parameter for each of the sets. Table 9 presents the descriptive statistics of the best randomization that was used when training ANN models.

Table 9: Descriptive statistics of the data division

PV module	Variable	Training set			Validation set			Test set		
		Mean	STD	CV	Mean	STD	CV	Mean	STD	CV
Suntech	GHI	229.36	232.05	**0.99**	222.47	226.54	**0.98**	226.06	229.34	**0.99**
	WS	2.43	1.25	**1.95**	2.43	1.27	**1.91**	2.47	1.26	**1.95**
	AT	11.49	6.35	**1.81**	11.27	6.16	**1.83**	11.48	6.21	**1.85**
	AM	11.37	17.77	**0.64**	11.98	18.23	**0.66**	11.57	17.91	**0.65**
	P_{AC}	1177.83	1334.48	**0.88**	1140.78	1315.29	**0.87**	1153.98	1318.50	**0.88**
Sharp	GHI	226.69	230.51	**0.98**	227.03	230.74	**0.98**	233.18	231.98	**1.01**
	WS	2.42	1.25	**1.94**	2.47	1.25	**1.97**	2.47	1.28	**1.94**
	AT	11.44	6.29	**1.82**	11.51	6.30	**1.83**	11.43	6.29	**1.82**
	AM	11.55	17.90	**0.65**	11.17	17.60	**0.63**	11.55	17.60	**0.64**
	P_{AC}	470.73	515.25	**0.91**	474.34	516.26	**0.92**	483.49	517.35	**0.93**
REC	GHI	229.10	231.80	**0.99**	223.42	225.63	**0.99**	224.81	230.85	**0.97**
	WS	2.43	1.25	**1.95**	2.45	1.28	**1.91**	2.43	1.25	**1.95**
	AT	11.45	6.27	**1.83**	11.56	6.35	**1.82**	11.37	6.34	**1.79**
	AM	11.49	17.84	**0.64**	11.65	18.06	**0.65**	11.38	18.06	**0.64**
	P_{AC}	1188.15	1303.35	**0.91**	1152.04	1262.94	**0.91**	1164.58	1297.09	**0.90**

In Table 9, *GHI* is the global horizontal irradiance, *WS* is the wind speed, *AT* air temperature, *AM* the air mass and P_{AC} is the AC power production.

4.3. Transfer Function

Before deciding on a transfer function that would be appropriate for the input-output variable relationship, different transfer functions were tested with the model configurations. Three different transfer functions were tested (*tansig, purelin and logsig*), whereas two of the three functions proved not to give the optimal input-output relationship. Figure 28 and Figure 29 show examples from unseen test data (from hold-out set of Suntech PV modules) by training models with *purelin* and *logsig* transfer functions, respectively, which provided the poorest relationships.

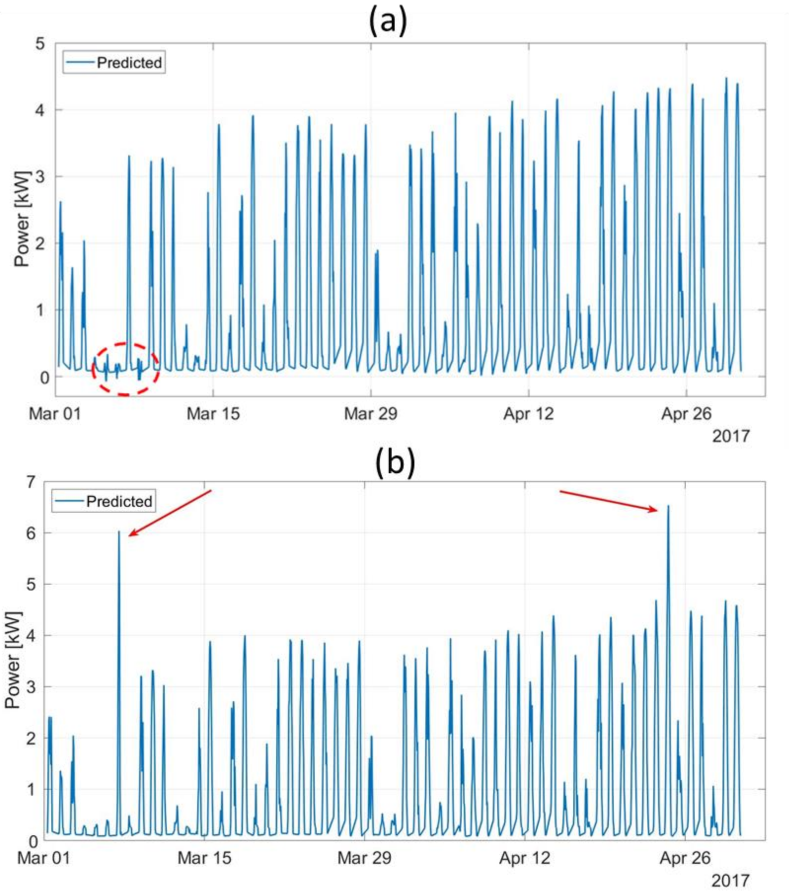


Figure 28: Example of (a) 10-1 and (b) 20-2-1 configuration and purelin as transfer function

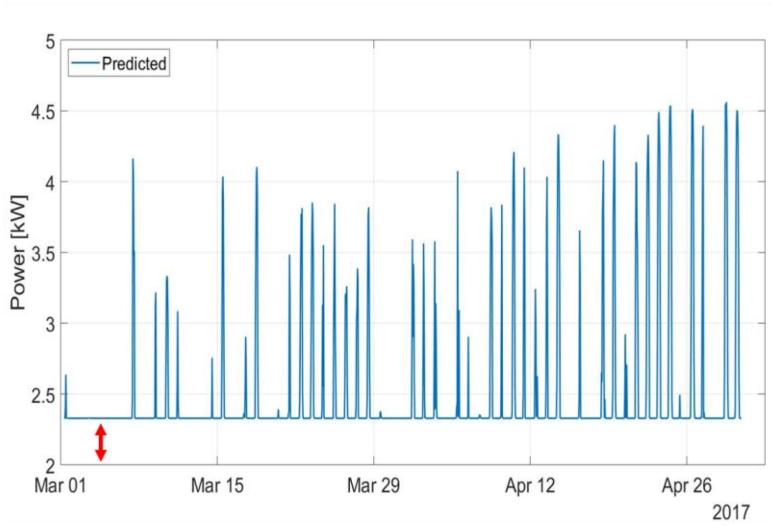


Figure 29: Example of network with 20-2-1 configuration and logsig as transfer function

Figure 28 illustrates prediction results by using *purelin* as transfer function in the output layer. The red circle in (a) marks an area where the models predicted negative power, which should never occur as the produced AC power is never measured less than 0. In Figure 28(b), the predicted power has high peaks on two occasions, which is during sunny days, but as the capacity of the inverter connected to Suntech PV modules is at 4.6 kWp, peaks should not be predicted this high.

In Figure 29, the *logsig* transfer function was used in the output layer, and it does not provide values down to 0. Because the *logsig* function only provides values between 0 and 1 and the standardization of the data has values less than 0, this would not be the appropriate transfer function for the model.

Because these two functions did not provide the wanted outcome, the *tansig* transfer function was chosen to further use in the model development. Results presented in the following subsections will therefore have *tansig* as the transfer function in hidden layers and output layer. A representation of the *tansig* function used in the hold-out set can be seen in Figure 30.

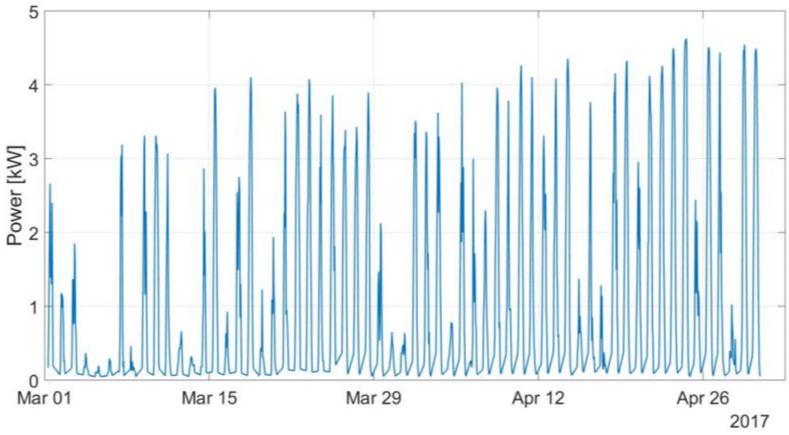


Figure 30: Example of network with 20-2-1 configuration and tansig as transfer function

4.4. Validation Checks

The default in MATLAB is to use a validation check of 6, meaning if the mean square error (MSE) value for the validation set would not improve after six epochs (backpropagations for weight alterations), the model would stop training. This was found to give lower performance parameters when training different models compared to using a higher validation check value. Because of this, it was tested with 1000 validations checks, meaning the model will run through all epochs (configuration set to maximum epochs 1000, see Table 5). Nevertheless, if the models find their optimal weights at a lower epoch, this will still be used as the optimal and performance parameters will be based on this. To illustrate with an example the difference using 6 validation checks to 1000, see Figure 31, where the *y*-axis is the percentage relative difference between the MSE values for 6 and 1000 validations checks. These configurations were tested with three different neuron structures; 4 hidden neurons in one layer (4-1), 10 hidden neurons in one layer (10-1) and 20 hidden neurons in first layer and 2 in second hidden layer (20-2-1).

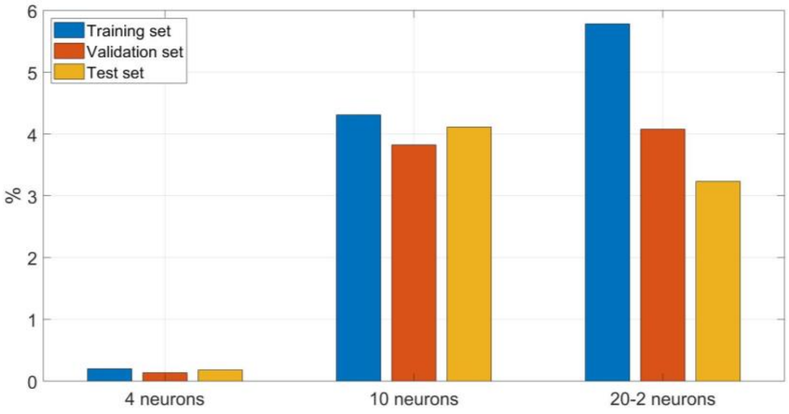


Figure 31: Percentage difference MSE (6 versus 1000 validation checks), Suntech PV modules

Figure 31 represents the mean square error deviation in percentage between a validation check of 6 and 1000. A positive percentage indicating MSE higher for 6 validation checks, which it is in all cases. This illustrates that a higher MSE was obtained for all datasets in the three different configurations of ANN models. Illustrated with an ANN model trained with Suntech PV modules. All performance parameters from using either 6 or 1000 validation checks, can be found in appendix A.12.

4.5. Model Structure and Performance Evaluation

The models have been trained with the seven different tests listed in Table 6 and results related to this will be presented in the following subsections. The models were first tested with air mass values that were scaled, but after observing predicted values far away from the target values, the models were also trained with an air mass with $\Theta_Z \leq 89°$. This was tested to see if it could improve the model performances. The results from using the two different methods showed an overall improvement of the models by having air mass with $\Theta_Z \leq 89°$, and performances based on this will be presented, as well as a regression representation of the models using the two different air masses for comparison purposes.

The configurations that provided the lowest mean square error (MSE) on the test set, were chosen as the optimal configuration for the different PV modules and used for testing with forecasted weather data. To evaluate the performance parameters, all training procedures were run 50 times to account for difference in initialization weights. All performance parameters will thus be presented with an averaged value from these tests, if not stated otherwise.

It was previously mentioned in 3.5 in the Methodology section that tests with 4 hidden neurons and 120 hidden neurons in one hidden layer were not included in the tests. These configurations appeared to have poorer performance compared to the seven tests presented in Table 6 and was therefore excluded from further testing. Examples from this can be seen in appendix A.13, where models were trained with Suntech PV modules.

4.5.1. Suntech PV Modules

Table 10 presents the results of the seven different tests performed using different numbers of hidden layers and hidden neurons.

Table 10: Test results Suntech PV modules, averaged for 50 runs

Performance Parameters Test set	**Test 1 8 neurons**	**Test 2 10 neurons**	**Test 3 20 neurons**	**Test 4 30 neurons**	**Test 5 50 neurons**	**Test 6 10_2 neurons**	**Test 7 20_2 neurons**
MSE	0.0503	0.0485	0.0464	0.0472	0.0481	0.0468	**0.0454**
MARE	0.1167	0.1088	0.1323	0.1388	0.1405	0.1225	**0.1392**
MAE	0.1434	0.1402	0.1362	0.1360	0.1366	0.1357	**0.1327**
MBE	-0.0054	-0.0047	-0.0043	-0.0048	-0.0050	-0.0071	**-0.0070**
R^2	0.9489	0.9507	0.9529	0.9521	0.9511	0.9524	**0.9539**
Best epoch	313	333	381	415	522	595	**584**
Time (s)	19	25	34	27	104	22	**23**

All tests appear to result with performance parameters relatively similar to each other, but test 7 gave the lowest mean square error (MSE) of 0.0454. This test had 2 hidden layers with 20 hidden neurons in the first and 2 hidden neurons in the second layer. Test 7 shows that mean average relative error (MARE) is relatively low, corresponding to approximately 14% absolute difference between the predicted and target values. Mean absolute error (MAE) and mean bias error (MBE) have a value of 0.1327 and -0.007 respectively. The MBE shows that the model overall tends to slightly overpredict. At last, the R^2-value (coefficient of determination) is above 0.95, meaning the prediction ability is high. The average epoch where the model found its lowest mean square error due to early stopping, can be seen increasing as the number of neurons increase. The same effect can be seen on average time it takes the model to finish training the network (until it reaches 1000 epochs). It appears that the network needs more time to solve the input-output relationship as the complexity of the model increases.

Table 11 presents the training, validation and test results using configurations in test 7 (20-2-1).

Table 11: Performance parameters for test 7, Suntech modules

Performance parameter	Test 7 20-2-1 model		
	Train set	Validation set	Test set
MSE	0.0461	0.0456	0.0454
MARE	0.3531	0.0695	0.1392
MAE	0.1323	0.1347	0.1327
MBE	0.0003	-0.0038	-0.0070
R^2	0.9543	0.9534	0.9539

The training set appears to have higher MSE and MARE values compared to the validation and test set, indicating that there are occasions with larger deviations from the target value in the training set. MARE is sensitive to outliers, meaning a large prediction deviation will influence MARE more than other performance parameters. MBE in the training set indicates slightly underprediction, while slightly overprediction for the validation and test set, while MAE is approximately equal for the three sets, slightly higher in the validation set. To see results from MATLAB for one network training example, see appendix A.15. Figure 63 appendix A.15 shows that there are more errors in the training set compared to the validation and test set, when the target power is supposed to be predicted low but is predicted high, which would appear to be the reason for the high MARE value in the training set. In appendix A.14, performance parameters for all 50 runs in test 7 are presented.

A network of 20-2-1, based on previously explained criteria was chosen as optimal configuration, where a schematic structure of the model can be seen in Figure 32 and prediction results from the test set is presented in Figure 33 and Figure 34. The prediction results (regression models) are based on test set with approximately equal MSE as for the average of 50 runs.

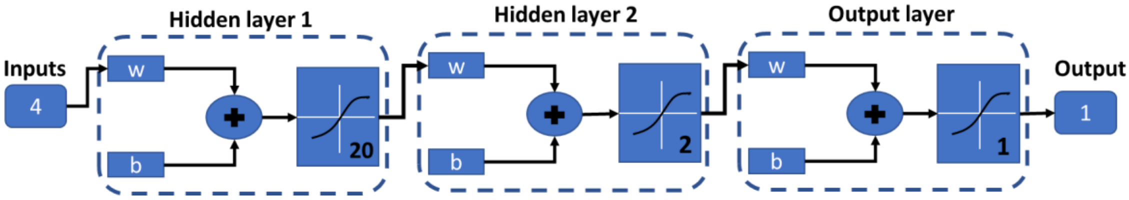


Figure 32: Neural network structure

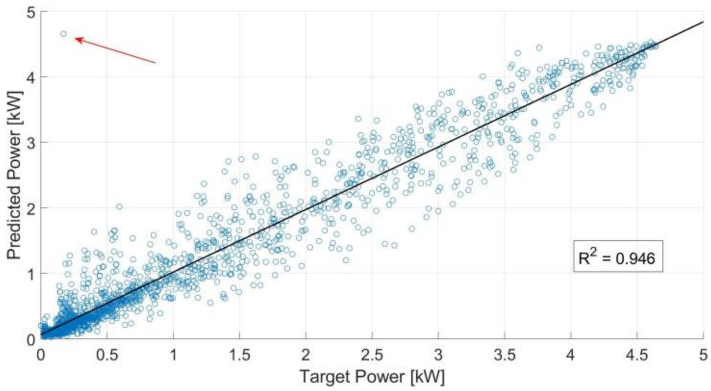


Figure 33: Prediction versus target values using scaled air mass (Suntech). MSE = 0.0533

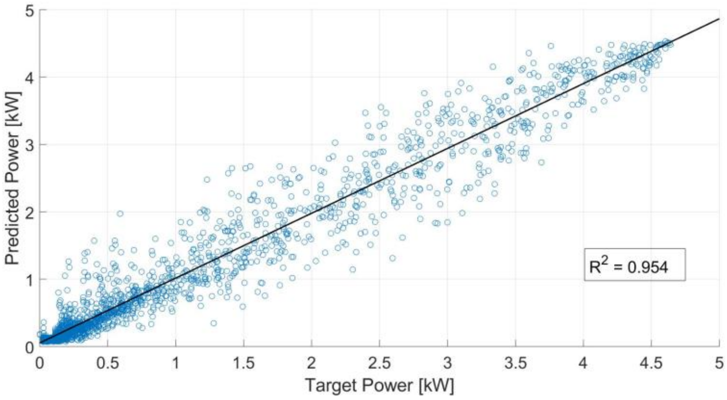


Figure 34: Prediction versus target values using air mass with $Θ_Z$ ≤ 89° (Suntech). MSE = 0.0454

The same patterns can be noticed for the two different figures (Figure 33 and Figure 34), but the outlier prediction (indicated with red arrow in Figure 33) has for the latter figure disappeared. Based on this finding model seems to behave more stable after removing the high peaks of the air mass and gives a higher R^2 value of 0.954 compared to 0.946 (averaged over 50 runs). Overall, the ANN model appears to do predictions with slightly higher precision during low and high power production, and where the largest deviations seem to be at approximately 1.5 kW. This deviation is relatively high, which can for some occasions have a deviation more than twice as high as the target value. As mentioned in the Limitation and Assumption section (1.2), snow cover was not included in the model and deviations related to this could be one explanation for deviations as the model would still predict power production if snow was covering the modules. Other than this, the deviations are difficult to assess without further investigations. Whether it is the trained model that have not managed to find the optimal input-output relationship, or if there are deviations in the raw data, is not clear by only looking at these results.

Model performances were also obtained for the scaled air mass input variable, see appendix A.16 Table 25 for results from this. Model performances using $Θ_Z$ ≤ 89° can be found in appendix A.17, Table 28.

4.5.2. Sharp PV Modules

Performance parameters from training a model with Sharp PV modules are presented in Table 12, representing the results for the test sets.

Table 12: Test results Sharp PV modules, averaged for 50 runs

Test set	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Test 6**	**Test 7**
	8 neurons	**10 neurons**	**20 neurons**	**30 neurons**	**50 neurons**	**10_2 neurons**	**20_2 neurons**
MSE	0.0351	0.0343	0.0328	0.0328	0.0330	0.0333	**0.0325**
MARE	0.0567	0.0529	0.0561	0.0585	0.0584	0.0567	**0.0576**
MAE	0.1223	0.1195	0.1141	0.1138	0.1133	0.1154	**0.1124**
MBE	-0.0006	-0.0012	-0.0003	0.0001	0.0004	-0.0012	**-0.0008**
R^2	0.9650	0.9659	0.9673	0.9673	0.9671	0.9668	**0.9676**
Best epoch	528	456	482	378	365	550	**494**
Time per run (s)	20	20	20	29	58	26	**41**

From the test results of model for the Sharp PV modules, the configurations with the lowest mean square error (MSE) is for test 7, with 20 neurons in first hidden layer and 2 neurons in second hidden layer, the same configuration as for Suntech. A representation of this can be found in Figure 32, in subsection 4.5.1 Suntech PV Modules, as they are equal. The difference between the MSE values for each test does not vary by much either, meaning the model is able to do predictions well with different numbers of neurons in hidden layers. The mean absolute relative error (MARE) is low, as well for mean bias error (MBE), with values of 0.0576 and -0.0008, respectively. The MBE indicates slightly overpredictions on the overall test set. The mean absolute error (MAE) is fairly low, with a value of 0.1124. The R^2 have a high value of 0.9676, which indicates high precision. It can be worth noticing that the errors for the Sharp modules are lower than for the Suntech modules. This suggests that the model configurations can better find the input-output relationship with the thin film PV modules (Sharp) than for the Suntech modules. The average best epochs appear to decrease as the complexity of the models increase, indicating that the model tend to have an early-stopping epoch at a lower value than with less complex models. The time it takes the models to run through all epochs (1000) increases with the complexity of the model.

Table 13 presents the performance parameters for test 7 for the Sharp modules.

Table 13: Performance parameters for test 7, Sharp modules

Performance parameter	**Test 7 20-2-1 model**		
	Train set	**Validation set**	**Test set**
MSE	0.0265	0.0268	0.0325
MARE	0.3394	0.0573	0.0576
MAE	0.1014	0.1029	0.1124
MBE	0.0002	0.0007	-0.0008
R^2	0.9735	0.9733	0.9676

The MSE is low for all sets, as well as the MAE and MBE. MAE is slightly higher for the test set compared to the training and validation, and MBE indicates modest overprediction for the training and validation set. MARE for training set is high compared to the validation and test set, meaning the training set appear to have larger prediction deviations than the two other sets. As MARE is sensitive to outliers, it appears that results from the training set have prediction values further from the target values, due to a higher MARE-value in this set. To see results from MATLAB for one training example, see appendix A.18, where less precise predictions can be observed for the training set (Figure 65), also indicated by MARE in Table 13. In appendix A.19, performance parameters for all 50 runs in test 7 are presented.

Reducing the peaks of the air mass appeared to remove predictions that were far from the target value for Suntech, but by studying Figure 35 and Figure 36, this does not appear to correct the one outlier prediction for the Sharp modules. The following figures are based on test set with approximately equal MSE as for the average of 50 runs.

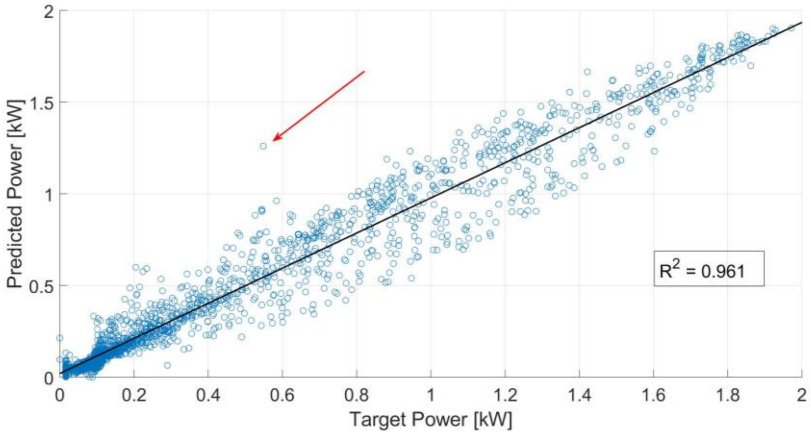


Figure 35: Prediction versus target values using scaled air mass (Sharp). MSE = 0.0335

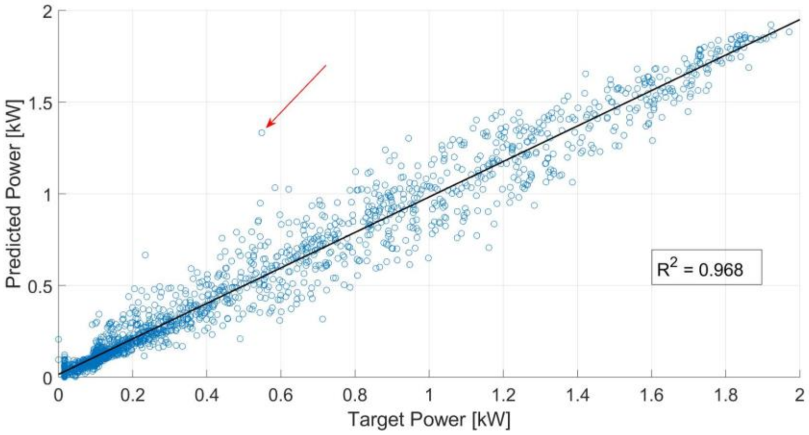


Figure 36: Prediction versus target values using air mass with $\Theta_Z \leq 89°$ (Sharp) MSE = 0.0325

By making this change in the air mass, it does however provide a R^2 value of 0.968 compared to 0.961, meaning the model is able to predict the observed outcome (targets) with higher precisions, as well as the MSE is lower for the latter case. The overall highest deviations appear to be between 0.5 and 1 kW, where the lowest errors seem to occur during low and high productions.

For performance parameter results for the scaled air mass, see appendix A.16, Table 26. Appendix A.17, Table 29 presents all performance parameters for air mass using theta ≤ 89°.

4.5.3. REC PV Modules

Finally, the model configurations tested with the REC PV modules, where the results can be seen in Table 14.

Table 14: Test results REC PV modules, averaged for 50 runs

Performance Parameters-Test set	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7
	8 neurons	10 neurons	20 neurons	30 neurons	50 neurons	10_2 neurons	20_2 neurons
MSE	0.0514	0.0511	0.0497	0.0496	0.0503	0.0500	**0.0492**
MARE	0.8000	0.8166	0.7454	0.7055	0.6384	0.8118	**0.7549**
MAE	0.1291	0.1283	0.1253	0.1244	0.1252	0.1245	**0.1223**
MBE	-0.0007	-0.0008	-0.0017	-0.0018	-0.0014	-0.0026	**-0.0033**
R^2	0.9486	0.9489	0.9503	0.9505	0.9497	0.9500	**0.9508**
Best epoch	283	325	380	335	199	651	**420**
Time (s)	14	16	25	40	81	21	**32**

The highest errors for the performance parameters were found for the REC PV modules when compared to the other two PV modules. The test that provided the lowest mean square error (MSE) was test 7 (20-2-1), which is also the same as for both Suntech and Sharp, and a representation of this can be found in Figure 32. The MSE was in this case 0.0492, but when it comes to the mean absolute relative error (MARE), the error has a high value of 0.75. MARE is, as previously mentioned, sensitive to larger errors, meaning it could appear to be certain predicted values that are further away from the target values. The mean bias error (MBE) appears to be relatively low, but as negative and positive cancel each other out when calculating this error, this may not give a fair representation of the model due to the high MARE value. The MAE is relatively low, meaning the overall average errors are not too high. R^2 is quite high with a value of approximately 0.95.

The performance parameters related to the REC modules for the best performing model based on the seven tests, are presented in Table 15.

Table 15: Performance parameters for test 7, REC modules

Performance Parameters	Test 7 20-2-1		
	Train set	Validation set	Test set
MSE	0.0424	0.0480	0.0492
MARE	0.0724	0.0246	0.7549
MAE	0.1173	0.1191	0.1223
MBE	0.0007	0.0002	-0.0033
R^2	0.9580	0.9494	0.9508

What can be observed for the different sets for the REC modules, is that the test set has higher values for the error parameters compared to the training and validation sets, see Table 15. Comparing this to the performance parameters for Suntech and Sharp, which indicated lower errors for the test and validation set, could give an indication of not optimal data division, errors in the datasets or a poor input-output prediction relationship. As the data division for the three PV modules was done separately, meaning the lines were shuffled in a way that would provide similar coefficient of variation depending on each of the modules, the data samples in the test set for REC modules may not be the same as for Suntech. If there are deviations in the raw data where the PV modules for some reason do not produce power that the model suspects, this could give large deviations in the outcome as MARE in Table 15. Even though the errors in the performance parameters in the test set appear to be relatively high, the R^2 value indicate that the overall predictions of the set are related to the target values. To see results from MATLAB for one training example, see appendix A.20. In appendix A.21, performance parameters for all 50 runs are presented.

Figure 37 and Figure 38 represents the predicted values compared to the target, with different air masses and the chosen configurations. Each result is represented by a test set with MSE approximately equal as for the average of 50 runs.

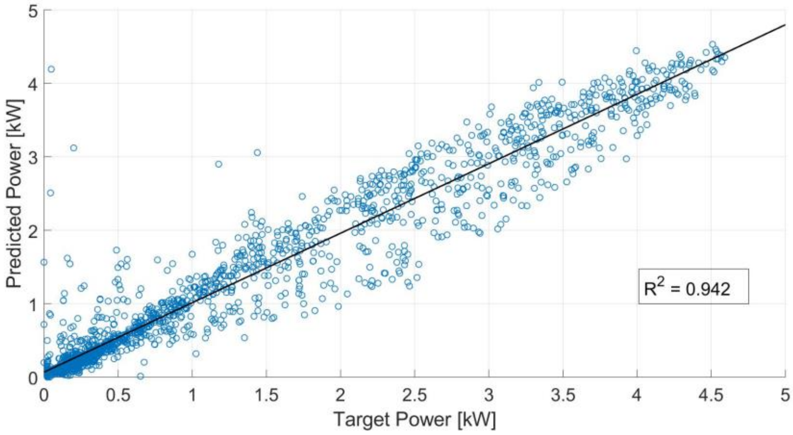


Figure 37: Prediction versus target values using scaled air mass (REC). MSE = 0.0576

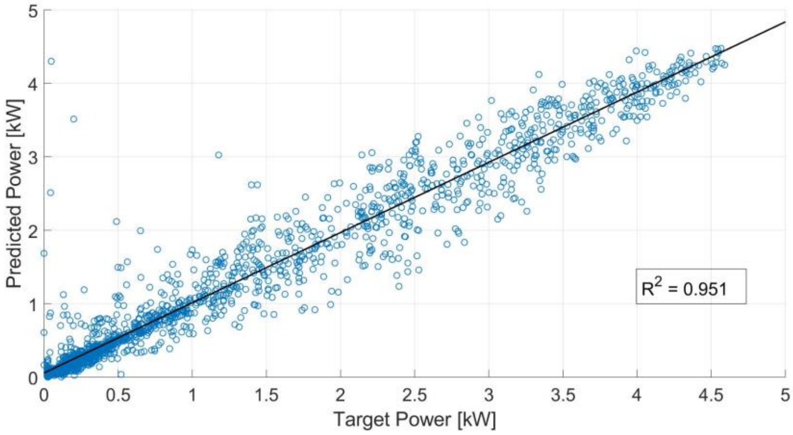


Figure 38: Prediction versus target values using air mass $\Theta_Z \leq 89°$(REC). MSE = 0.0492

As the MARE suggested, there are several outliers of predicted values that have larger deviations compared to the target values. This appears to occur mostly as the power production is low, between no production and 1.5 kW. After adjusting the air mass, both the MSE and R^2-value appear to improve, but there are still several predictions far from the targets.

Performance parameters using scaled air mass can be found in appendix A.16, Table 27 and results using $\Theta_Z \leq 89°$ can be found in Appendix A.17, Table 30.

4.6. Model choice

The three PV modules have the same input data, but the weights can still be assigned differently due to the output data not being the same. As seen from the test results for all the PV modules (Table 10, Table 12 and Table 14), the variations between the different tests within each PV module were all comparable. Because the training-procedures were run 50 times, the average for each run would differ slightly, but not enough to change the average values significantly by doing more re-runs. All tests were based on the same number of re-runs and should therefore be comparable. As the models all had the same input variables and the output values would presumably have variation similar to each other, it was before the testing assumed that the models for each of the PV modules would have the same configurations. The performance parameters from the seven tests did show some variations for each of the PV modules, but at the end indicated the best performing models to be with a configuration of 20-2-1 (20 neurons in first hidden layer, 2 in second hidden layer) for all three modules.

The mean squared error (MSE) for the Sharp modules is lower than for Suntech and REC, meaning the model had better fitting to the target values. MARE is sensitive to larger deviations, which for the REC test set indicated large deviations, which the regression plot (Figure 38) supported. The mean absolute error (MAE) and mean bias error (MBE) indicated that the models did tend to either over- or underpredict, where the overall MBE indicated slight overprediction for all three models (in the test set). The R^2- values indicated good fitting for all models, with values between 0.9508 and 0.9676.

Another aspect to consider when training a model is the amount of time it requires. As each model needs to be trained individually to assert for different behaviors of the different PV modules, three models were created. The results of training the models indicated that the more complex the model is, the longer it takes to train the model. But by studying the best epoch, it is the configurations with the highest amounts of neurons that tend to find its lowest mean square error (MSE) first. Meaning that the models with many neurons takes longer to complete each iteration (epoch) but can usually find its optimal weights at an earlier epoch (except for Suntech, where a higher complexity of the model also involved a higher number of best epoch). The results for using a high number of hidden neurons, such as the test with 120 neurons (see appendix A.13), required on average 423 second to run through 1000 epoch for one training. This involved a total time of almost 6 hours to train 50 separate models, which is a time-consuming process and with that many neurons would also make the network complex.

As a model with too many hidden layers and/or hidden neurons can make the model too complex and decrease the ability of generalization, it is important to consider this while choosing a model. However, depending on the input variables and the relationship to the output, the optimal number of neurons can vary based on the desired performance. The models chosen for the three PV modules in this work were based on a model-based approach and can only represent the model with best performance out of the tests that have been performed. This can, however, be a good indication of how ANN models can perform doing power prediction with the relevant PV system based on three years of historical data.

Based on this, models with configurations shown in Table 16 gave the best model performances based on the tests that have been performed. The configurations apply for all three PV modules.

Table 16: Chosen model configurations for all three PV modules

Parameters	Configuration
Network	Feedforward
Data division (training-validation-testing)	70-15-15
Training algorithm	Levenberg-Marquardt
Transfer function (hidden layer(s))	*Tansig*
Transfer function (output layer)	*Tansig*
Weight Initializer	Random
Validation check (MSE)	1000
Maximum epochs	1000
Runs	50
Number of hidden layers	2
Number of hidden neurons in layer #1	20
Number of hidden neurons in layer #2	2

4.7. Model Testing using Hold-Out Set

Based on the chosen model configurations for each type of PV modules found in subsection 4.5, continuous days in March and April 2017 were chosen to test with these configurations. A confidence interval of 95% was chosen to represent the variation of training the same model 50 times, to get the variations from the weight initializations.

4.7.1. Suntech PV Modules

The ANN model trained with Suntech modules gave an average R^2 of 0.9539 and a mean square error (MSE) of 0.0454 (test set), meaning the relationship between the predicted output power and the target output power was relatively good. The chosen configuration was a 20-2-1 feedforward neural network and by testing the continuous days in March and April 2017, the results for the hold-out set became as in Table 17.

Table 17: Performance parameters Suntech modules March/April 2017

Performance parameters	Suntech PV Modules
MSE	0.0602
MARE	0.3356
MAE	0.1448
MBE	0.0021
R^2	0.9403

The MSE appears to have a higher value compared to the test set from the training of the network, 0.0602 versus 0.0454. The R^2-value is relatively high, meaning the model still has a good fit to the targets. MARE does, however, indicate predictions with more deviations, while the MAE indicates an average error of 0.1448. The MBE imply that the model tends to underpredict on average for the entire set.

Figure 39 presents the prediction results from the two relevant months in 2017 with the chosen configuration of the ANN model. The results from one training that gave approximately the same MSE as for the 50 tests averaged have been chosen to demonstrate the performance.

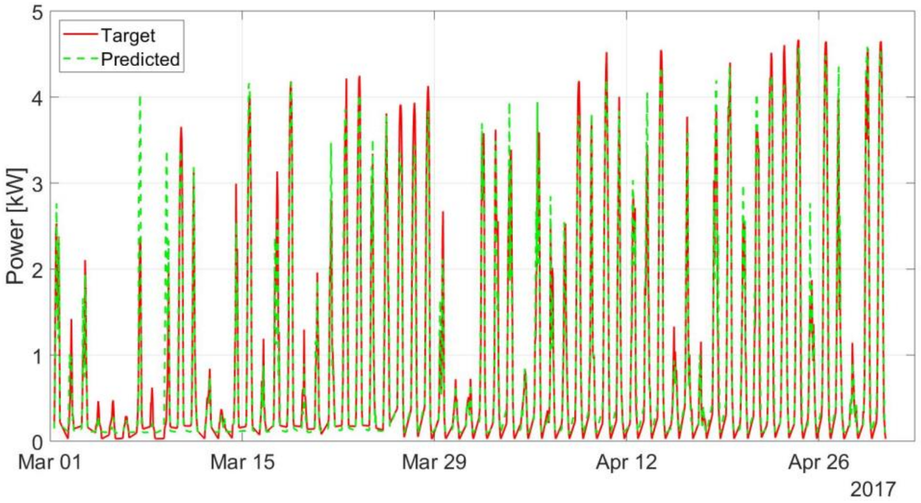


Figure 39: Test results from model with Suntech modules, hold-out set. MSE =0.0598

By visually inspect the prediction results compared to the target values in Figure 39, the model appears to have certain occurrences with both under- and overpredictions. For higher production-periods (during peaks around midday), the model tends to underpredict, but does at certain occasions appear to have a relatively close fitting to the target as well. There can also be observed higher prediction deviations in the first week of March. This will be investigated further in upcoming subsections. To inspect the model performance further, two random weeks have been chosen to graphically present the predictions. One week is in the beginning of March and one week in the end of April 2017 are presented in see Figure 40 and Figure 41, respectively. A 95% confidence interval will demonstrate most of the 50 runs, along with the averaged predictions from these runs.

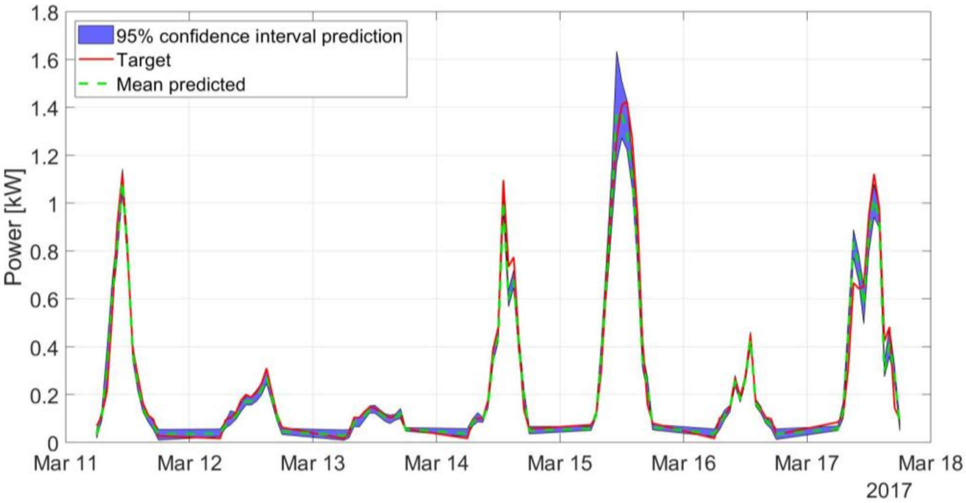


Figure 40: Prediction results Suntech for a week in March 2017

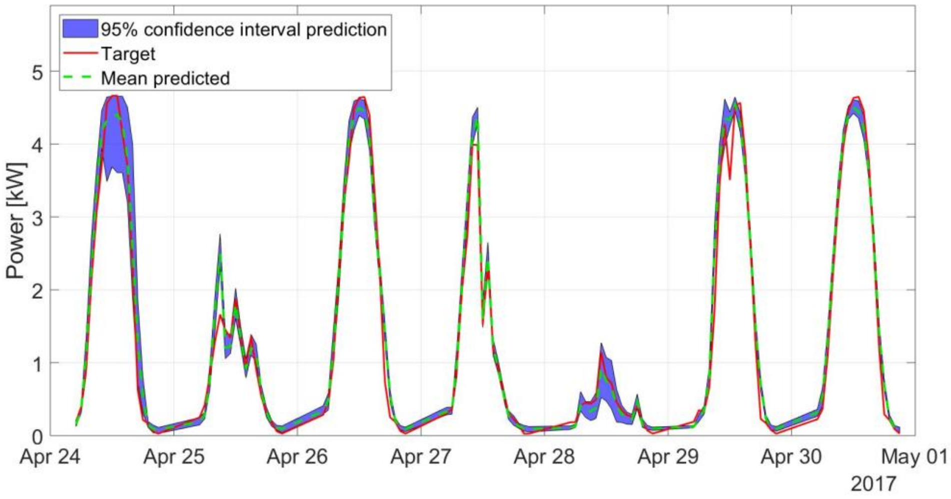


Figure 41: Prediction results Suntech for a week in April 2017

Figure 40 and Figure 41 show that the predictions and target relationship is relatively good for most hours during the relevant weeks. However, there are some hours the model seems to vary the predictions with up to 1 kW, which is a significant deviation. The model appears to underestimate slightly for high production (close to peak power, 4.6 kW for the Suntech modules). On 24. April 2017, the model seems to have a more difficult time predicting the power. By studying the input variables during these few hours (between 10:00 and 14:00), there seem to be a high wind speed compared to the average. Other input parameters seem to be consistent without any extreme values, which raises the idea that the wind speed might make the model more uncertain. Also, a high peak in the predictions can be observed on 25. April compared to the target power (first peak during this day). A closer look at the input variables during these days will be investigated at the end of this subsection.

The overall outcome of the two weeks tested seem to have a relationship fairly similar to the target, with some uncertainties.

4.7.2. Sharp PV Modules

The same two months as for Suntech have been tested for the 20-2-1 network for the Sharp modules, and the results are presented in Table 18.

Table 18: Performance parameters Sharp March/April 2017

Performance parameter	Sharp PV modules
MSE	0.0306
MARE	0.0290
MAE	0.1088
MBE	0.0010
R^2	0.9657

The model performance for March and April 2017 for the Sharp PV modules indicates that the model performs better with the hold-out set compared to the test set in the training process (see Table 13), meaning the predictions made for these continuous days appear more precise than the test set. MARE is low with a value of 0.029 and the model is slightly underpredicting as the MBE is 0.001. The R^2- value is, however, slightly lower than the test set (0.966 compared to 0.968), meaning certain predictions affects the regression line to be slightly lower. MAE shows an error of 0.1088, which is relatively low.

Figure 42 presents the predictions for one of the trained models that had similar MSE as the average for 50 runs.

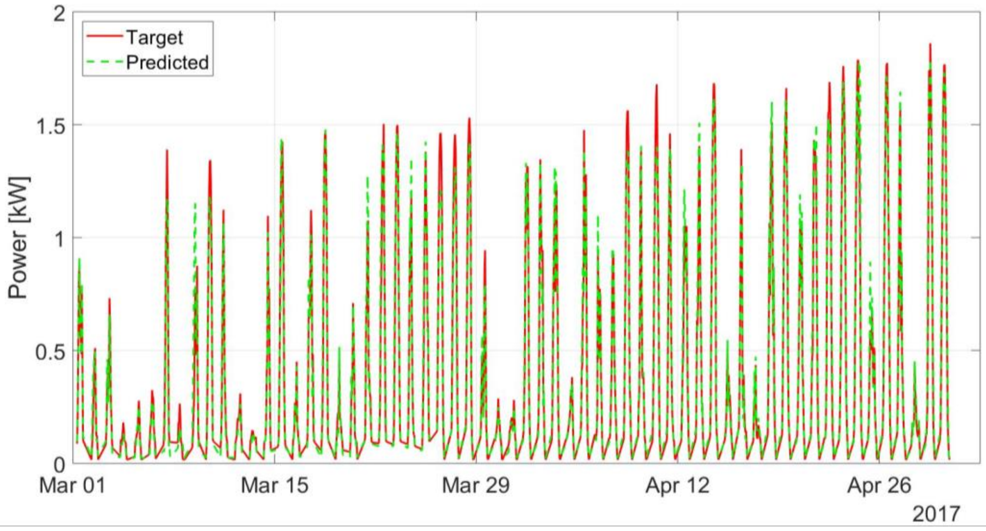


Figure 42: Test results from model with Sharp modules, hold-out set. MSE = 0.030

It can be observed certain periods where the model underpredicts compared to the target values, but the deviation often seem to be less if compared to the results in Figure 39, for the Suntech modules. A few occurrences with higher predictions than the targets can also be observed.

Figure 43 and Figure 44 presents the predictions with 95% confidence interval and the average over the 50 re-runs, for the same two weeks as for Suntech.

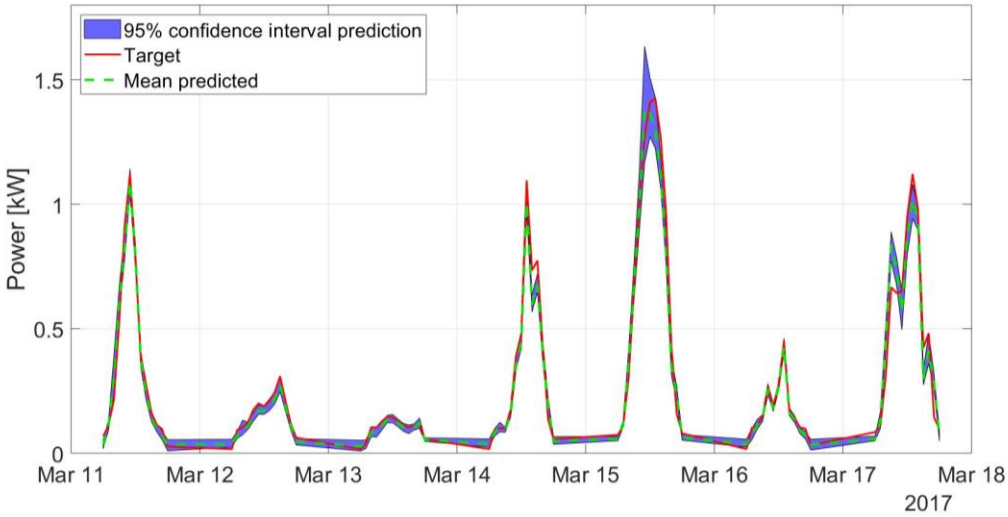


Figure 43: Prediction results Sharp for a week in March 2017

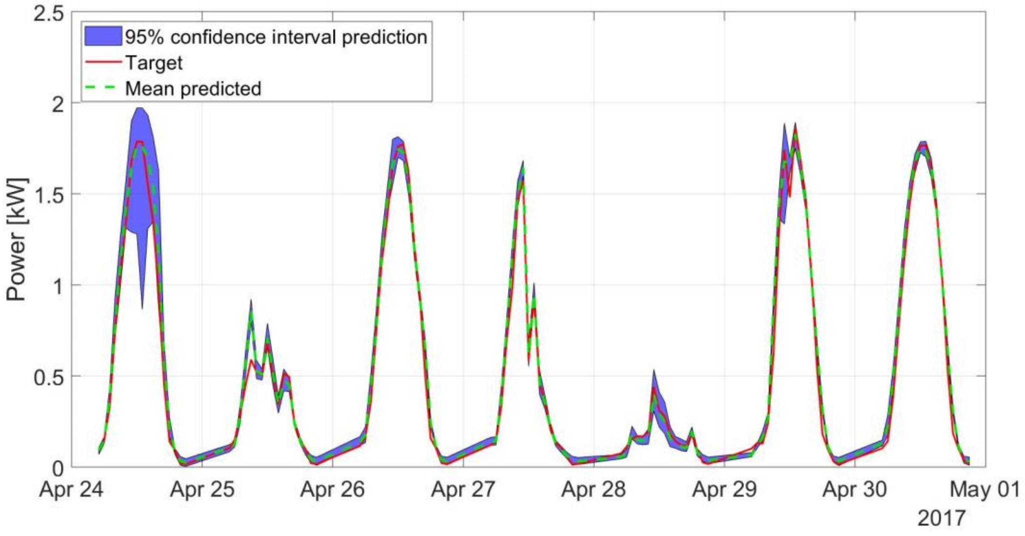


Figure 44: Prediction results Sharp for a week in April 2017

The same patterns can be observed for the Sharp PV modules as for the Suntech modules. The network output deviates slightly more as the power production is high. But by studying the mean predicted (green dotted line) and the target in Figure 43 and Figure 44, they seem to overlap each other better compared to Suntech. It is worth noticing that for 24. April 2017 the predicted power deviates with approximately 2 kW. As mentioned in the previous subsection (4.7.1), the wind speed is high during this time period and various weights given for this parameter can influence the outcome during extreme values. This drop in the power prediction did not occur for all training runs, meaning some training rounds had weights that obviously had the deviations higher. Further investigations of this will be done in later subsection.

4.7.3. REC PV Modules

The model configurations for REC PV modules was the same as for Suntech and Sharp, a network with configuration of 20-2-1. The R^2 value was 0.9508 and the mean square error (MSE) was 0.0492. Results for the model using data for the same two months as for Suntech and Sharp, are presented in Table 19.

Table 19: Performance parameters REC modules March/April 2017

Performance parameters	REC PV modules
MSE	0.0627
MARE	0.0567
MAE	0.1345
MBE	0.0030
R^2	0.9359

The performance parameters for the REC modules in the hold-out set appeared to perform slightly poorer compared to the test set in the training process (see Table 15). MARE, however, has a low value of approximately 5.7% absolute percentage error. The MSE for the hold-out set was 0.0627 and the MSE from the test set was 0.0492. As the MBE for the hold-out set is proposing that the model is slightly underpredicting, while for the test set it is slightly overpredicting. MAE indicates that the overall average absolute error is fairly low, yet higher compared to the test set. A representation for the entire hold-out set is presented in Figure 45.

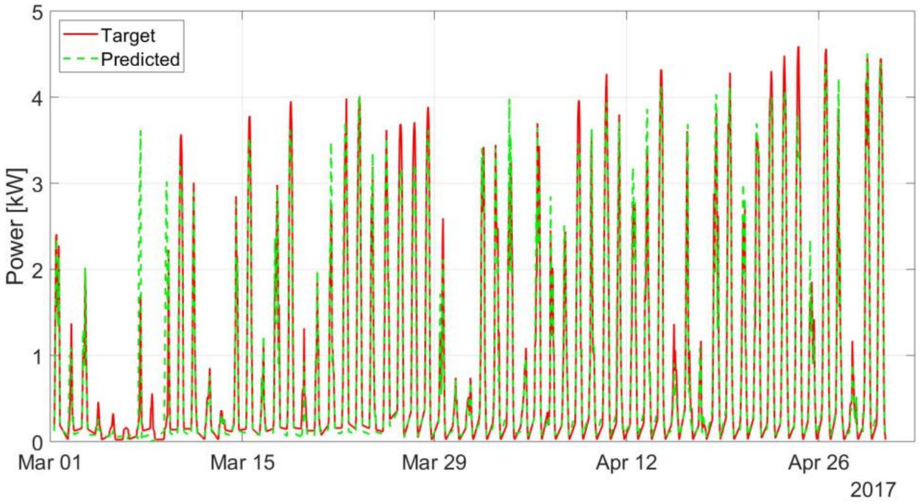


Figure 45: Test results from model with REC modules, hold-out set. MSE = 0.0627

The model appears to do prediction well compared to the target values, but with some over- and underpredictions occurring for certain days. The MBE for the hold-out set shows a slight underprediction, with a value of 0.0030. Certain days do tend to have a high prediction compared to the target, such as for 7. March, and low prediction for 8. March, which appeared to have the same as for Suntech, see Figure 39. Further investigation regarding these days will be presented in the nest subsection.

Prediction results for the two weeks in March and April 2017 for the REC modules are presented in Figure 46 and Figure 47.

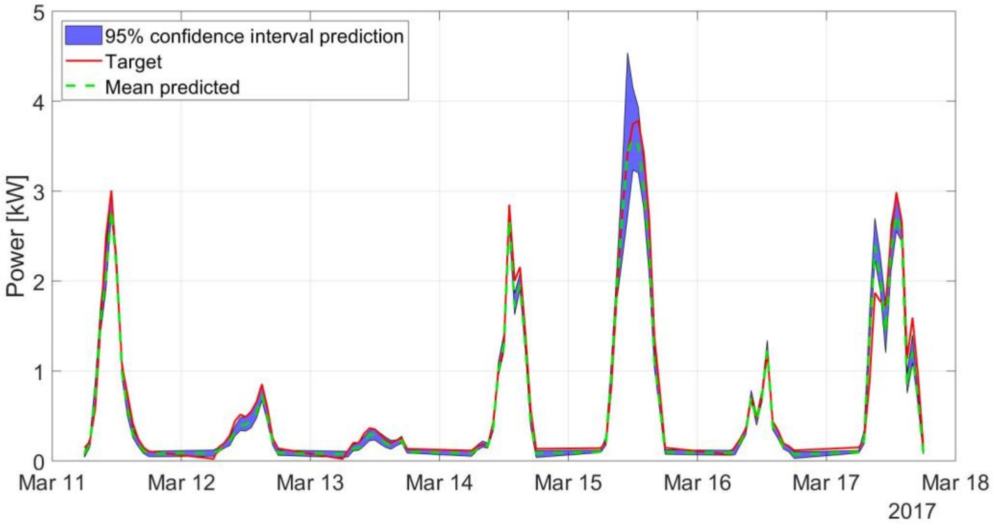


Figure 46: Prediction results REC for a week in March 2017

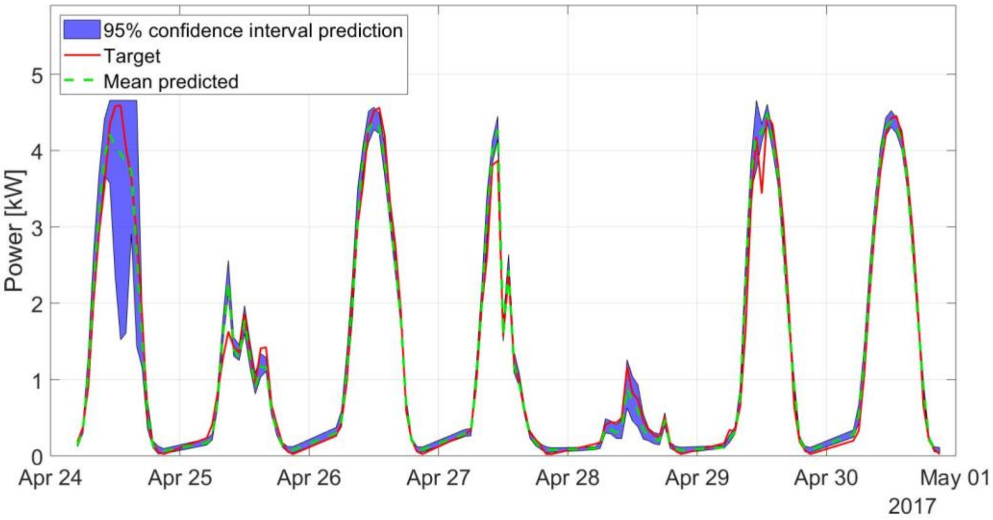


Figure 47: Prediction results REC for a week in April 2017

As for the other two PV modules (see subsection 4.7.1 and 4.7.2), it can be observed the same patterns. This was however expected due to the same input variables, but with a different output (power production), the results could deviate slightly. Also, for 24. April 2017, the model has difficulties giving an output similar to the target and the dip appears to be even more significant than for the other two PV modules and has the same peak for 25. April.

4.7.4. Analysis March and April 2017

Because of distinctive deviations in the first week of March 2017 and on 24. April and 25. April 2017, it was desired to further analyze the input variables during these days. It was observed a drop in the predicted power at around 12:00 on 24. April and a higher predicted power at around 10:00 on 25. April 2017. Figure 48 and Figure 49 illustrates the weather parameters for relevant days.

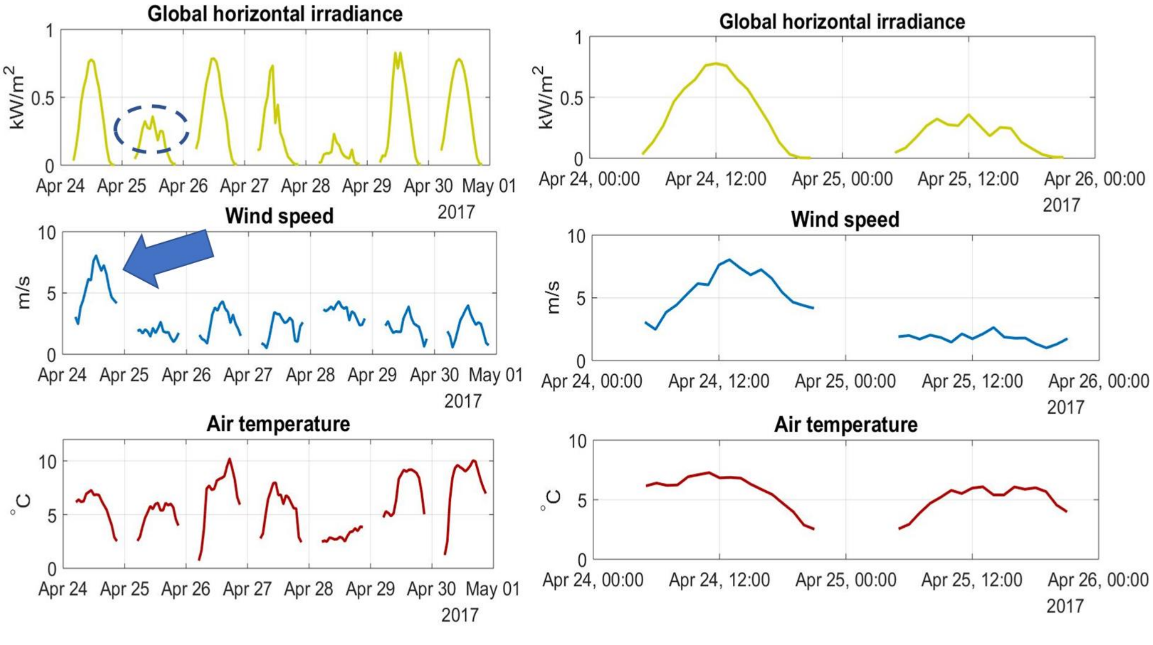


Figure 48: Weather parameters 24. -30. April 2017 (left) and 24.-25. April 2017 (right)

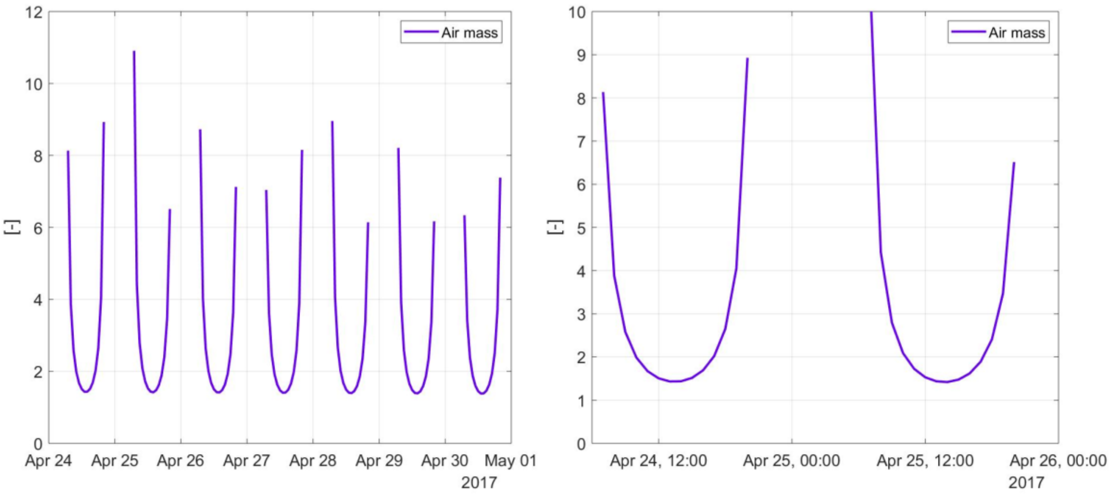


Figure 49: Air mass 24. -30. April 2017 (left) and 24.-25. April 2017 (right)

The only parameter that appears to have a variation larger than others for 24. April is the wind speed, see blue arrow in Figure 48. Allowing the suggestion that wind speed can compromise the models in such a way that can occasionally lower the performance of the models. For 25. April, all models appeared to give a prediction with a high peak at 10:00 (previously observed in Figure 41, Figure 44 and Figure 47). By studying the weather parameters in Figure 48 during the relevant time, it does not appear to be as high peak in the irradiance as the prediction results suggested. Neither the air temperature, wind speed or the air mass indicate any extreme values that could imply the models predicting such a high power production at this hour. This could indicate an input-output relationship not optimal.

It was also noticed higher prediction errors for Suntech and REC on 07. And 09 March. Figure 50 presents the relationship between the global horizontal irradiance and the power production for respective days.

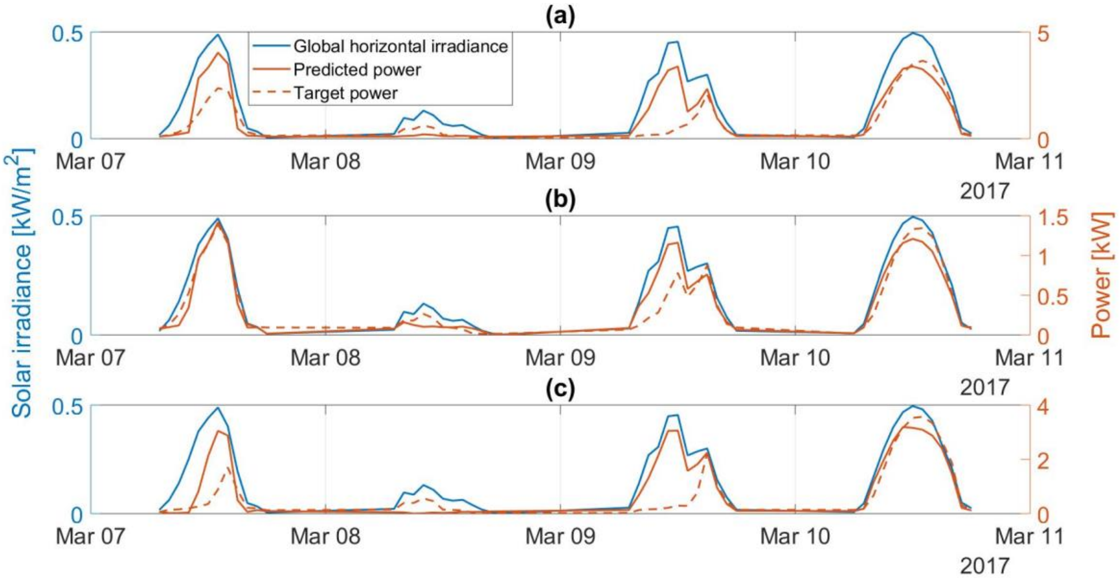


Figure 50: (a) Suntech, (b) Sharp and (c) REC power production versus solar irradiance

It is obvious from observing Figure 50 that the model predicts mostly based on the solar irradiance. However, the targets appear to be lower for the two days commented above, where Suntech and REC have the highest prediction deviations. From investigating the historical weather from a nearby weather station (Grimstad), it can be observed precipitation during these days and a negative air temperature, indicating snow fall. This could suggest that there is snow on top of the modules, causing low power production. As the ANN models do not consider snow cover, they will thus not discover this deviation. The precipitation from a weather station located at Grimstad was used as it was the closest weather station that had logged the precipitation for the relevant days.

4.8. Model Testing using MEPS Weather Forecasts

The final part of this work was to investigate how the different models behave with weather forecasts. Data obtained from MEPS was used as input to the trained models that gave the best performances for each of the PV modules. Due to uncertainties in weather forecasts, the performance parameters will also be affected by this. However, as these models are to predict future power production, it is crucial that the forecasted weather data can be used with the trained models. Data obtained from 24. April to 30. April 2017, with a forecast horizon of 67 hours have been used to illustrate how the models perform with the forecasted weather. The results have been averaged from 50 runs, as the predicted power for each hour had slight deviations for each run. Each day has been compared to the actual power production for the relevant day. To review the forecasted weather in comparison to the measured on-site weather parameters, see appendix A.4.

All models for the following subsections have the configuration listed in Table 16 and are trained on the sets for each individual PV module.

4.8.1. Suntech PV Modules

Because of the uncertainties related to weather forecasts, the ANN models will also give predictions based on these uncertainties and larger errors compared to using on-site weather data are to be expected.

Table 20 presents the performance parameters for the Suntech PV modules with one- and two-days ahead forecast horizons.

Table 20: Performance parameters with forecasted weather from MEPS, Suntech modules

Performance parameters	Suntech	
	One-day ahead	Two-days ahead
MSE	0.2647	0.2996
MARE	0.7755	0.7168
MAE	0.2991	0.3020
MBE	-0.1194	0.0800
R^2	0.7991	0.7995

As expected, the mean square error (MSE) is higher when using forecasted parameters compared to using on-site weather data. The parameters presented in Table 20 will represent both the model performance as well as the weather forecast performance, where deviations from both will affect the outcome of the predictions. As weather forecasts tend to be more precise with a forecast horizon closer to the actual day, the results from the model indicates the same. The mean absolute relative error (MARE) is high, meaning certain predictions are deviating from the target. Both MAE and MBE are higher compared to the test and hold-out sets. MBE indicates an overall overprediction for the one-day ahead and underprediction for the two-days ahead. R^2 is also lower than for the test and hold-out set, indicating a lower prediction ability.

Prediction results related to the forecasted weather data with the best performing model configurations can be seen in Figure 51.

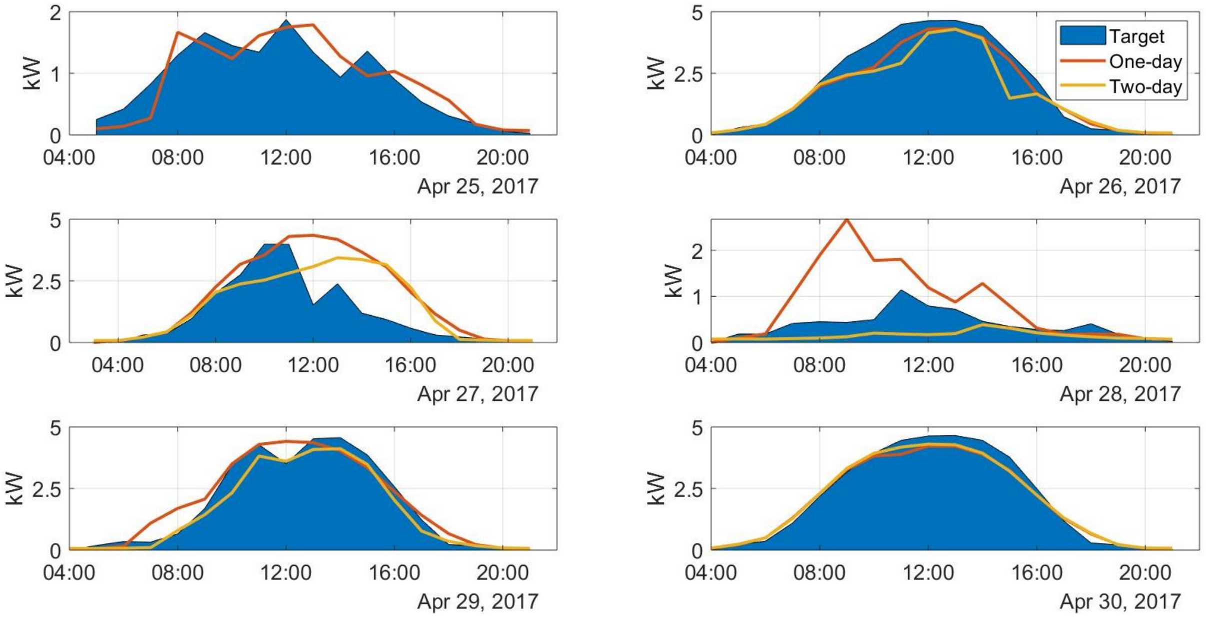


Figure 51: Power predictions (red and yellow line) of 25.-30. April 2017, Suntech PV modules

As 25. April, in Figure 51, only has one-day ahead forecast horizon is due to the obtained data was from 24. April 2017, which prevented to get data from two-days ahead of the 25. April. The predictions in the figure illustrates the importance of accurate weather forecasts and how the deviations of the predicted values from the model can differ from the target values. For clear days (such as 26. April and 30. April), the model tends to predict a lower power production compared to the actual production. As these days probably did not have much clouds, the model should be able to give a prediction close to the target. However, studying the forecasted weather parameters in comparison to the measured on-site parameters in appendix A.4, the forecasted solar irradiance, which is considered the highest influencer to the power production, is lower than the actual irradiance for the same time period. Due to this, the predicted power is lower compared to the target. It can also be observed deviations related to dips and peaks in the power production, as can clearly be seen for days 25. April, where a time shift of when the predicted power appears compared to the target. Comparing the forecasts from two days ahead to the one-day ahead, most predictions using the one-day ahead forecast seem to be more precise compared to the two-days ahead, which should be expected due to the uncertainties related to weather change.

4.8.2. Sharp PV Modules

The model configuration with 20-2-1 ANN was used to predict power using weather forecast for the Sharp modules, and results can be seen in Table 21.

Table 21: Performance parameters with forecasted weather from MEPS, Sharp modules

Performance parameters	**Sharp**	
	One-day ahead	**Two-days ahead**
MSE	0.2378	0.2252
MARE	0.8884	1.0678
MAE	0.2723	0.2595
MBE	-0.1295	0.0119
R^2	0.8069	0.8389

Also the errors are higher for the Sharp model compared to the test and hold-out set, but the two-days ahead forecast appears to have a slightly lower MSE compared to the one-day ahead. However, MARE is higher for the two-days ahead compared to the one-day ahead forecast. MAE is higher for the one-day ahead, where MBE indicates that the predictions tend to overestimate the power production for the one-day ahead, but underestimate the power for the two-days ahead. To visually inspect the prediction results, see Figure 52.

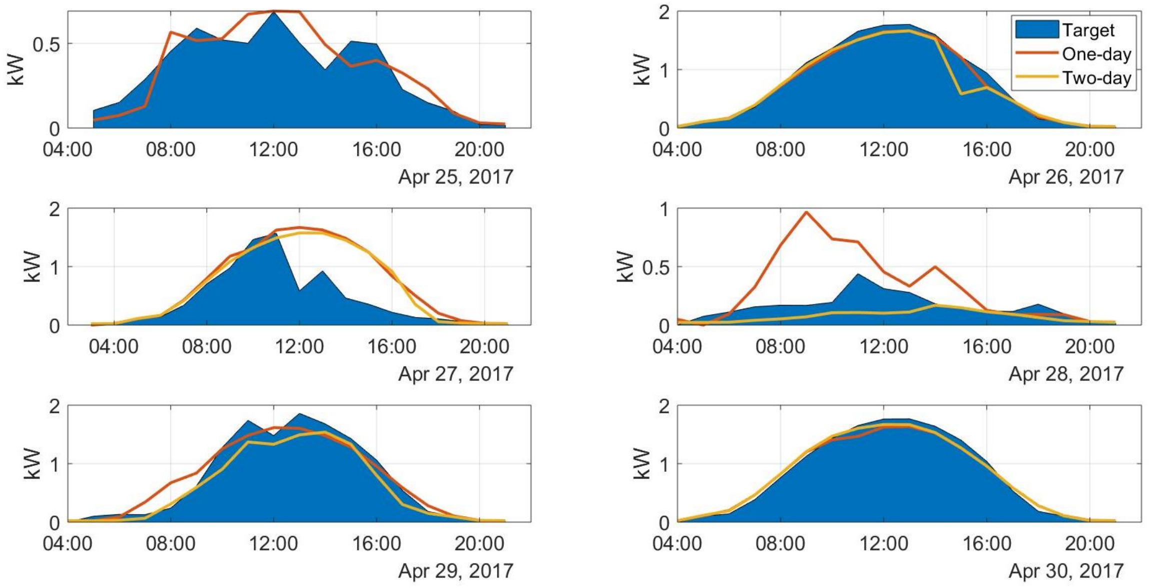


Figure 52: Power predictions (red and yellow line) of 25.-30. April 2017, Sharp PV modules

The same pattern for the predicted values using the forecasted weather parameters can be seen for the Sharp modules, as it was for Suntech. The predicted power during sunny days (26. and 30. April) do appear to be slightly closer to the target power during midday, compared to the Suntech model. 27. April was forecasted to be a clear day, which is why the predicted values became as shown in Figure 52. The MBE shows that the one-day ahead forecast had the tendency to overpredict, which 28. April could be a large contributor to, and the two-days ahead MBE indicated underprediction, which appears to be the case for most days, except for 27. April after 11:00.

4.8.3. REC PV Modules

Table 22 presents the results from a one- and two-days ahead forecast horizon, with the model trained with REC PV modules.

Table 22: Performance parameters with forecasted weather from MEPS, REC modules

Performance parameters	REC	
	One-day ahead	Two-days ahead
MSE	0.2647	0.2719
MARE	0.4703	0.8570
MAE	0.3040	0.3005
MBE	-0.1100	0.0576
R^2	0.7955	0.8145

Also the errors are higher for the forecasted weather data compared to the on-site data for the REC modules, which was expected. The errors are lower for the one-day ahead forecast horizon compared to the two-day forecast horizon, but the R^2-value appear to be slighly lower. As for the other two PV modules, MARE is relavitely high and MBE for one-day ahead indicates overprediction and for two-days ahead underprediction.

Figure 53 presents the predictions using a trained model for the REC modules, using the forecasted weather parameters.

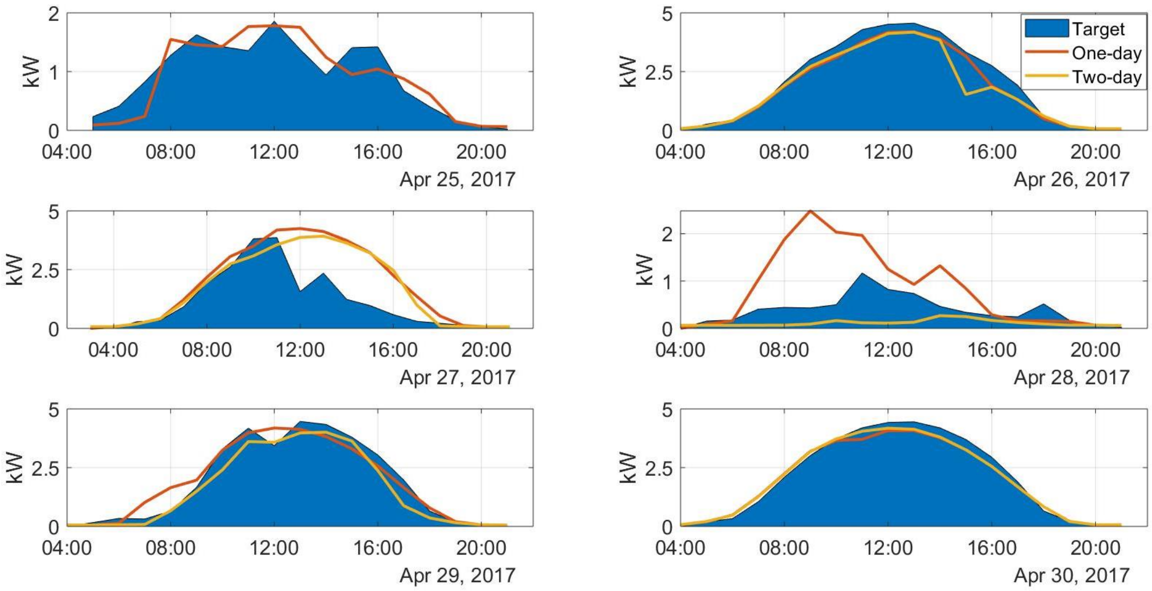


Figure 53: Power predictions (red and yellow line) of 25.-30. April 2017, REC modules

As for the other two models, the model trained for the REC PV modules follow the patterns of the forecasted weather parameters, where the solar irradiance appears to have the largest impact on the power production.

4.9. COMPARISON OF MODEL PERFORMANCES

The three different types of PV modules ended up having the same model configurations of 20-2-1, but the performance parameters were different based on the training of the models. Figure 54 compare the mean square errors (MSE) for each of the PV modules from the results obtained in this work.

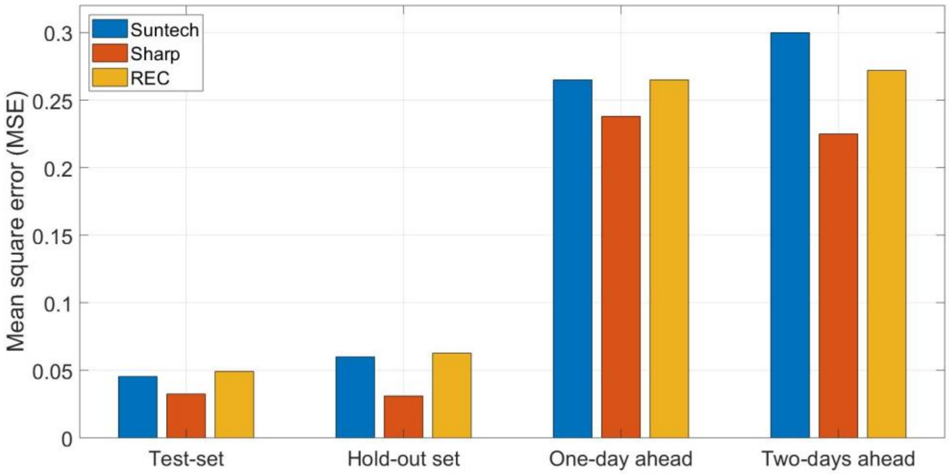


Figure 54: Mean square error (MSE) comparison of all PV modules

Based on Figure 54, Sharp PV modules show an overall lower mean square error compared to the other two PV modules, for all sets the models were tested on. The test set and hold out set have comparable errors for all PV modules, as for the one-day ahead and two-days ahead forecast horizon. Sharp has a lower rated power and is made of thin film material, which provides the module with different technical aspect compared to Suntech and REC, which is of multicrystalline silicon. Whether this is an impact to why the model has lower error when based on the Sharp modules cannot be decided based only on these results but is an interesting observation for the PV modules located at the same location.

One other aspect to consider is how the data division was performed for each of the PV modules. As the different subsets (training, validation and test set) may include different hours for the different PV modules (meaning the test set for Suntech most likely does not include the same hour sample as REC or Sharp), the results from the test set may not be directly comparable. To make a fully comparable investigation of the model performances, additional analysis of the raw data should be performed, or a comparison of i.e. the hold-out set could be more representable. It was seen in subsection 4.7.4, when analyzing input data for days in March and April 2017 that snow cover most likely was an aspect that could cause larger prediction errors. This could be the case for several days, and where these deviations can be divided into different subsets for each of the PV modules.

5. CONCLUSION

A dataset from a PV system located at Kjøita, Kristiansand was used to train and test ANN models. The raw data from the measurements on-site involved several uncertainties, thus a thoroughly review of a maintenance log was performed. The remaining data was divided into subsets used for training, validation and testing purposes.

Principal component analysis (PCA) indicated that the variables wind speed, air temperature and global horizontal were needed to explain the variance and the average mutual information (AMI) indicated no time lag influence. A model-based approach was used when including air mass to the input variables, as this appeared to improve the overall precision of the prediction models.

Seven different tests were compared, where each test included a model with a various number of hidden layers and neurons. Based on these tests, all three types of PV modules indicated that a configuration of 20-2-1 would provide the lowest mean square error (MSE) in the test set. Suntech obtained an MSE=0.0454, Sharp MSE=0.0325 and REC MSE=0.0492 using the test set. All R^2-values from test set indicated values higher than 0.95, meaning the regression of the predicted values versus the target values had a good fit.

The data division for the three PV modules was performed using coefficient of variation dividing into subsets training, validation and test. It was however observed similar patterns of mispredictions in the three sets. Due to these errors, mean absolute relative error (MARE) would show high values for all PV modules (but depending on subset) as it is sensitive to larger deviations. Larger deviations did for some occasions appear to be caused by high wind speed, while other deviations appear to occur because of snow cover which was not included in the ANN models. From a hold-out set it could be observed higher errors for Suntech and REC, while lower errors for Sharp compared to the test sets. Sharp had an overall better performance for all tests performed compared to the other two types of PV modules.

Testing the models with data from MEPS (MetCoOp Ensemble Prediction System) weather forecast, showed that the ANN models can use the input data provided, but have MSE values higher than for the test and hold-out set, which was expected. This appeared to be mostly due to uncertainties in the weather forecasts. The tests were based on weather predictions from one-day ahead and two-days ahead forecast horizons, where the one-day ahead gave a lower MSE for Suntech and REC, but the two-day ahead have lower MSE for Sharp.

Based on these findings, it can be concluded that ANN models for Kjøita PV system with three different types of PV modules show a potential of predicting PV power. The configurations with 20-2-1 with the relevant input selection used in this work, did show that for some occasion’s the predictions had relatively high deviations from the target values, while for other occasions were able to do predictions close to the target. The predictions performed with forecasted data did indicate that the models were able to use this data, and a potential can be found in using ANN models for power predictions. More precise prediction models can contribute with useful information to grid operators as an increase of installed PV power can be seen every year and is assumed to continue to increase for coming years. Large grid-connected PV systems or several smaller systems will contribute to feed the grid with fluctuating power, especially during days with changing weather conditions.

5.1. Further Work

Studying some of the prediction results with high deviations to the target values, high wind speed can be observed and how the models treat this value should be investigated further. As a principal component analysis may not always provide the optimal variables to include in an ANN model, due to it only investigates the variance of the dataset not the direct affect to the output (target), additional analysis will be necessary. Also, the air temperature was used in this work instead of the module temperature. Another attempt with using the module temperature should be performed to see if this can contribute to more precise predictions as the module temperature can have a direct effect on the efficiency of the modules. This would, however, involve having to convert air temperature to module temperature as the weather forecasts provide air temperature. One last input parameter that can be of interest to include is the snow cover. It was observed larger deviations for some of the predictions compared to the targets, which presumably was because of snow covering the PV modules. Including this variable can contribute to lower these deviations. Further optimization of ANN models can also be compared with a linear model, to investigate difference in performances.

The models created for the purpose of this thesis include data from a 3-year period. Over time, degradation of the PV cells can occur, and as the feedforward network did not consider time-series, this degradation can influence the prediction results. As the degradation will only have limited effect on the output, this can be of importance if a model based on a longer time period is considered.

The method for testing the models with different configurations was based on a trial and error model-based approach by having seven different tests adjusting the hidden neurons and hidden layers of the models. A possibility to test models with several different parameters should be performed. This way more hyperparameters can be tested against each other and a more throughout decision of model configurations can be obtained. This can be done with the use of a grid search approach, which tests several different hyperparameters at the time.

Another interesting aspect to look further into is how this model can be used with other PV systems. As the models have been trained specifically to this system, a more dynamic model that can be used with an ensemble of systems to be able to see the overall grid influence for an entire area could be relevant.

With these points, the following suggestions to improve an ANN model of the purpose of prediction power production from the PV modules located at Kjøita are:

- Excluding wind speed as input selection, using module temperature and/or snow cover
- Additional data filtering and/or data mining can introduce further improvement of the results, thus obtaining a more precise prediction model.
- Including a degradation model, or time series that takes the degradation into account
- Grid search approach
- Expand the ANN model to deal with an ensemble of PV systems within same area

LIST OF REFERENCES

[1] L. Henden and T. Ericson, "Teknologianalyser 2018," in "FAKTA," NVE 2019, Available: http://publikasjoner.nve.no/faktaark/2019/faktaark2019_01.pdf.

[2] M. Ding, L. Wang, and R. Bi, "An ANN-based Approach for Forecasting the Power Output of Photovoltaic System," *Procedia Environmental Sciences,* vol. 11, pp. 1308-1315, 2011/01/01/ 2011.

[3] G. Zhang, B. Eddy Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks:: The state of the art," *International Journal of Forecasting,* vol. 14, no. 1, pp. 35-62, 1998/03/01/ 1998.

[4] C. Lie, "Characterization of Solar PV System Located in Kjøita, Kristiansand," M.S. Research energy project, Agder University, 2018.

[5] J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F. J. Martinez-de-Pison, and F. Antonanzas-Torres, "Review of photovoltaic power forecasting," *Solar Energy,* vol. 136, pp. 78-111, 2016/10/15/ 2016.

[6] M. Kudo, A. Takeuchi, Y. Nozaki, H. Endo, and J. Sumita, "Forecasting electric power generation in a photovoltaic power system for an energy network," *Electrical engineering in Japan,* vol. 167, pp. 16-23, 2009.

[7] E. Lorenz, T. Scheidsteger, J. Hurka, D. Heinemann, and C. Kurz, "Regional PV power prediction for improved grid integration," *Progress in photovoltaics research and application,* vol. 19, pp. 757-771, 2011.

[8] M. Omar, A. Dolara, G. Magistrati, M. Mussetta, E. Ogliari, and F. Viola, "Day-ahead forecasting for photovoltaic power using artificial neural networks ensembles," in *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*, 2016, pp. 1152-1157.

[9] L. Liu, D. Liu, Q. Sun, H. Li, and R. Wennersten, "Forecasting Power Output of Photovoltaic System Using A BP Network Method," *Energy Procedia,* vol. 142, pp. 780-786, 2017/12/01/ 2017.

[10] S. H. Oudjana, A. Hellal, and I. H. Mahammed, "Power Forecasting of Photovoltaic Generation," *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering,* vol. 7, 2013.

[11] J. K. Kaldellis, M. Kapsali, and K. A. Kavadias, "Temperature and wind speed impact on the efficiency of PV installations. Experience obtained from outdoor measurements in Greece," *Renewable Energy,* vol. 66, pp. 612-624, 2014/06/01/ 2014.

[12] S. I. Sulaiman, T. K. Abdul Rahman, I. Musirin, and S. Shaari, "Artificial neural network versus linear regression for predicting Grid-Connected Photovoltaic system output," in *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2012, pp. 170-174.

[13] Wikimedia Commons. (2013, 15.03.2019). *Solar spectrum* [Online]. Available: https://commons.wikimedia.org/wiki/File:Solar_spectrum_en.svg

[14] S. R. Wenham, M. A. Green, M. E. Watt, R. Corkish, and A. Sproul, *Applied Photovoltaics*, 3. ed. Abigon, Oxon: Earthscan, 2011.

[15] C. Honsberg and B. Bowden. (2019, 22.04.2019). *Module Materials* [Online]. Available: https://www.pveducation.org/pvcdrom/modules-and-arrays/module-materials

[16] P. A. Basore, "Defining terms for crystalline silicon solar cells," ed. [Chichester, West Sussex, England] :, 1994, pp. 177-179.

[17] Wikimedia Commons. (2008, 18.04.2019). *I-V Curve T* [Online]. Available: https://commons.wikimedia.org/wiki/File:I-V_Curve_T.png

[18] E. Rich and K. Knight, *Artificial intelligence*, 2. ed. McGraw-Hill, 1991.

[19] W. Ertel, *Introduction to Artificial Intelligence*, 2. ed. Springer London, 2017.

[20] R. Hoehndorf and N. Queralt-Rosinach, "Data Science and symbolic AI: Synergies, challenges and opportunities," *Data Science,* vol. 1, no.1-2, pp. 27-38, 2017.

[21] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, 2014, p. 640.

[22] S. Theodoridis, "Chapter 1 - Introduction," in *Machine Learning* Oxford: Academic Press, 2015, pp. 1-8.

[23] MathWorks. (n.a., 26.03.2019). *Machine Learning in MATLAB* [Online]. Available: https://se.mathworks.com/help/stats/machine-learning-in-matlab.html

[24] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks,* vol. 61, pp. 85-117, 2015/01/01/ 2015.

[25] H. H. Bock, "Clustering Methods: A History of k-Means Algorithms," in *Selected Contributions in Data Analysis and Classification*Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 161-172.

[26] MathWorks. (n.a., 11.05.2019). *Multilayer Shallow Neural Network Architecture* [Online]. Available: https://se.mathworks.com/help/deeplearning/ug/multilayer-neural-network-architecture.html

[27] MathWorks. (n.a., 30.03.2019). *trainlm* [online]. Available: https://se.mathworks.com/help/deeplearning/ref/trainlm.html

[28] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Feedforward Networks," in *Deep Learning*: MIT Press, 2016, pp. 164-223.

[29] Wikimedia Commons. (2015, 11.03.2019). *Neural Network* [Online]. Available: https://commons.wikimedia.org/wiki/Neural_network

[30] H. R. Maier and A. Jain, "Methods used for the development of neural networks for the prediction of water resource variables in river systems: Current status and future directions," *Environmental Modelling and Software,* vol. 25, no. 8, pp. 891-909, 2010.

[31] S. Haykin, *Neural Networks and Learning Machines*, 3. ed. New Jersey, USA: Pearson Education, 2009.

[32] Agder Energi, Collaboration with Teknova, University og Agder, Agder Energi Nett, Elkem Solar, and Eltek, "Solar electricity to grid (Solstrøm på nett)," ed: Funded by the Regional Research Funds (RFF Agder), Project period 2010-2012 (Phase 1) and 2012-2014 (Phase 2).

[33] The MetCoOp Team, "The MetCoOp Ensable MEPS," *HIRLAM Newsletter No. 8,* pp. 98-103, 2017.

[34] M. Mahooti. (2016, 18.02.2019). *Sunrise Sunset* [Online]. Available: https://se.mathworks.com/matlabcentral/fileexchange/55312-sunrise-sunset

[35] M. Brems. (2017, 06.03.2019). *A One-Stop Shop for Principal Component Analysis* [Online]. Available: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

[36] MathWorks. (n.a., 21.03.2019). *pca* [Online]. Available: https://se.mathworks.com/help/stats/pca.html

[37] R. D. Thomas, N. C. Moses, E. A. Semple, and A. J. Strang, "An efficient algorithm for the computation of average mutual information: Validation and implementation in Matlab," *Journal of Mathematical Psychology,* vol. 61, pp. 45-59, 2014/08/01/ 2014.

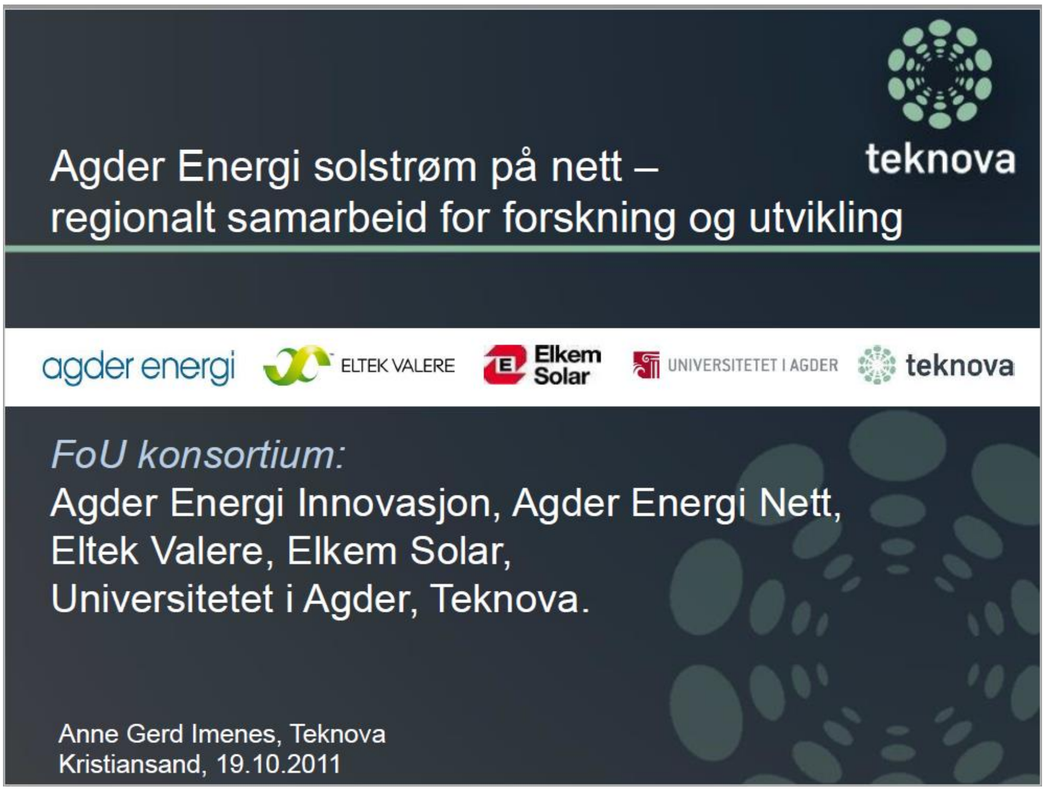
[38] E. G. L. Miller, "Entropy and Mutual Information," Department of Computer Science, University of Massachusetts, Amherst, 2013.

[39] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*. Boston, MA: PWS Publishing, 1996.

APPENDICES

A.1 PROJECT INFORMATION "SOLSTRØM PÅ NETT»

Presentasjon av prosjektet Solstrøm på nett hos Agder Energi, Kristiansand. Presentasjon av Anne Gerd Imenes, Teknova AS, 19/10-2011.



teknova

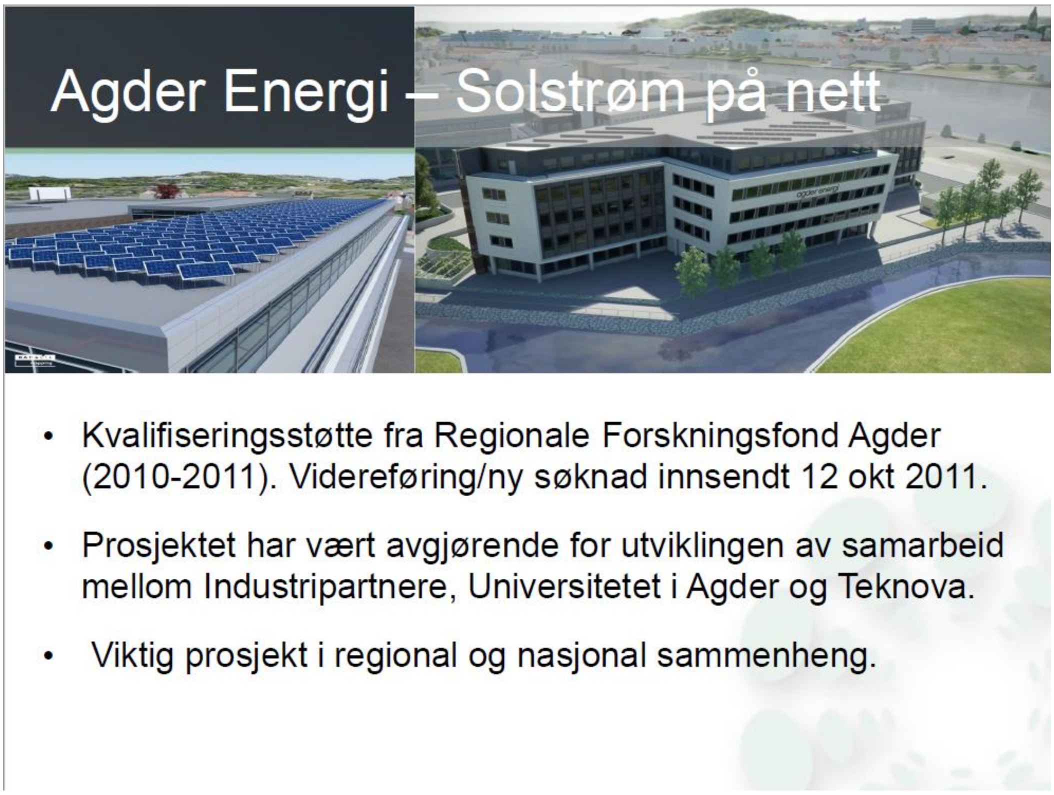
Agder Energi solstrøm på nett –
regionalt samarbeid for forskning og utvikling

agder energi ELTEK VALERE Elkem Solar UNIVERSITETET I AGDER teknova

FoU konsortium:

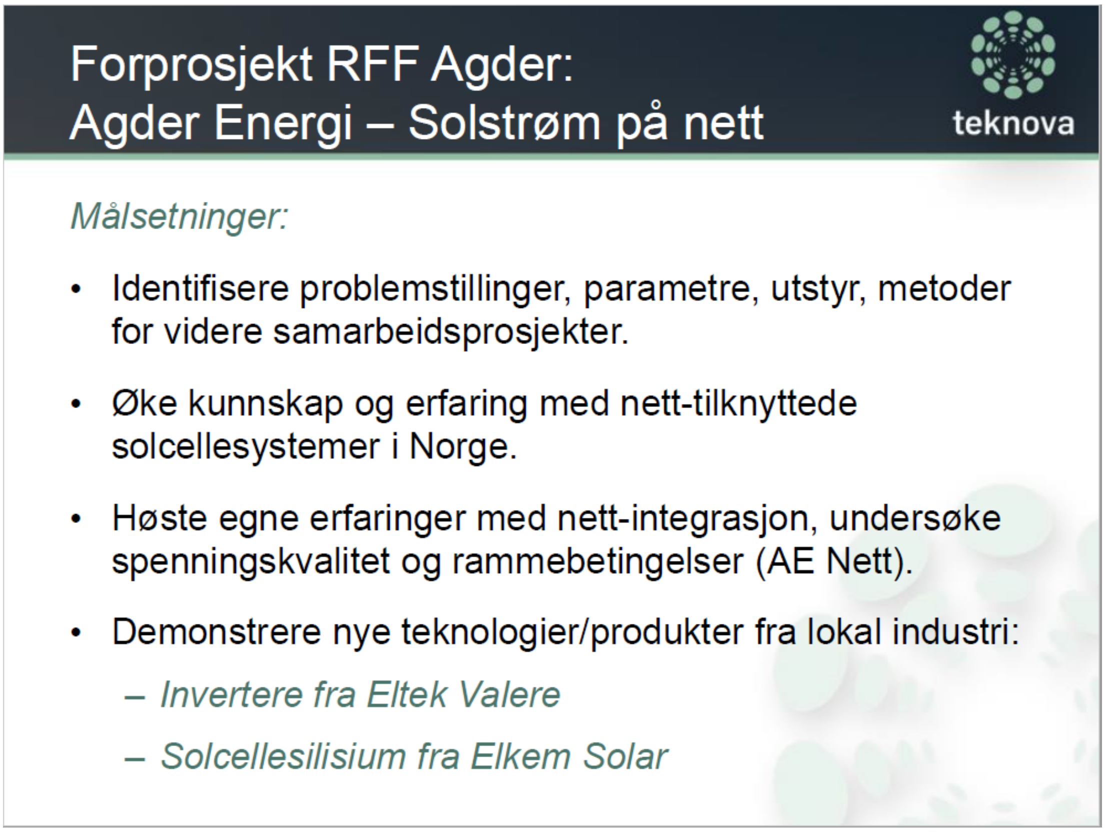
Agder Energi Innovasjon, Agder Energi Nett,
Eltek Valere, Elkem Solar,
Universitetet i Agder, Teknova.

Anne Gerd Imenes, Teknova
Kristiansand, 19.10.2011



Agder Energi – Solstrøm på nett

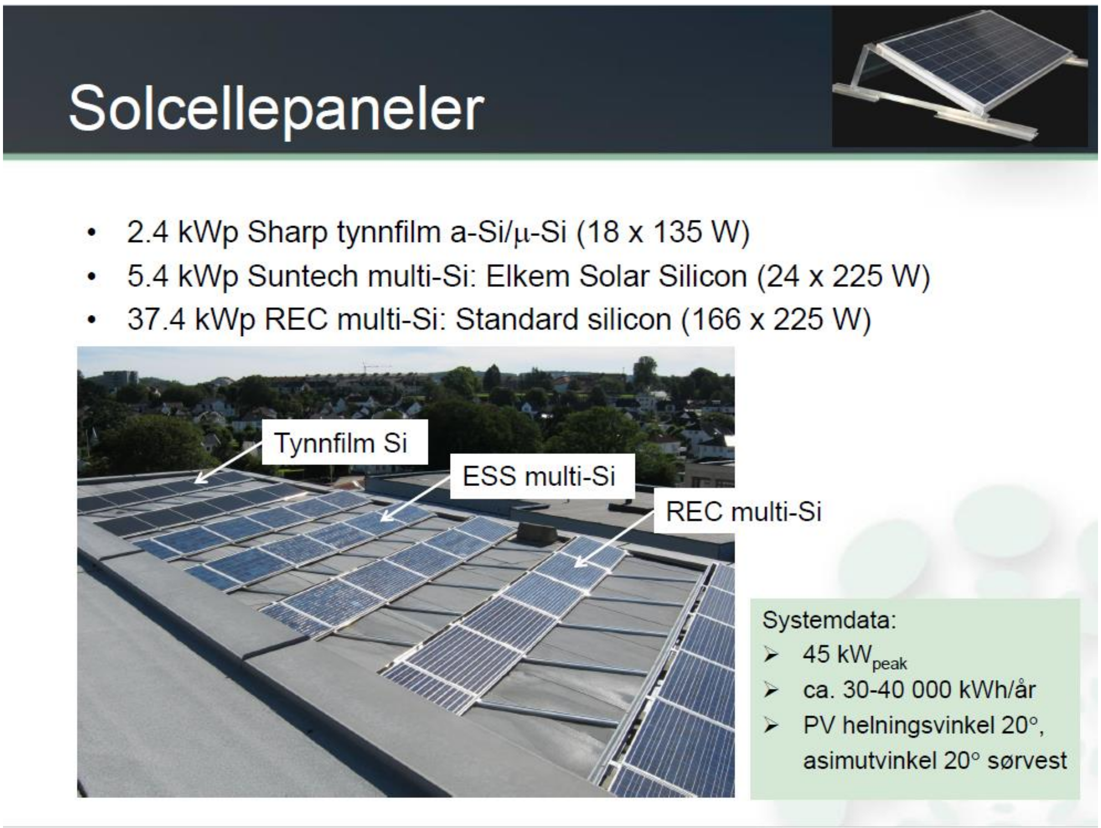
- Kvalifiseringsstøtte fra Regionale Forskningsfond Agder (2010-2011). Videreføring/ny søknad innsendt 12 okt 2011.
- Prosjektet har vært avgjørende for utviklingen av samarbeid mellom Industripartnere, Universitetet i Agder og Teknova.
- Viktig prosjekt i regional og nasjonal sammenheng.



Forprosjekt RFF Agder: Agder Energi – Solstrøm på nett

Målsetninger:

- Identifisere problemstillinger, parametre, utstyr, metoder for videre samarbeidsprosjekter.
- Øke kunnskap og erfaring med nett-tilknyttede solcellesystemer i Norge.
- Høste egne erfaringer med nett-integrasjon, undersøke spenningskvalitet og rammebetingelser (AE Nett).
- Demonstrere nye teknologier/produkter fra lokal industri:
  - *Invertere fra Eltek Valere*
  - *Solcellesilisium fra Elkem Solar*

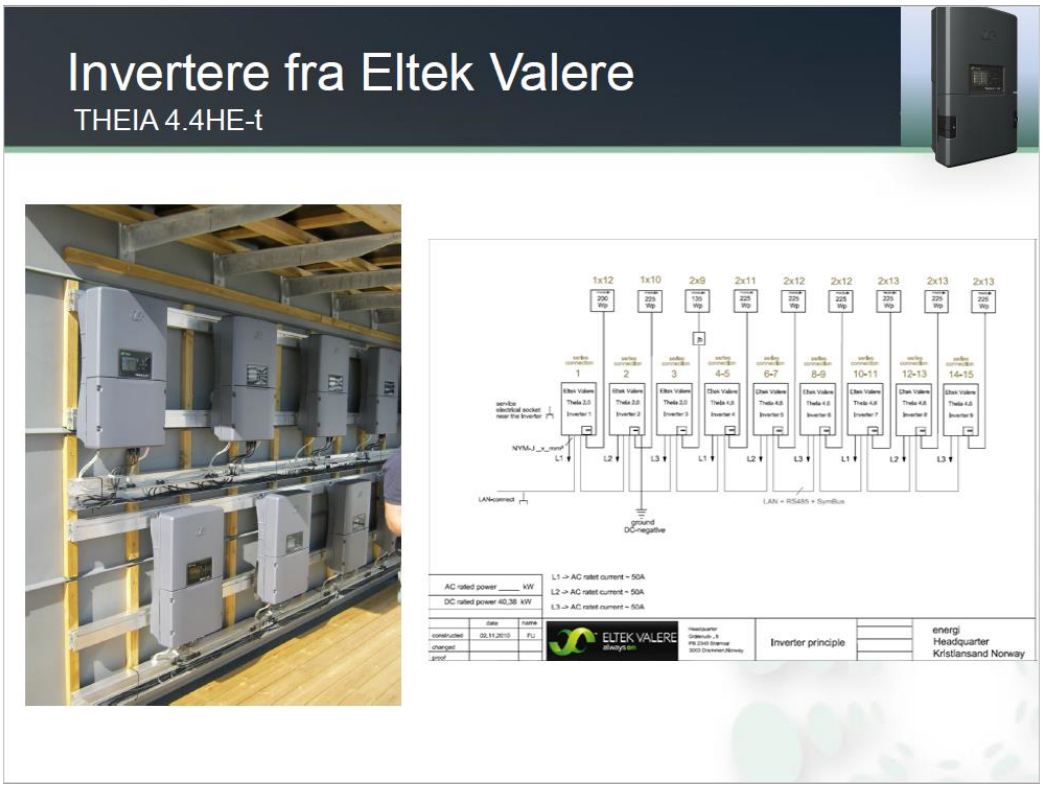


Solcellepaneler

- 2.4 kWp Sharp tynnfilm a-Si/μ-Si (18 x 135 W)
- 5.4 kWp Suntech multi-Si: Elkem Solar Silicon (24 x 225 W)
- 37.4 kWp REC multi-Si: Standard silicon (166 x 225 W)

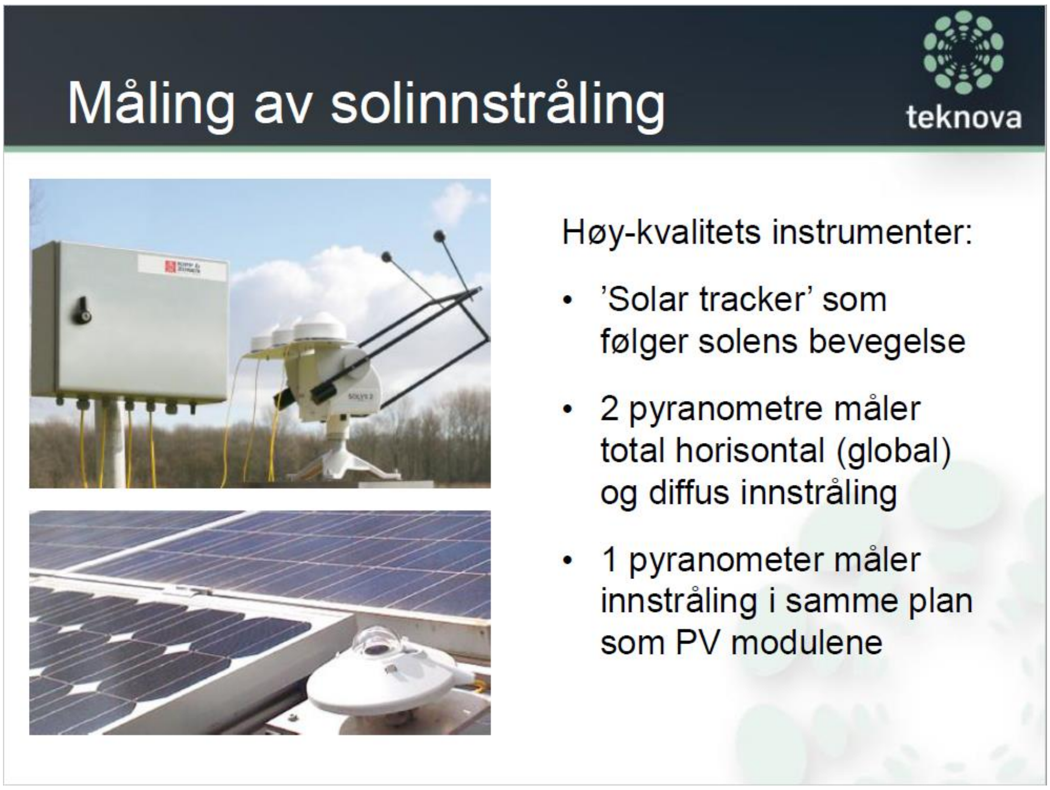
Systemdata:

- 45 kW_{peak}
- ca. 30-40 000 kWh/år
- PV helningsvinkel 20°, asimutvinkel 20° sørvest



Invertere fra Eltek Valere

THEIA 4.4HE-t

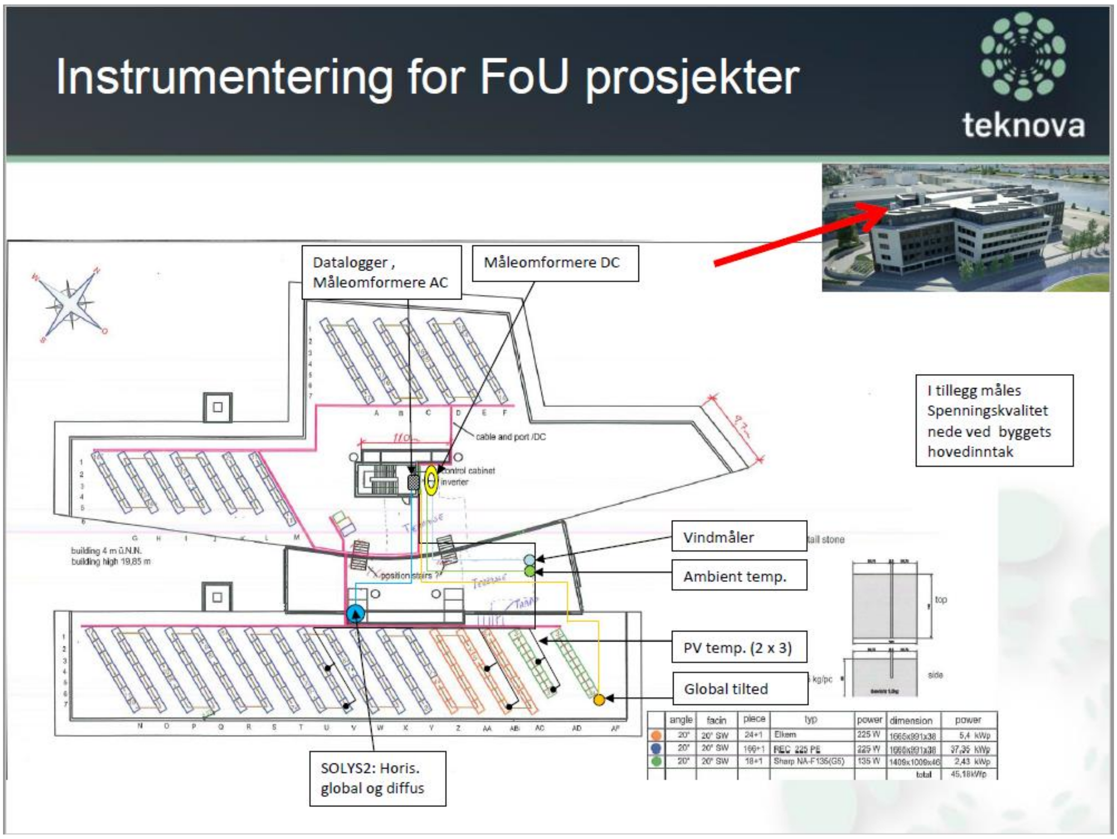


Måling av solinnstråling

teknova

Høy-kvalitets instrumenter:

- ’Solar tracker’ som følger solens bevegelse
- 2 pyranometre måler total horisontal (global) og diffus innstråling
- 1 pyranometer måler innstråling i samme plan som PV modulene



Instrumentering for FoU prosjekter

teknova

Datalogger , Måleomformere AC

Måleomformere DC

I tillegg måles Spenningskvalitet nede ved byggets hovedinntak

Vindmåler

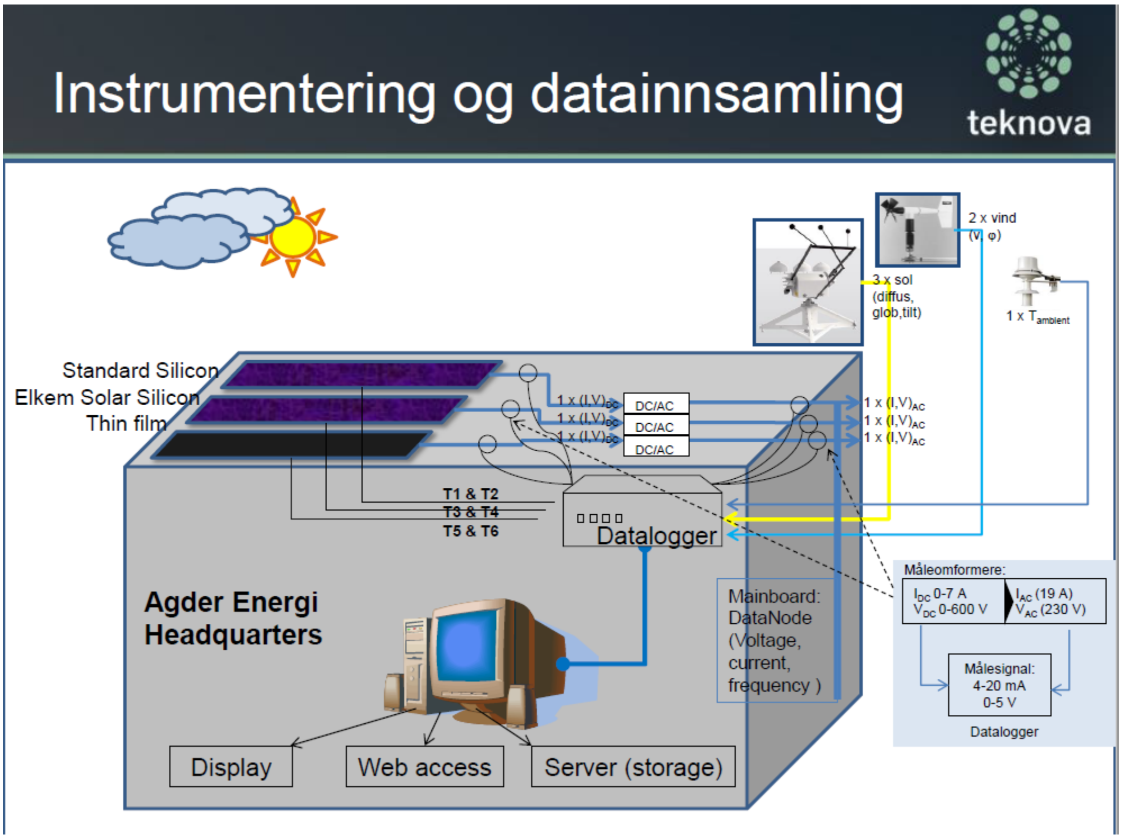
Ambient temp.

PV temp. (2 x 3)

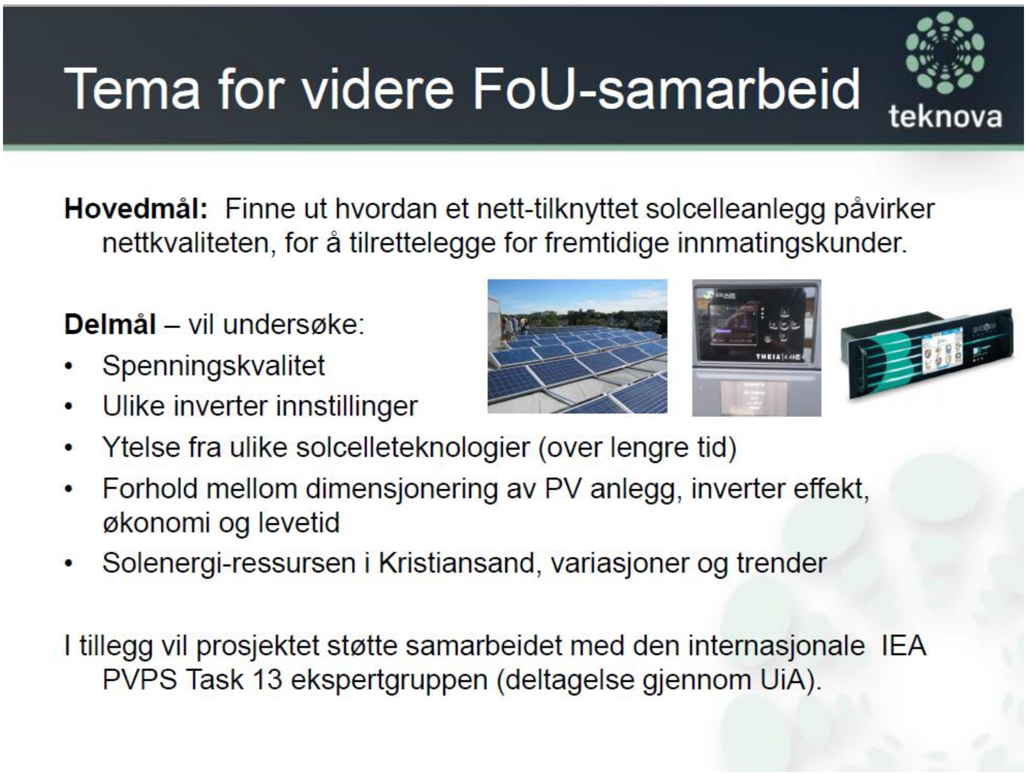
Global tilted

SOLYS2: Horis. global og diffus

	angle	facin	piece	typ	power	dimension	power
●	20°	20° SW	24+1	Elkem	225 W	1665x991x38	5,4 kWp
●	20°	20° SW	166+1	REC 225 PE	225 W	1665x991x38	37,35 kWp
●	20°	20° SW	18+1	Sharp NA-F135(G5)	135 W	1409x1009x46	2,43 kWp
						total	45,18kWp



Instrumentering og datainnsamling



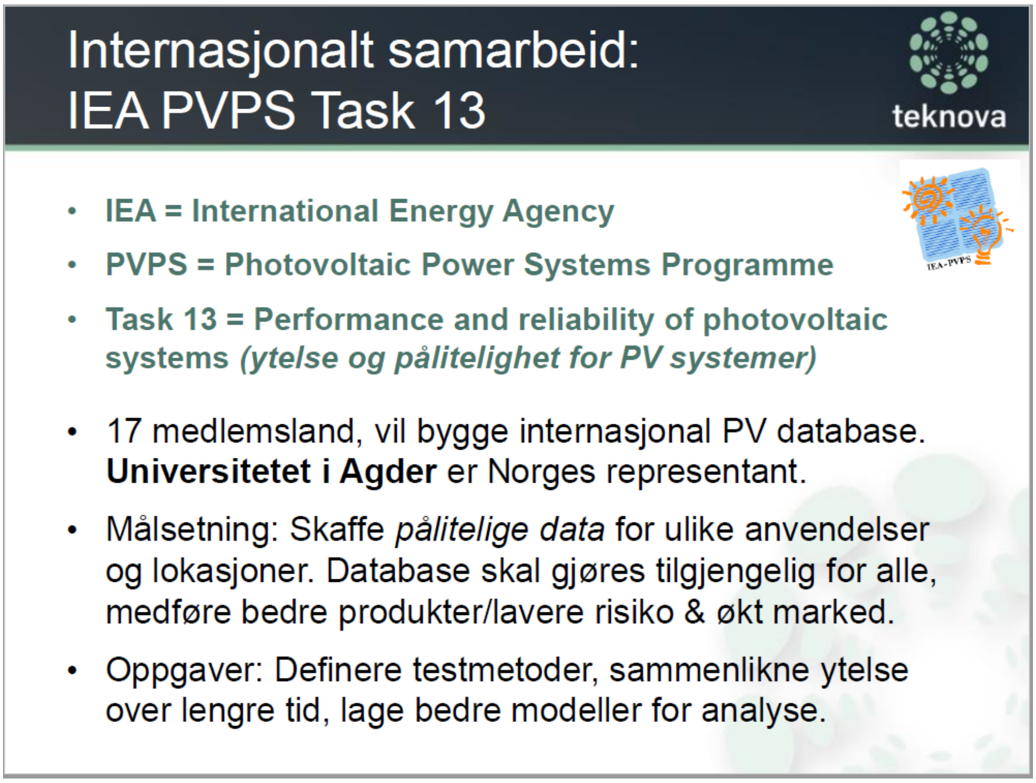
Tema for videre FoU-samarbeid

Hovedmål: Finne ut hvordan et nett-tilknyttet solcelleanlegg påvirker nettkvaliteten, for å tilrettelegge for fremtidige innmatingskunder.

Delmål – vil undersøke:

- Spenningskvalitet
- Ulike inverter innstillinger
- Ytelse fra ulike solcelleteknologier (over lengre tid)
- Forhold mellom dimensjonering av PV anlegg, inverter effekt, økonomi og levetid
- Solenergi-ressursen i Kristiansand, variasjoner og trender

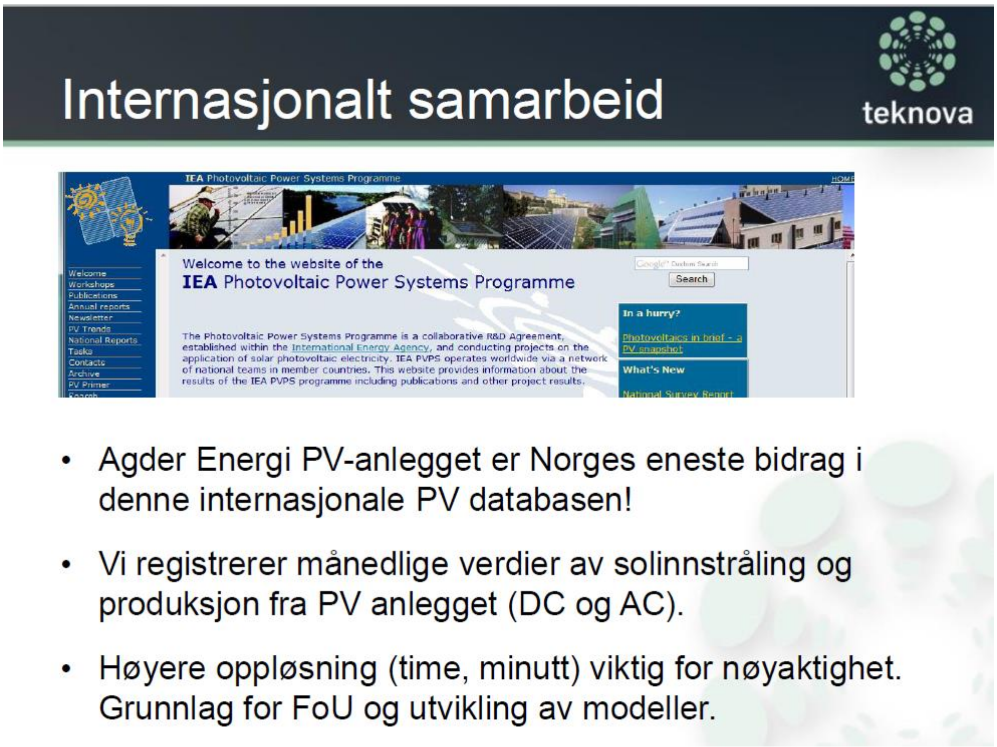
I tillegg vil prosjektet støtte samarbeidet med den internasjonale IEA PVPS Task 13 ekspertgruppen (deltagelse gjennom UiA).


teknova
IEA-PVPS

Internasjonalt samarbeid: IEA PVPS Task 13

- IEA = International Energy Agency
- PVPS = Photovoltaic Power Systems Programme
- Task 13 = Performance and reliability of photovoltaic systems *(ytelse og pålitelighet for PV systemer)*

- 17 medlemsland, vil bygge internasjonal PV database. **Universitetet i Agder** er Norges representant.
- Målsetning: Skaffe *pålitelige data* for ulike anvendelser og lokasjoner. Database skal gjøres tilgjengelig for alle, medføre bedre produkter/lavere risiko & økt marked.
- Oppgaver: Definere testmetoder, sammenlikne ytelse over lengre tid, lage bedre modeller for analyse.

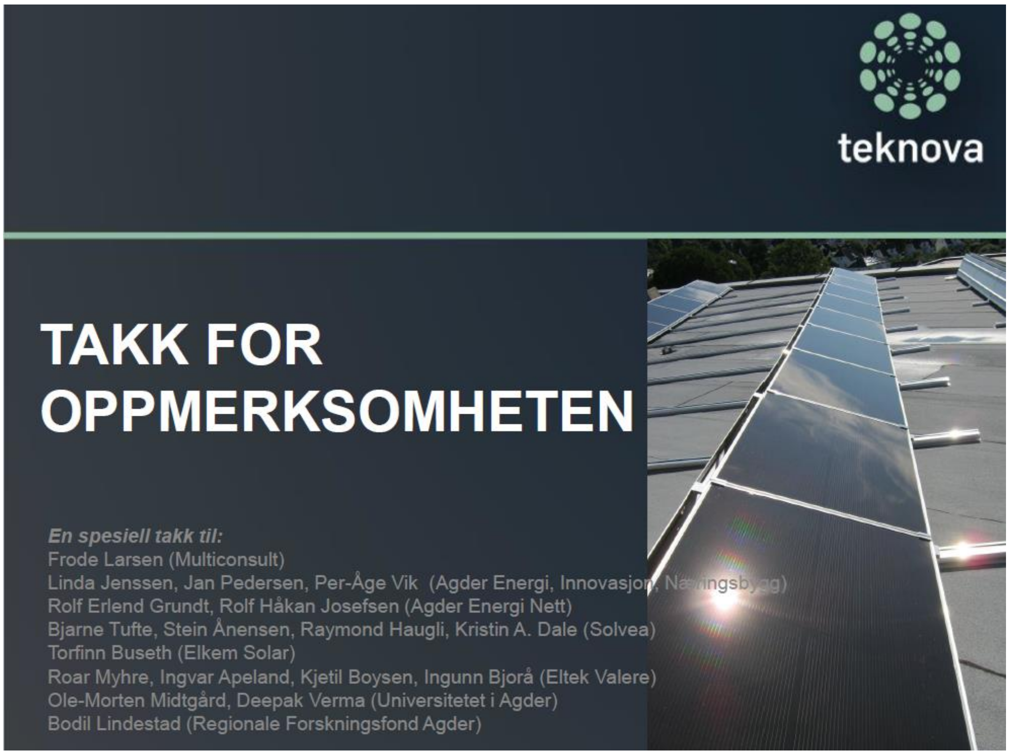


Internasjonalt samarbeid

Welcome to the website of the
IEA Photovoltaic Power Systems Programme

The Photovoltaic Power Systems Programme is a collaborative R&D Agreement, established within the International Energy Agency, and conducting projects on the application of solar photovoltaic electricity. IEA PVPS operates worldwide via a network of national teams in member countries. This website provides information about the results of the IEA PVPS programme including publications and other project results.

- Agder Energi PV-anlegget er Norges eneste bidrag i denne internasjonale PV databasen!
- Vi registrerer månedlige verdier av solinnstråling og produksjon fra PV anlegget (DC og AC).
- Høyere oppløsning (time, minutt) viktig for nøyaktighet. Grunnlag for FoU og utvikling av modeller.

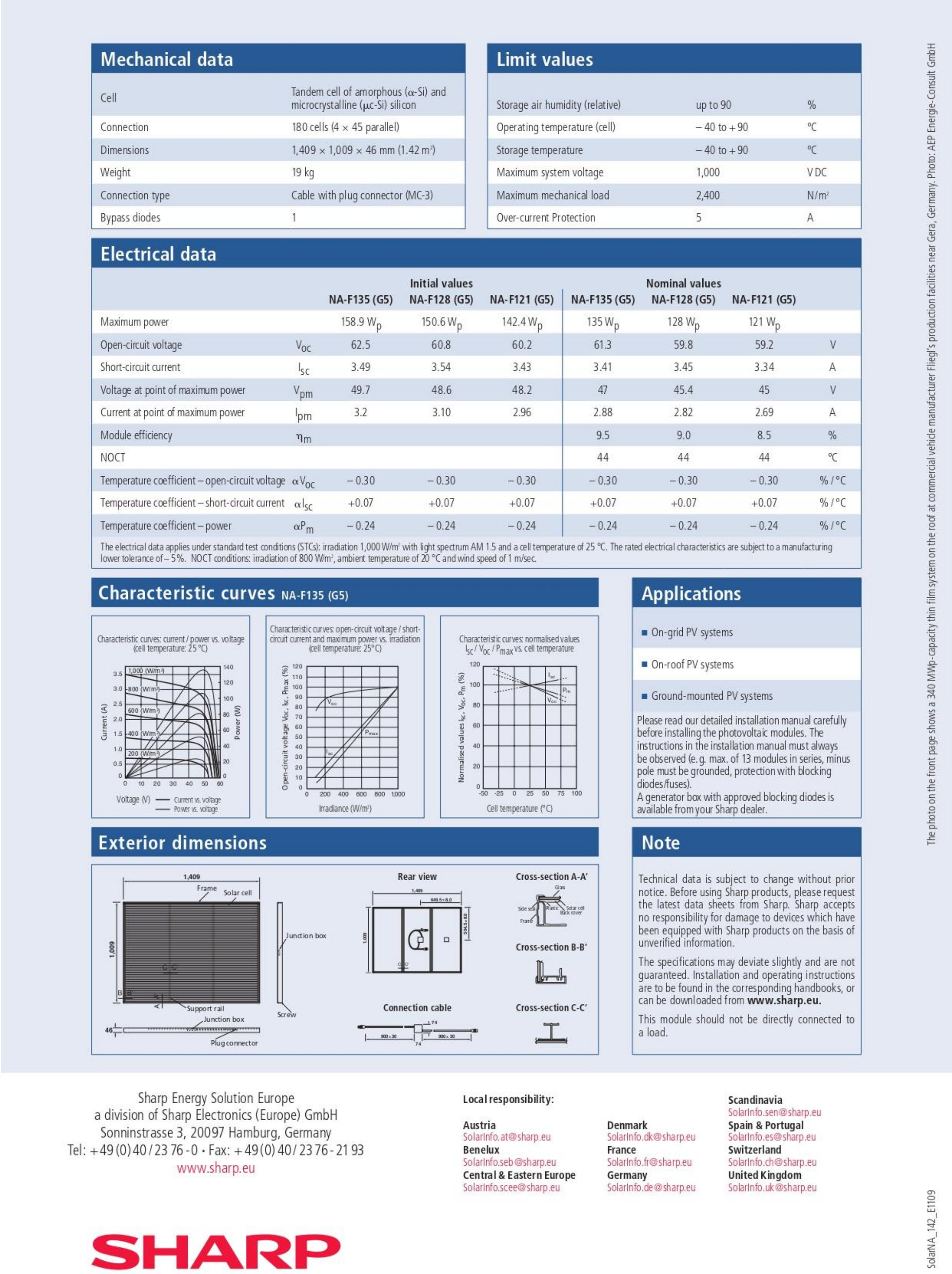


TAKK FOR OPPMERKSOMHETEN

En spesiell takk til:
Frode Larsen (Multiconsult)
Linda Jenssen, Jan Pedersen, Per-Åge Vik (Agder Energi, Innovasjon, Næringsbygg)
Rolf Erlend Grundt, Rolf Håkan Josefsen (Agder Energi Nett)
Bjarne Tufte, Stein Ånensen, Raymond Haugli, Kristin A. Dale (Solvea)
Torfinn Buseth (Elkem Solar)
Roar Myhre, Ingvar Apeland, Kjetil Boysen, Ingunn Bjorå (Eltek Valere)
Ole-Morten Midtgård, Deepak Verma (Universitetet i Agder)
Bodil Lindestad (Regionale Forskningsfond Agder)

A.2 TECHNICAL DATA – PV MODULES

Sharp NA-135 (G5) Datasheet



Mechanical data

Cell	Tandem cell of amorphous (α-Si) and microcrystalline (μc-Si) silicon
Connection	180 cells (4 × 45 parallel)
Dimensions	1,409 × 1,009 × 46 mm (1.42 m²)
Weight	19 kg
Connection type	Cable with plug connector (MC-3)
Bypass diodes	1

Limit values

Storage air humidity (relative)	up to 90	%
Operating temperature (cell)	– 40 to + 90	°C
Storage temperature	– 40 to + 90	°C
Maximum system voltage	1,000	V DC
Maximum mechanical load	2,400	N/m²
Over-current Protection	5	A

Electrical data

		Initial values			Nominal values			
		NA-F135 (G5)	NA-F128 (G5)	NA-F121 (G5)	NA-F135 (G5)	NA-F128 (G5)	NA-F121 (G5)	
Maximum power		158.9 W_p	150.6 W_p	142.4 W_p	135 W_p	128 W_p	121 W_p	
Open-circuit voltage	V_{OC}	62.5	60.8	60.2	61.3	59.8	59.2	V
Short-circuit current	I_{SC}	3.49	3.54	3.43	3.41	3.45	3.34	A
Voltage at point of maximum power	V_{pm}	49.7	48.6	48.2	47	45.4	45	V
Current at point of maximum power	I_{pm}	3.2	3.10	2.96	2.88	2.82	2.69	A
Module efficiency	η_m				9.5	9.0	8.5	%
NOCT					44	44	44	°C
Temperature coefficient – open-circuit voltage	αV_{OC}	– 0.30	– 0.30	– 0.30	– 0.30	– 0.30	– 0.30	% / °C
Temperature coefficient – short-circuit current	αI_{SC}	+0.07	+0.07	+0.07	+0.07	+0.07	+0.07	% / °C
Temperature coefficient – power	αP_m	– 0.24	– 0.24	– 0.24	– 0.24	– 0.24	– 0.24	% / °C

The electrical data applies under standard test conditions (STCs): irradiation 1,000 W/m² with light spectrum AM 1.5 and a cell temperature of 25 °C. The rated electrical characteristics are subject to a manufacturing lower tolerance of – 5 %. NOCT conditions: irradiation of 800 W/m², ambient temperature of 20 °C and wind speed of 1 m/sec.

Characteristic curves NA-F135 (G5)

Characteristic curves: current / power vs. voltage (cell temperature: 25 °C)

Characteristic curves: open-circuit voltage / short-circuit current and maximum power vs. irradiation (cell temperature: 25°C)

Characteristic curves: normalised values I_{SC} / V_{OC} / P_{max} vs. cell temperature

Applications

- On-grid PV systems
- On-roof PV systems
- Ground-mounted PV systems

Please read our detailed installation manual carefully before installing the photovoltaic modules. The instructions in the installation manual must always be observed (e. g. max. of 13 modules in series, minus pole must be grounded, protection with blocking diodes/fuses).
A generator box with approved blocking diodes is available from your Sharp dealer.

Exterior dimensions

Note

Technical data is subject to change without prior notice. Before using Sharp products, please request the latest data sheets from Sharp. Sharp accepts no responsibility for damage to devices which have been equipped with Sharp products on the basis of unverified information.

The specifications may deviate slightly and are not guaranteed. Installation and operating instructions are to be found in the corresponding handbooks, or can be downloaded from **www.sharp.eu.**

This module should not be directly connected to a load.

The photo on the front page shows a 340 MWp-capacity thin film system on the roof at commercial vehicle manufacturer Fliegl's production facilities near Gera, Germany. Photo: AEP Energie-Consult GmbH

Sharp Energy Solution Europe
a division of Sharp Electronics (Europe) GmbH
Sonninstrasse 3, 20097 Hamburg, Germany
Tel: + 49 (0) 40 / 23 76 - 0 · Fax: + 49 (0) 40 / 23 76 - 21 93
www.sharp.eu

Local responsibility:

Austria
SolarInfo.at@sharp.eu
Benelux
SolarInfo.seb@sharp.eu
Central & Eastern Europe
SolarInfo.scee@sharp.eu

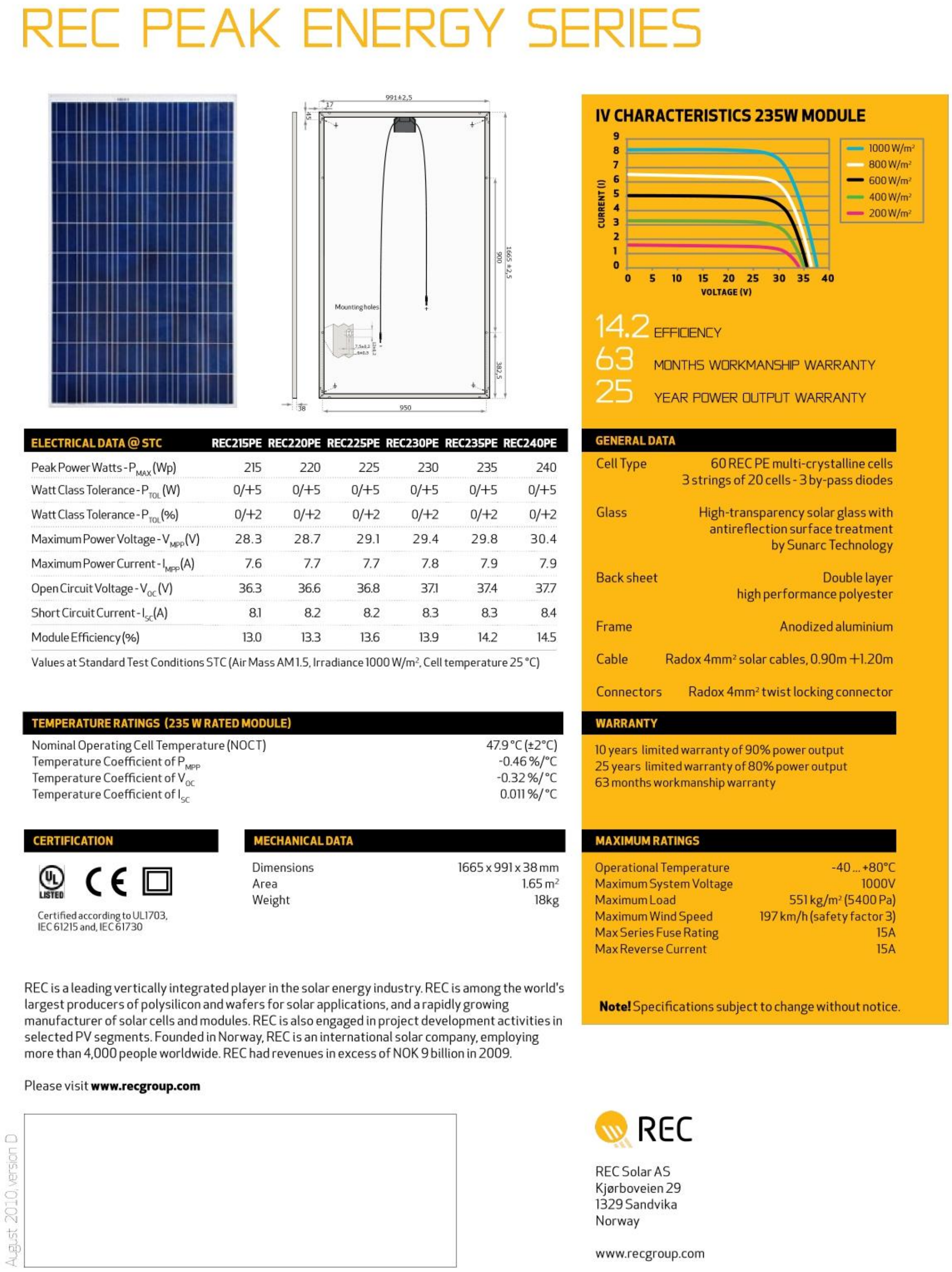
Denmark
SolarInfo.dk@sharp.eu
France
SolarInfo.fr@sharp.eu
Germany
SolarInfo.de@sharp.eu

Scandinavia
SolarInfo.sen@sharp.eu
Spain & Portugal
SolarInfo.es@sharp.eu
Switzerland
SolarInfo.ch@sharp.eu
United Kingdom
SolarInfo.uk@sharp.eu

SHARP

SolarNA_142_E1109

REC 225PE – Datasheet


991±2,5
17
45
1665 ±2,5
900
Mounting holes
7,5±0,2
8±0,5
12±0,2
382,5
38
950
IV CHARACTERISTICS 235W MODULE
CURRENT (I)
0
1
2
3
4
5
6
7
8
9
VOLTAGE (V)
0
5
10
15
20
25
30
35
40
1000 W/m²
800 W/m²
600 W/m²
400 W/m²
200 W/m²
August 2010, version D

REC PEAK ENERGY SERIES

14.2 EFFICIENCY

63 MONTHS WORKMANSHIP WARRANTY

25 YEAR POWER OUTPUT WARRANTY

ELECTRICAL DATA @ STC	REC215PE	REC220PE	REC225PE	REC230PE	REC235PE	REC240PE
Peak Power Watts - P_{MAX} (Wp)	215	220	225	230	235	240
Watt Class Tolerance - P_{TOL} (W)	0/+5	0/+5	0/+5	0/+5	0/+5	0/+5
Watt Class Tolerance - P_{TOL} (%)	0/+2	0/+2	0/+2	0/+2	0/+2	0/+2
Maximum Power Voltage - V_{MPP} (V)	28.3	28.7	29.1	29.4	29.8	30.4
Maximum Power Current - I_{MPP} (A)	7.6	7.7	7.7	7.8	7.9	7.9
Open Circuit Voltage - V_{OC} (V)	36.3	36.6	36.8	37.1	37.4	37.7
Short Circuit Current - I_{SC} (A)	8.1	8.2	8.2	8.3	8.3	8.4
Module Efficiency (%)	13.0	13.3	13.6	13.9	14.2	14.5

Values at Standard Test Conditions STC (Air Mass AM1.5, Irradiance 1000 W/m², Cell temperature 25 °C)

TEMPERATURE RATINGS (235 W RATED MODULE)	
Nominal Operating Cell Temperature (NOCT)	47.9 °C (±2°C)
Temperature Coefficient of P_{MPP}	-0.46 %/°C
Temperature Coefficient of V_{OC}	-0.32 %/°C
Temperature Coefficient of I_{SC}	0.011 %/°C

CERTIFICATION

Certified according to UL1703, IEC 61215 and, IEC 61730

MECHANICAL DATA	
Dimensions	1665 x 991 x 38 mm
Area	1.65 m²
Weight	18kg

GENERAL DATA	
Cell Type	60 REC PE multi-crystalline cells 3 strings of 20 cells - 3 by-pass diodes
Glass	High-transparency solar glass with antireflection surface treatment by Sunarc Technology
Back sheet	Double layer high performance polyester
Frame	Anodized aluminium
Cable	Radox 4mm² solar cables, 0.90m +1.20m
Connectors	Radox 4mm² twist locking connector

WARRANTY

10 years limited warranty of 90% power output
25 years limited warranty of 80% power output
63 months workmanship warranty

MAXIMUM RATINGS	
Operational Temperature	-40 ... +80°C
Maximum System Voltage	1000V
Maximum Load	551 kg/m² (5400 Pa)
Maximum Wind Speed	197 km/h (safety factor 3)
Max Series Fuse Rating	15A
Max Reverse Current	15A

Note! Specifications subject to change without notice.

REC is a leading vertically integrated player in the solar energy industry. REC is among the world's largest producers of polysilicon and wafers for solar applications, and a rapidly growing manufacturer of solar cells and modules. REC is also engaged in project development activities in selected PV segments. Founded in Norway, REC is an international solar company, employing more than 4,000 people worldwide. REC had revenues in excess of NOK 9 billion in 2009.

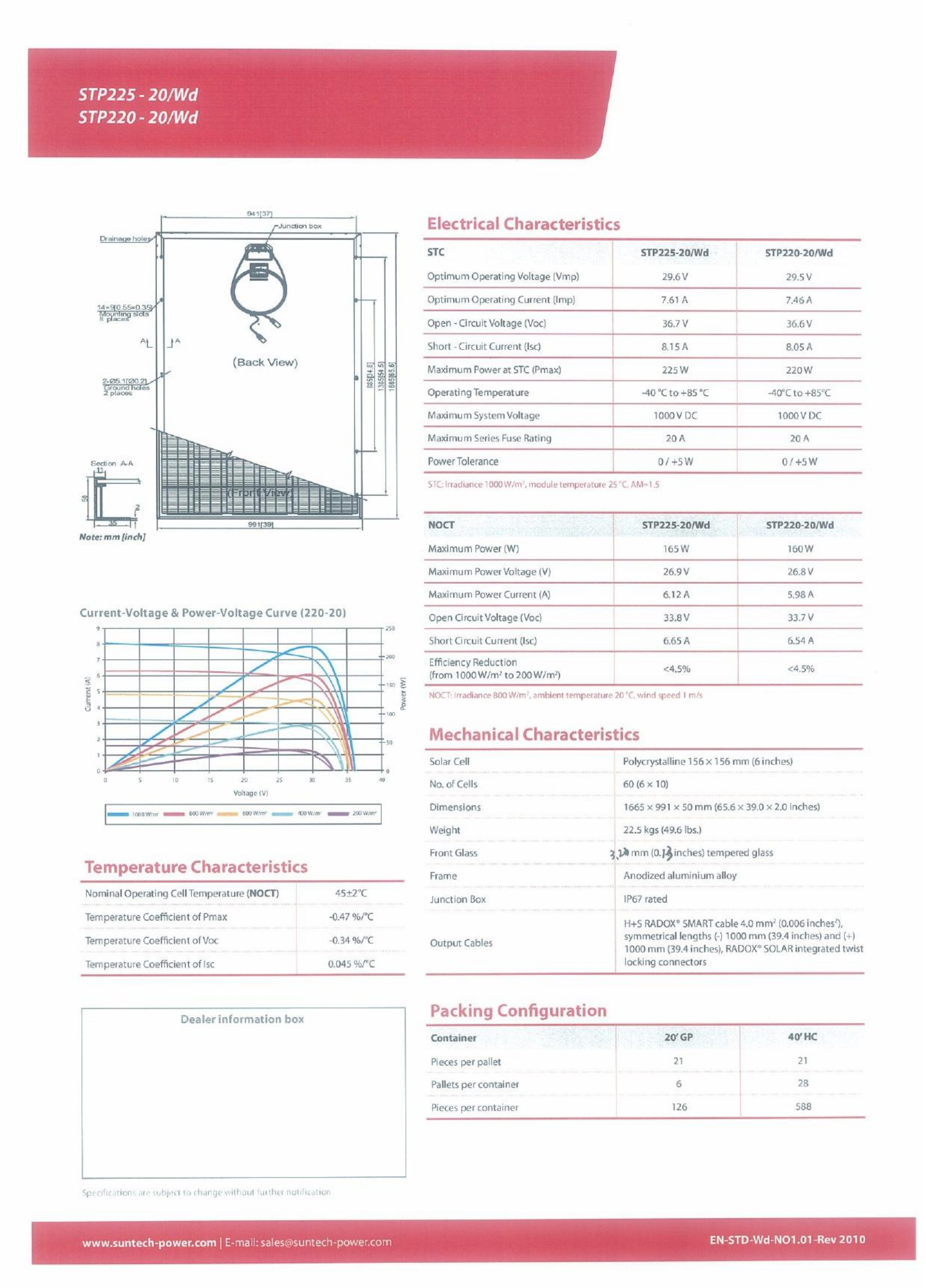
Please visit **www.recgroup.com**

REC

REC Solar AS
Kjørboveien 29
1329 Sandvika
Norway

www.recgroup.com

Suntech STP225 - 20/Wd Datasheet



941[37]
Junction box
Drainage holes
14×9[0.55×0.35]
Mounting slots
8 places
A
A
(Back View)
2-Ø5.1[Ø0.2]
Ground holes
2 places
885[34.8]
1385[54.5]
1665[65.6]
Section A-A
11
50
2
35
(Front View)
991[39]
Note: mm [inch]
9
8
7
6
5
4
3
2
1
0
Current (A)
0
5
10
15
20
25
30
35
40
Voltage (V)
250
200
150
100
50
0
Power (W)
1000 W/m²
800 W/m²
600 W/m²
400 W/m²
200 W/m²

STP225 - 20/Wd
STP220 - 20/Wd

Current-Voltage & Power-Voltage Curve (220-20)

Electrical Characteristics

STC	STP225-20/Wd	STP220-20/Wd
Optimum Operating Voltage (Vmp)	29.6 V	29.5 V
Optimum Operating Current (Imp)	7.61 A	7.46 A
Open - Circuit Voltage (Voc)	36.7 V	36.6 V
Short - Circuit Current (Isc)	8.15 A	8.05 A
Maximum Power at STC (Pmax)	225 W	220 W
Operating Temperature	-40 °C to +85 °C	-40°C to +85°C
Maximum System Voltage	1000 V DC	1000 V DC
Maximum Series Fuse Rating	20 A	20 A
Power Tolerance	0 / +5 W	0 / +5 W

STC: Irradiance 1000 W/m², module temperature 25 °C, AM=1.5

NOCT	STP225-20/Wd	STP220-20/Wd
Maximum Power (W)	165 W	160 W
Maximum Power Voltage (V)	26.9 V	26.8 V
Maximum Power Current (A)	6.12 A	5.98 A
Open Circuit Voltage (Voc)	33.8 V	33.7 V
Short Circuit Current (Isc)	6.65 A	6.54 A
Efficiency Reduction (from 1000 W/m² to 200 W/m²)	<4.5%	<4.5%

NOCT: Irradiance 800 W/m², ambient temperature 20 °C, wind speed 1 m/s

Temperature Characteristics

Nominal Operating Cell Temperature (**NOCT**)	45±2°C
Temperature Coefficient of Pmax	-0.47 %/°C
Temperature Coefficient of Voc	-0.34 %/°C
Temperature Coefficient of Isc	0.045 %/°C

Mechanical Characteristics

Solar Cell	Polycrystalline 156 × 156 mm (6 inches)
No. of Cells	60 (6 × 10)
Dimensions	1665 × 991 × 50 mm (65.6 × 39.0 × 2.0 inches)
Weight	22.5 kgs (49.6 lbs.)
Front Glass	3,2 mm (0.13 inches) tempered glass
Frame	Anodized aluminium alloy
Junction Box	IP67 rated
Output Cables	H+S RADOX® SMART cable 4.0 mm² (0.006 inches²), symmetrical lengths (-) 1000 mm (39.4 inches) and (+) 1000 mm (39.4 inches), RADOX® SOLAR integrated twist locking connectors

Dealer information box

Packing Configuration

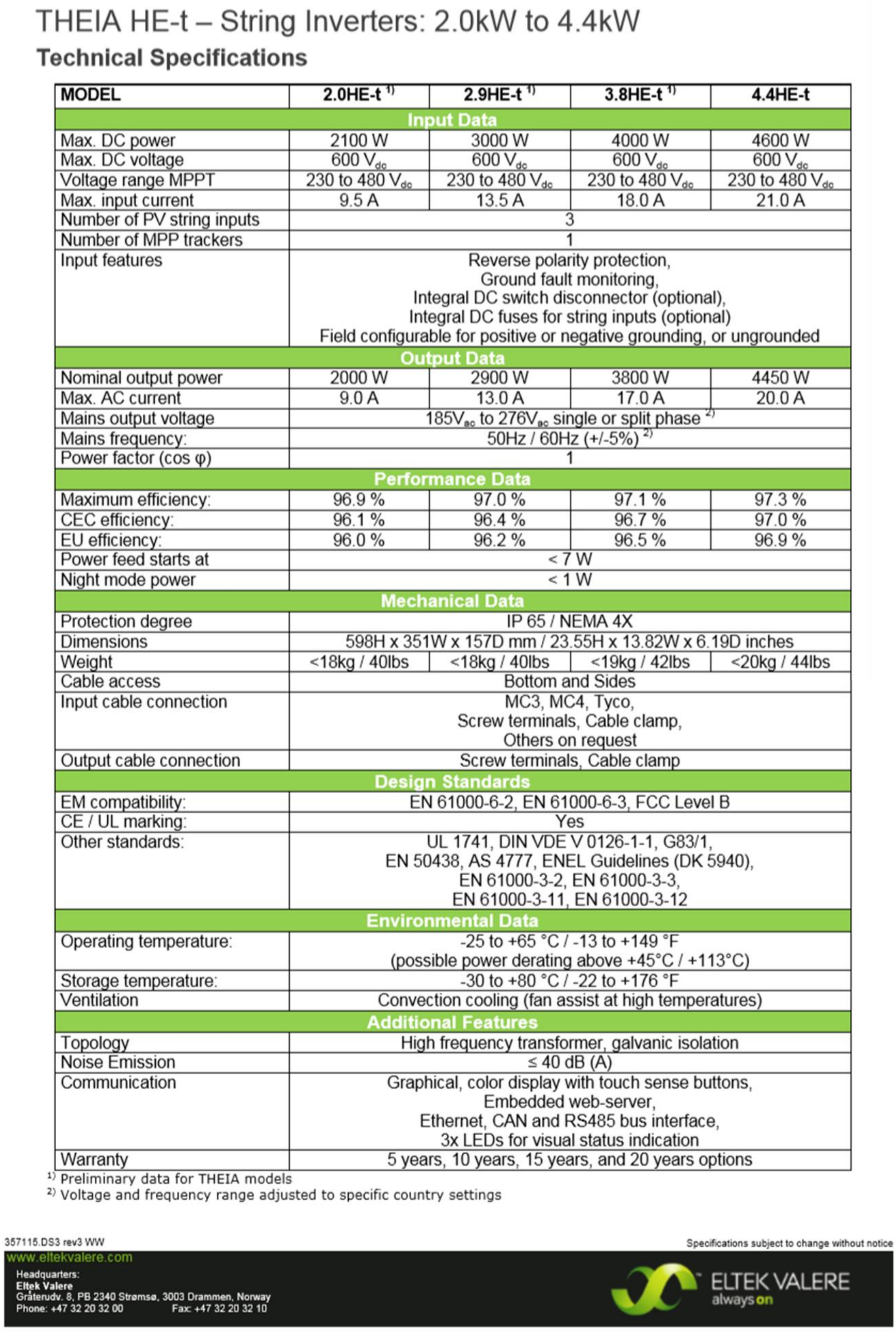
Container	20′ GP	40′ HC
Pieces per pallet	21	21
Pallets per container	6	28
Pieces per container	126	588

Specifications are subject to change without further notification

www.suntech-power.com | E-mail: sales@suntech-power.com

EN-STD-Wd-NO1.01-Rev 2010

A.3 Technical Data – Eltek Inverter



THEIA HE-t – String Inverters: 2.0kW to 4.4kW

Technical Specifications

MODEL	2.0HE-t [1]	2.9HE-t [1]	3.8HE-t [1]	4.4HE-t
Input Data				
Max. DC power	2100 W	3000 W	4000 W	4600 W
Max. DC voltage	600 V_{dc}	600 V_{dc}	600 V_{dc}	600 V_{dc}
Voltage range MPPT	230 to 480 V_{dc}	230 to 480 V_{dc}	230 to 480 V_{dc}	230 to 480 V_{dc}
Max. input current	9.5 A	13.5 A	18.0 A	21.0 A
Number of PV string inputs	3			
Number of MPP trackers	1			
Input features	Reverse polarity protection, Ground fault monitoring, Integral DC switch disconnector (optional), Integral DC fuses for string inputs (optional) Field configurable for positive or negative grounding, or ungrounded			
Output Data				
Nominal output power	2000 W	2900 W	3800 W	4450 W
Max. AC current	9.0 A	13.0 A	17.0 A	20.0 A
Mains output voltage	185V_{ac} to 276V_{ac} single or split phase [2]			
Mains frequency:	50Hz / 60Hz (+/-5%) [2]			
Power factor (cos φ)	1			
Performance Data				
Maximum efficiency:	96.9 %	97.0 %	97.1 %	97.3 %
CEC efficiency:	96.1 %	96.4 %	96.7 %	97.0 %
EU efficiency:	96.0 %	96.2 %	96.5 %	96.9 %
Power feed starts at	< 7 W			
Night mode power	< 1 W			
Mechanical Data				
Protection degree	IP 65 / NEMA 4X			
Dimensions	598H x 351W x 157D mm / 23.55H x 13.82W x 6.19D inches			
Weight	<18kg / 40lbs	<18kg / 40lbs	<19kg / 42lbs	<20kg / 44lbs
Cable access	Bottom and Sides			
Input cable connection	MC3, MC4, Tyco, Screw terminals, Cable clamp, Others on request			
Output cable connection	Screw terminals, Cable clamp			
Design Standards				
EM compatibility:	EN 61000-6-2, EN 61000-6-3, FCC Level B			
CE / UL marking:	Yes			
Other standards:	UL 1741, DIN VDE V 0126-1-1, G83/1, EN 50438, AS 4777, ENEL Guidelines (DK 5940), EN 61000-3-2, EN 61000-3-3, EN 61000-3-11, EN 61000-3-12			
Environmental Data				
Operating temperature:	-25 to +65 °C / -13 to +149 °F (possible power derating above +45°C / +113°C)			
Storage temperature:	-30 to +80 °C / -22 to +176 °F			
Ventilation	Convection cooling (fan assist at high temperatures)			
Additional Features				
Topology	High frequency transformer, galvanic isolation			
Noise Emission	≤ 40 dB (A)			
Communication	Graphical, color display with touch sense buttons, Embedded web-server, Ethernet, CAN and RS485 bus interface, 3x LEDs for visual status indication			
Warranty	5 years, 10 years, 15 years, and 20 years options			

[1] Preliminary data for THEIA models

[2] Voltage and frequency range adjusted to specific country settings

357115.DS3 rev3 WW

Specifications subject to change without notice

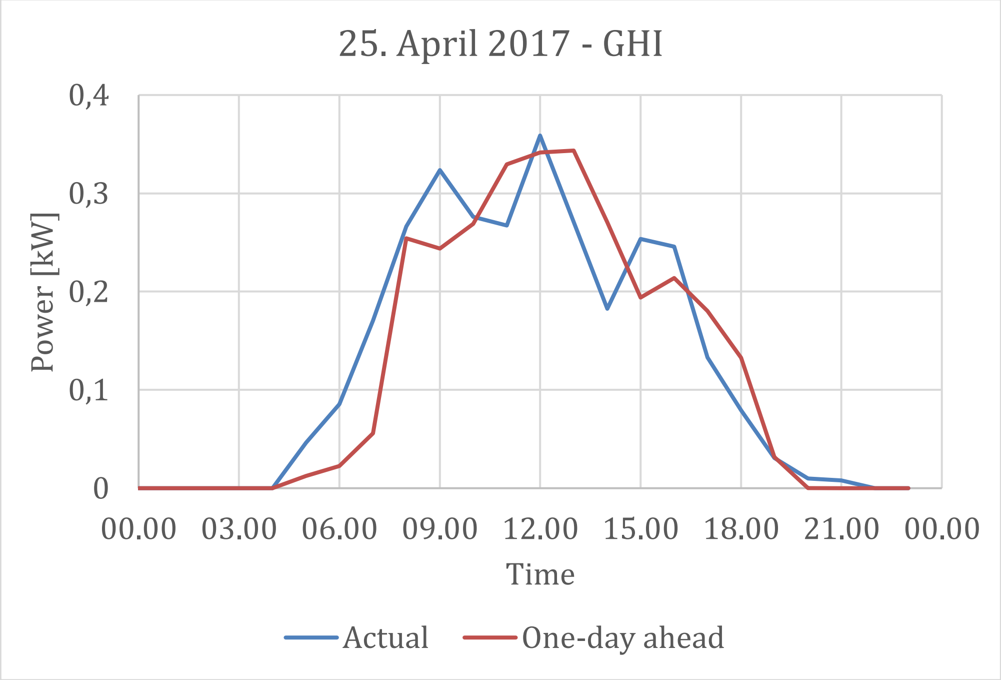
www.eltekvalere.com

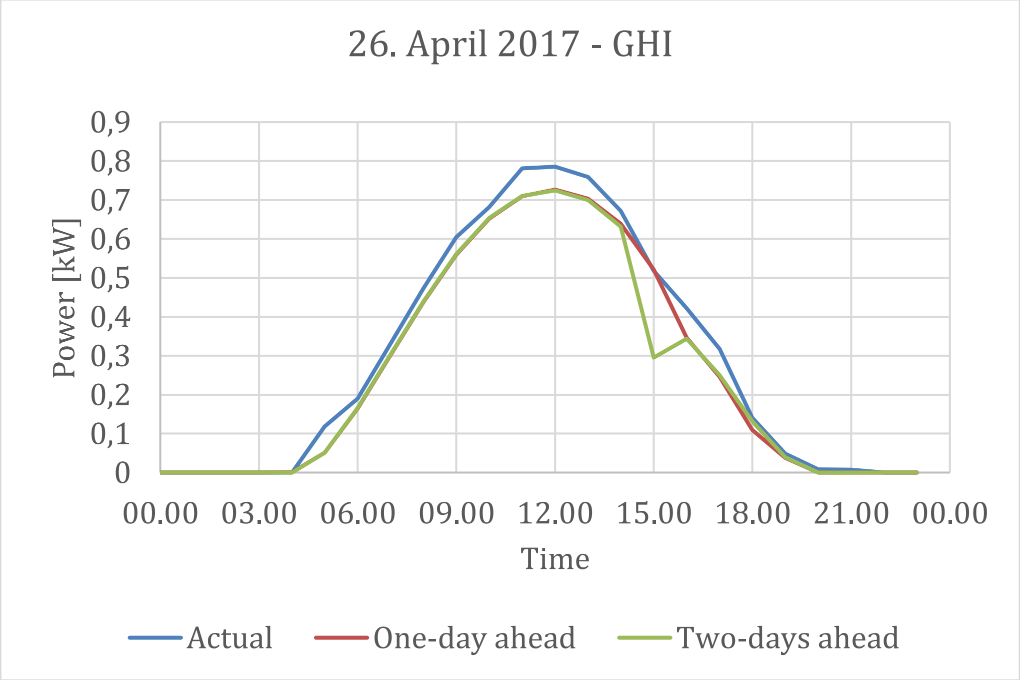
Headquarters:
Eltek Valere
Gråterudv. 8, PB 2340 Strømsø, 3003 Drammen, Norway
Phone: +47 32 20 32 00 Fax: +47 32 20 32 10

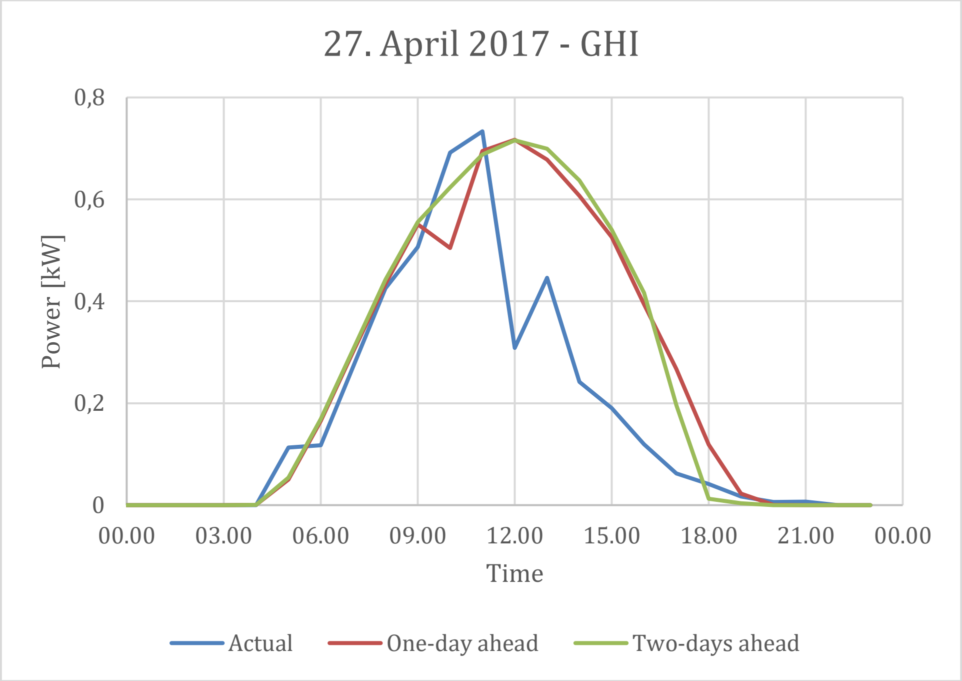
ELTEK VALERE
always on

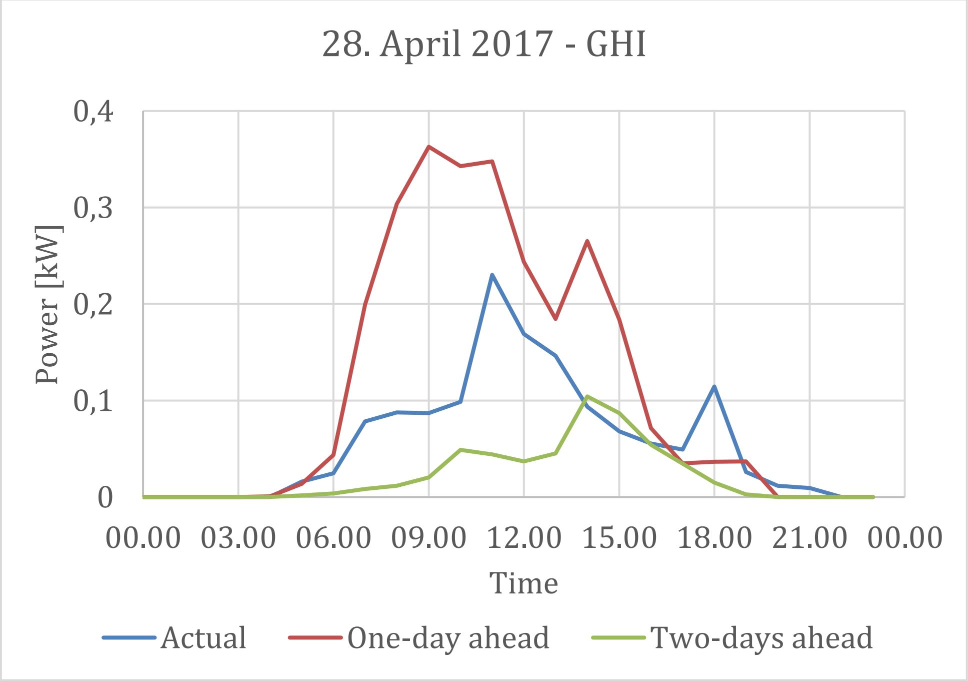
A.4 MEPS – Weather Forecast

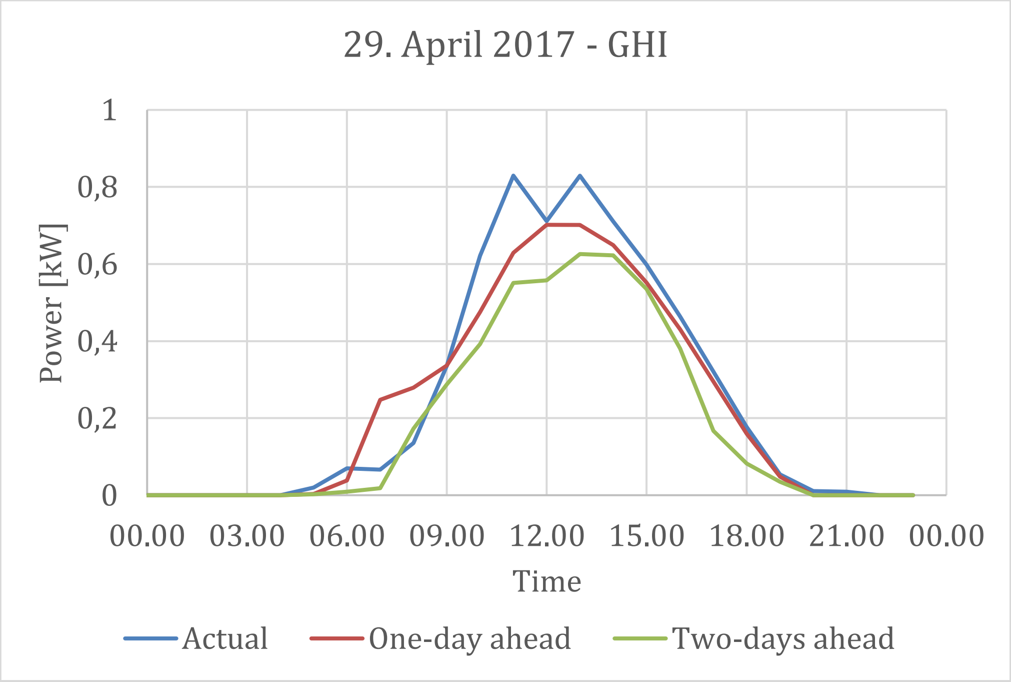
Global Horizontal Irradiance

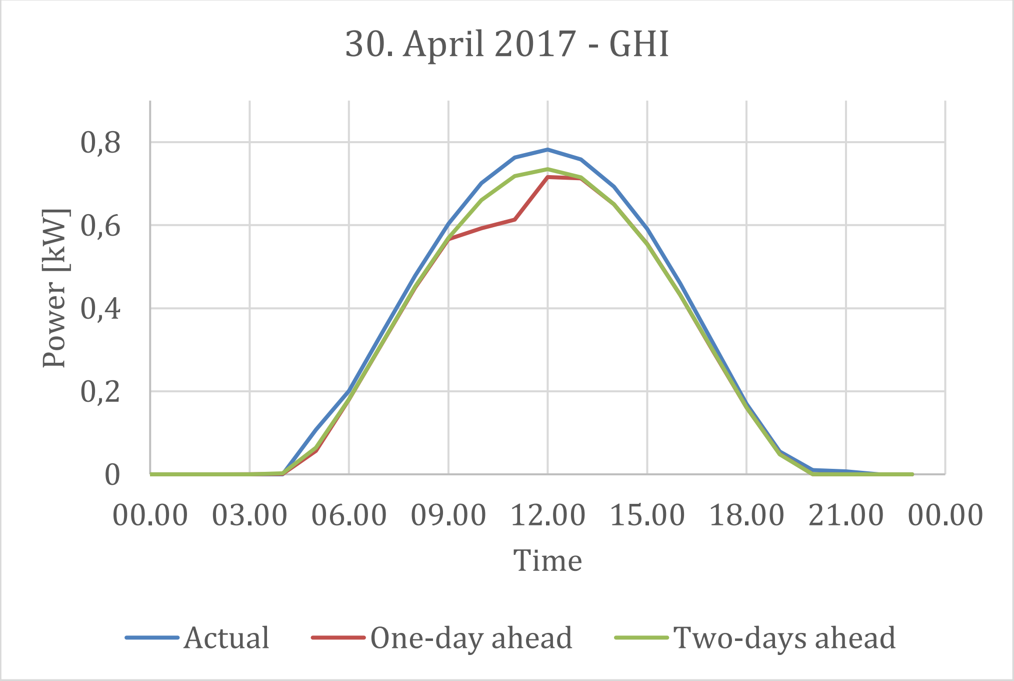




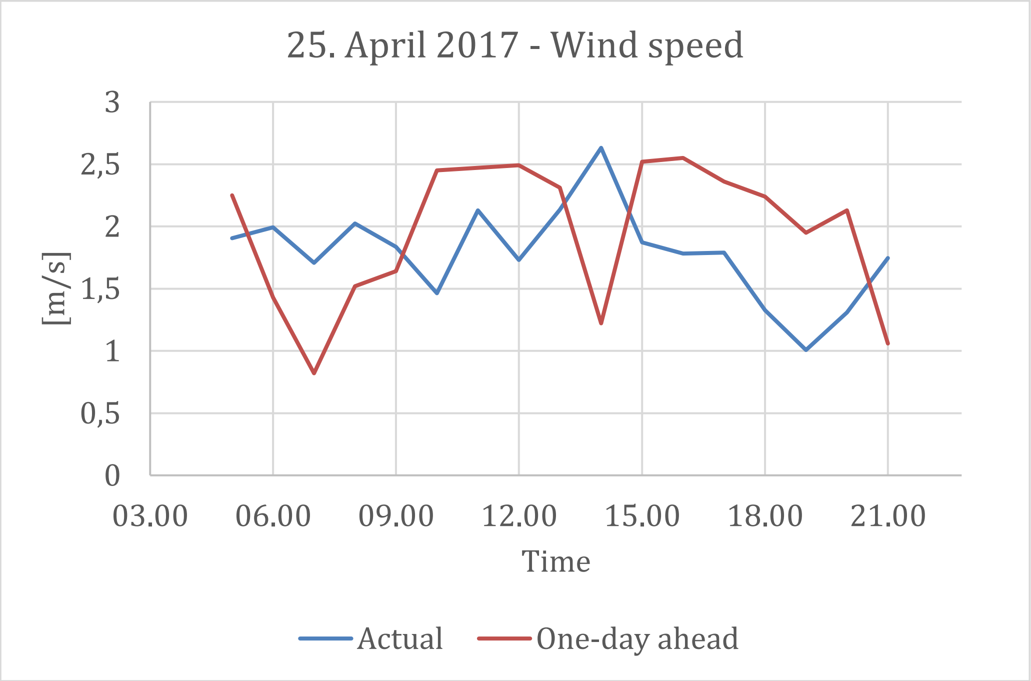

27. April 2017 - GHI
Power [kW]
0,8
0,6
0,4
0,2
0
00.00
03.00
06.00
09.00
12.00
15.00
18.00
21.00
00.00
Time
Actual
One-day ahead
Two-days ahead

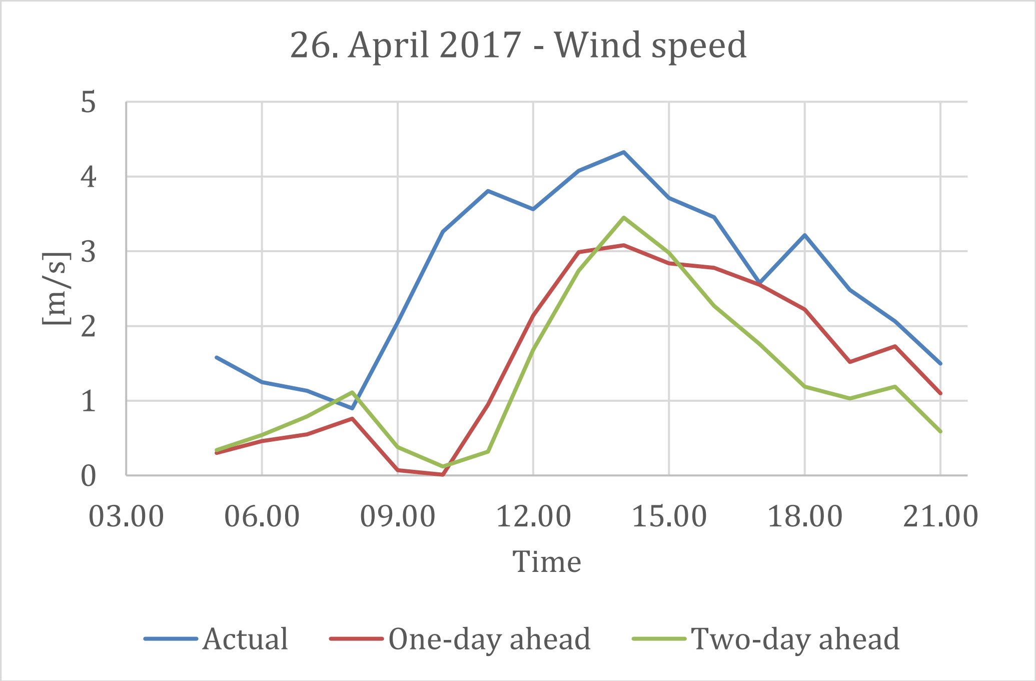

28. April 2017 - GHI
Power [kW]
0,4
0,3
0,2
0,1
0
00.00
03.00
06.00
09.00
12.00
15.00
18.00
21.00
00.00
Time
Actual
One-day ahead
Two-days ahead

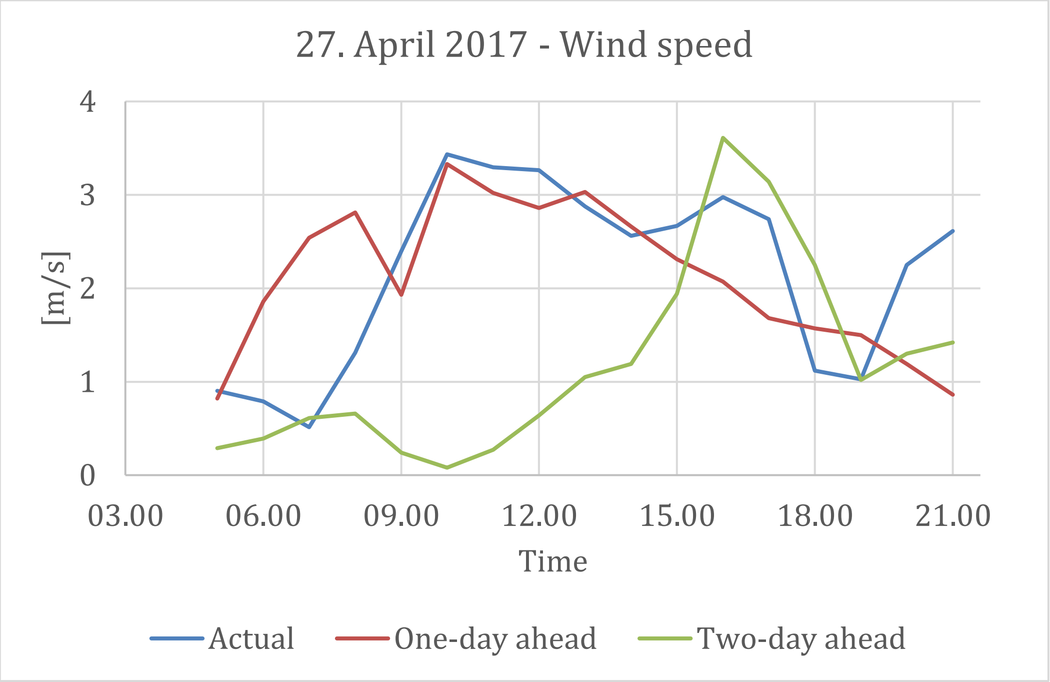

29. April 2017 - GHI
Power [kW]
1
0,8
0,6
0,4
0,2
0
00.00
03.00
06.00
09.00
12.00
15.00
18.00
21.00
00.00
Time
Actual
One-day ahead
Two-days ahead

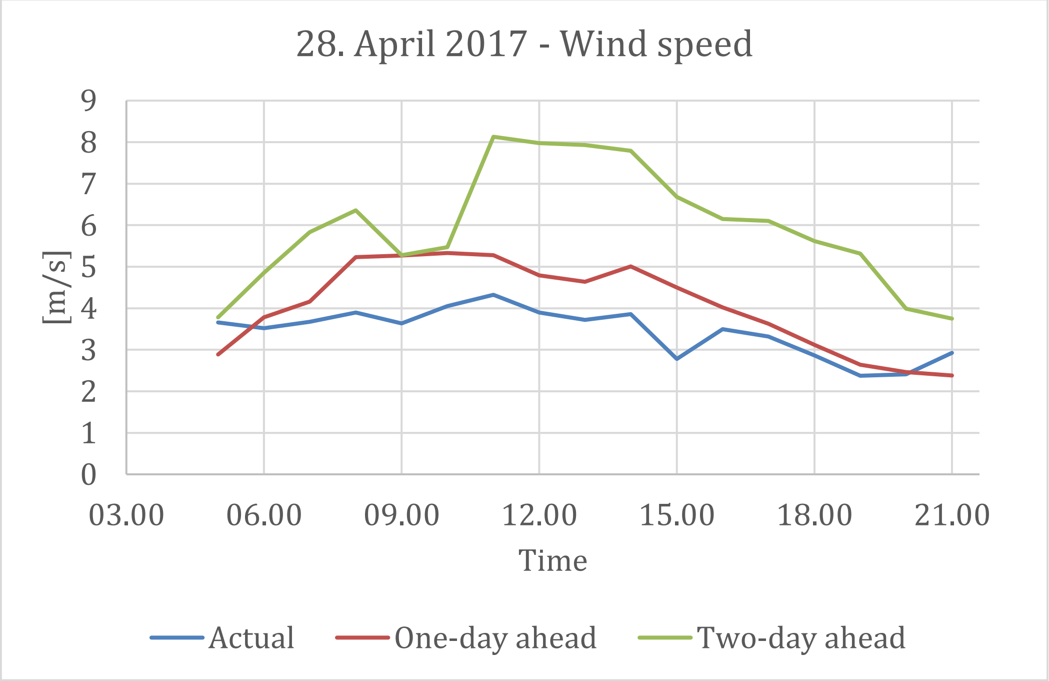

30. April 2017 - GHI
Power [kW]
0,8
0,6
0,4
0,2
0
00.00
03.00
06.00
09.00
12.00
15.00
18.00
21.00
00.00
Time
Actual
One-day ahead
Two-days ahead

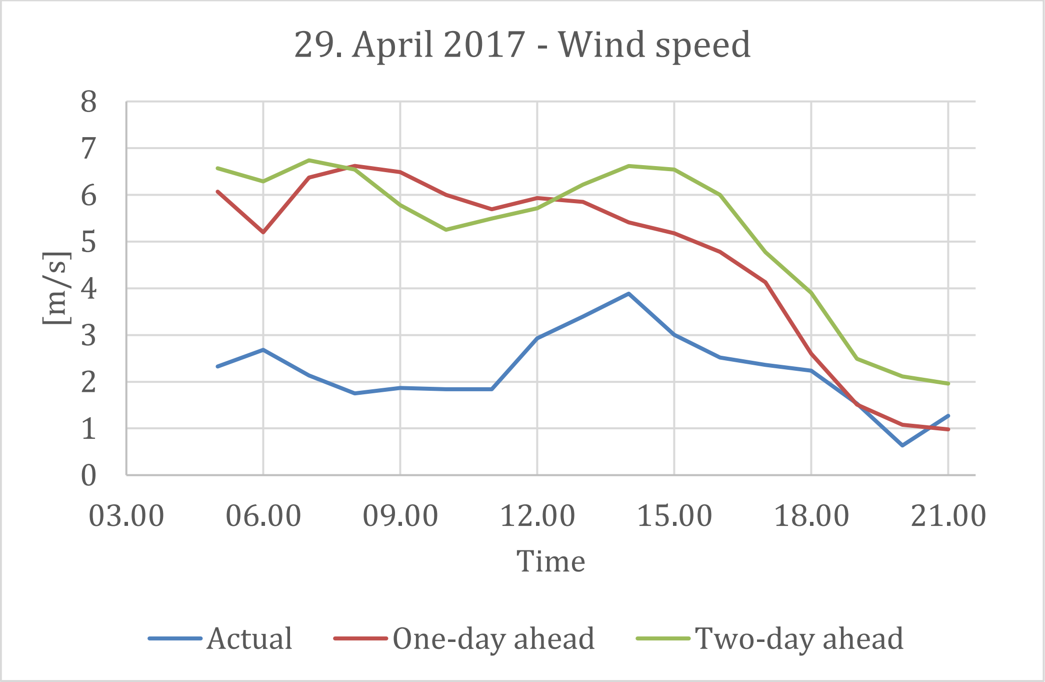
Wind speed:

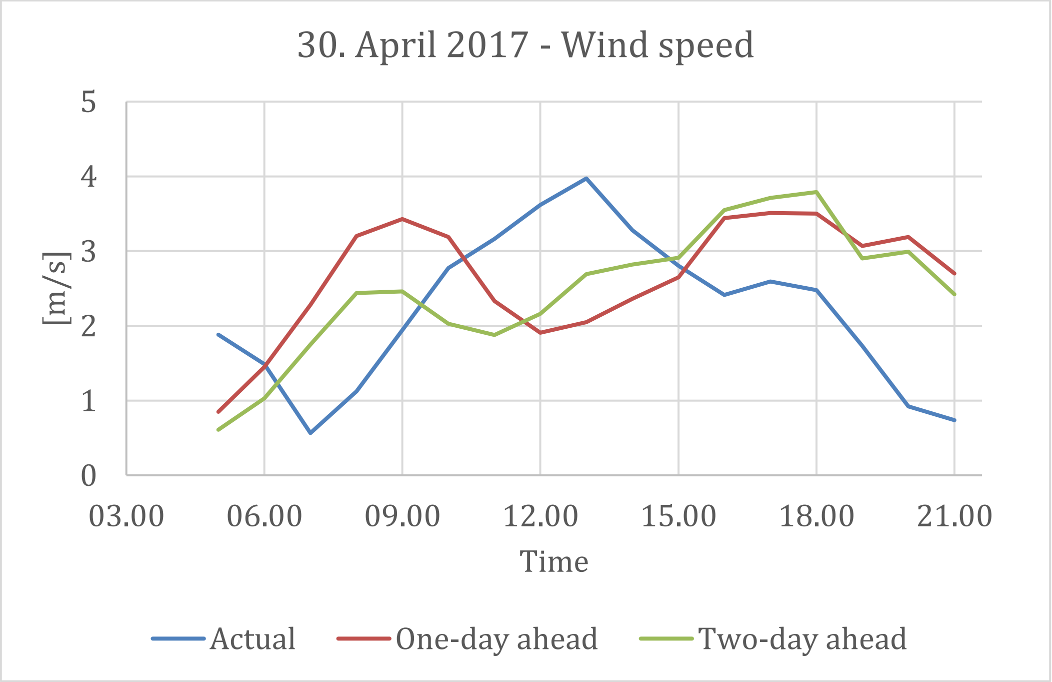

25. April 2017 - Wind speed
3
2,5
2
1,5
1
0,5
0
[m/s]
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead


26. April 2017 - Wind speed
5
4
3
2
1
0
[m/s]
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead
Two-day ahead

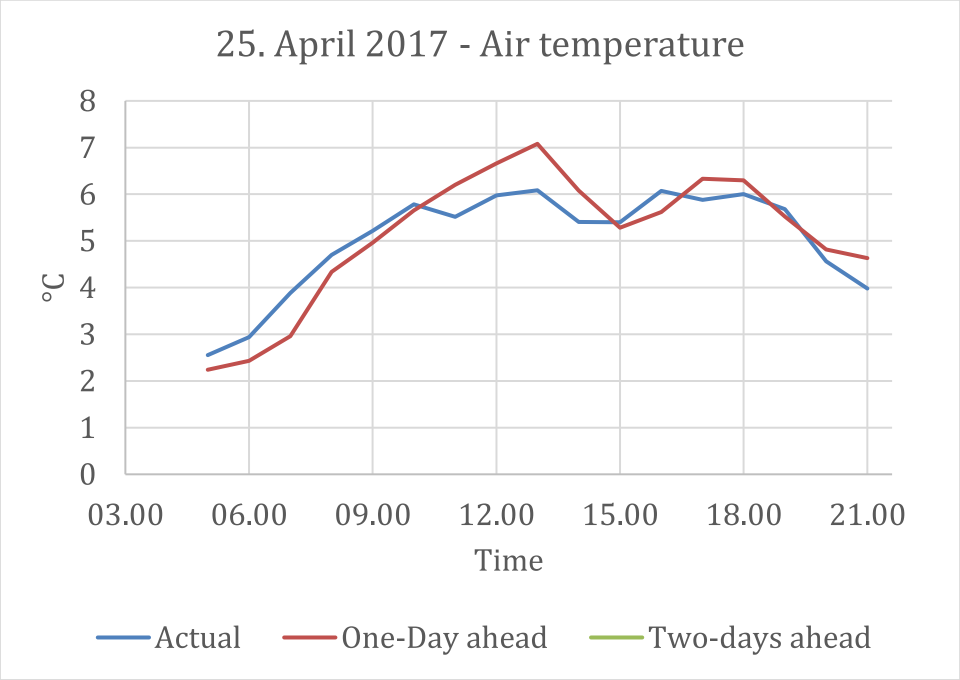

27. April 2017 - Wind speed
[m/s]
0
1
2
3
4
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead
Two-day ahead

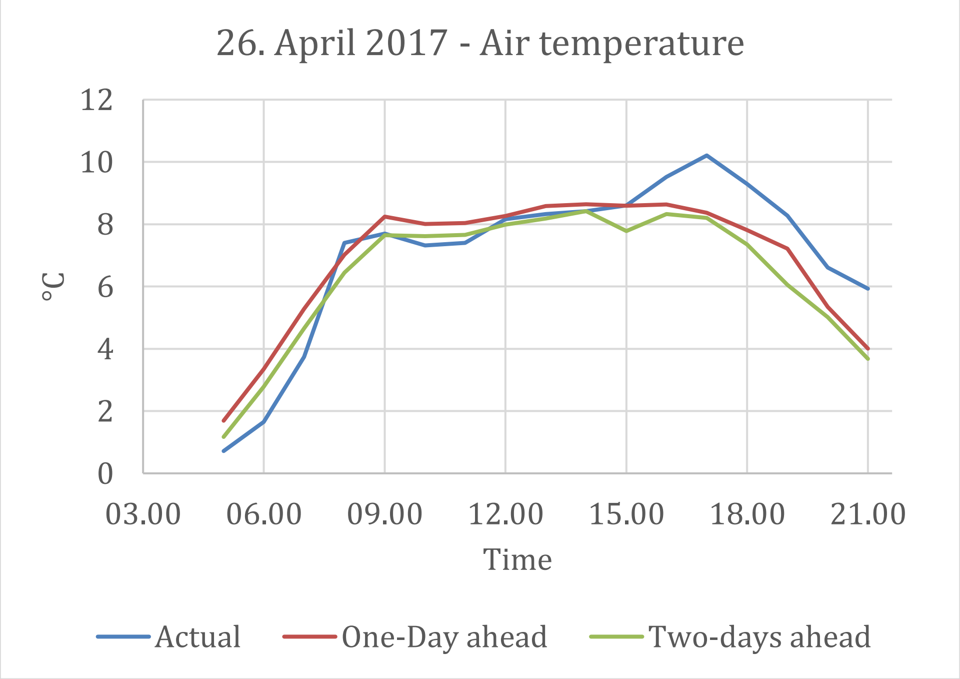

28. April 2017 - Wind speed
[m/s]
0
1
2
3
4
5
6
7
8
9
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead
Two-day ahead

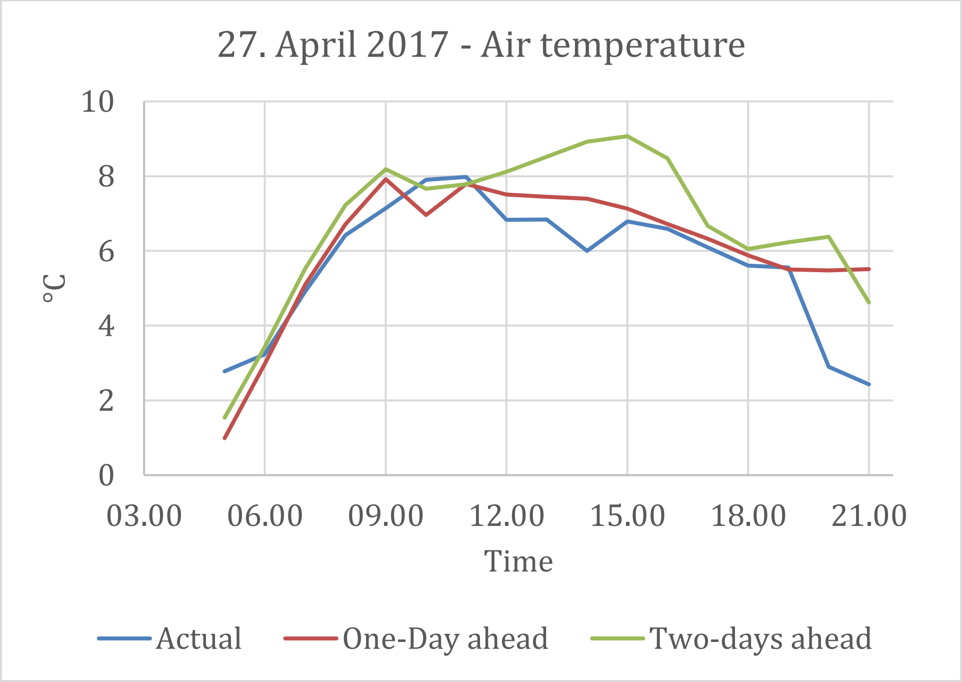

29. April 2017 - Wind speed
[m/s]
0
1
2
3
4
5
6
7
8
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead
Two-day ahead

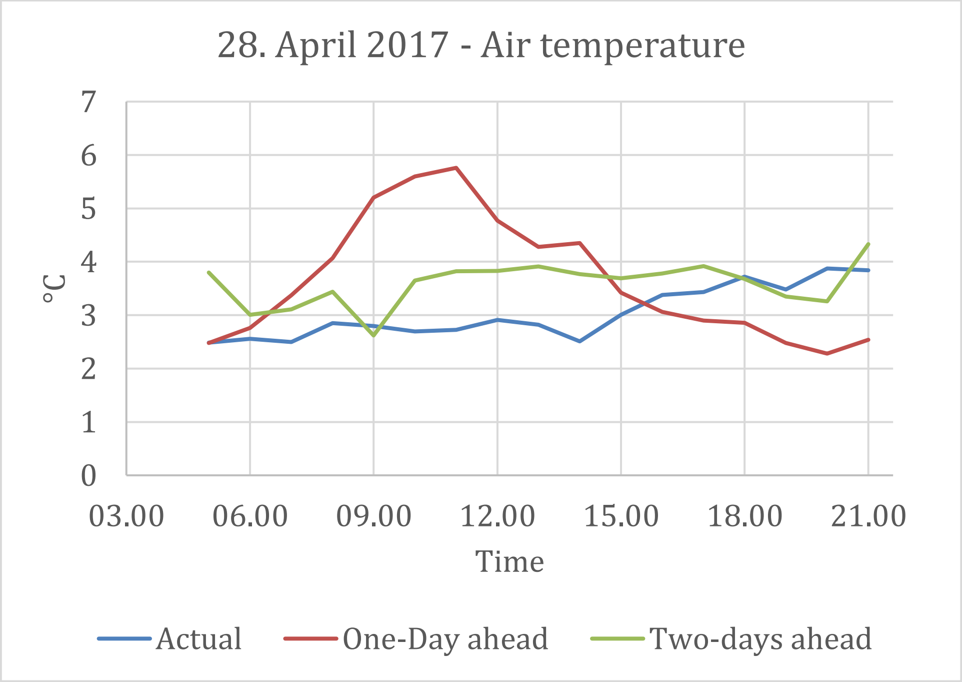

30. April 2017 - Wind speed
[m/s]
0
1
2
3
4
5
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-day ahead
Two-day ahead

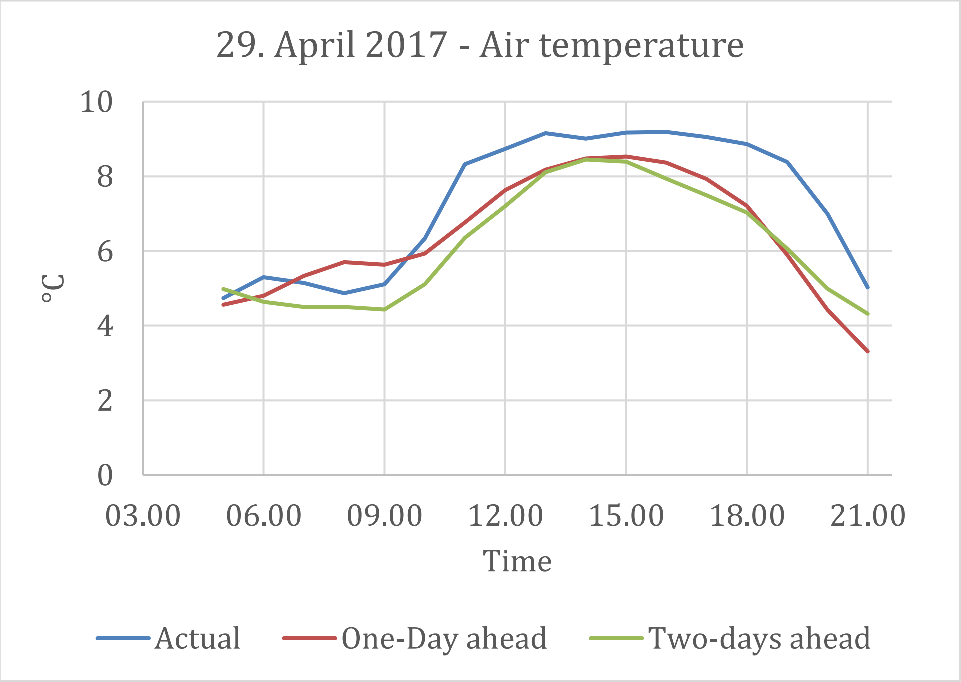
AIR TEMPERATURE

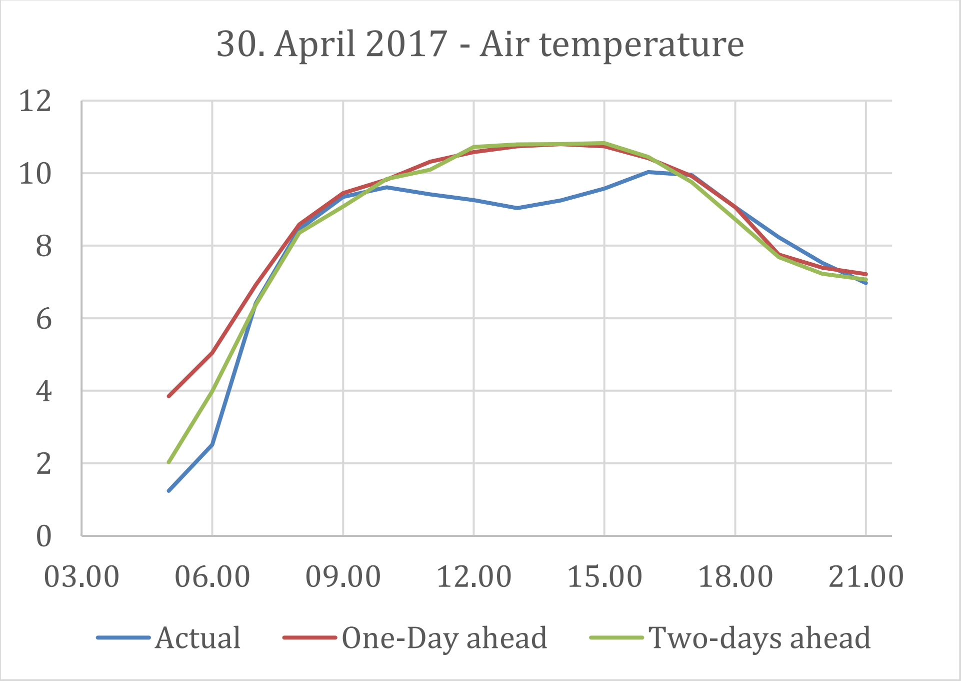





27. April 2017 - Air temperature
10
8
6
4
2
0
°C
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-Day ahead
Two-days ahead


28. April 2017 - Air temperature
7
6
5
4
3
2
1
0
°C
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-Day ahead
Two-days ahead


29. April 2017 - Air temperature
°C
10
8
6
4
2
0
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Time
Actual
One-Day ahead
Two-days ahead


30. April 2017 - Air temperature
12
10
8
6
4
2
0
03.00
06.00
09.00
12.00
15.00
18.00
21.00
Actual
One-Day ahead
Two-days ahead

A.5 SUNRISE/SUNSET – MATLAB SCRIPT

```
% Author: Camilla Lie
% Date: 24.05.2019
```

Load data

```
data = readtable('AllData.txt');              % Load all data
sun = readtable('SunData_fixed.txt');         % Load sunrise/sunset times
sun.SunRise =datetime(sun.SunRise);
sun.SunSet =datetime(sun.SunSet);
```

Removal of hours before sunrise and after sunset

```
for i = 1:height(sun)
    year = data.time.Year==sun.SunRise.Year(i);

    month = data.time.Month==sun.SunRise.Month(i);

    day = data.time.Day==sun.SunRise.Day(i);

    trueIdx = all([year,month,day],2);
    outsideScope = data.time(trueIdx)>=sun.SunRise(i) &...
                   data.time(trueIdx)<=sun.SunSet(i);

    trueIdx(trueIdx) = ~outsideScope;
    data(trueIdx,:) = [];
end
data.Properties.VariableNames = {'GHIavg' 'DHIavg' 'GTIavg' 'VSavg' 'VDavg' 'IDCinv1'
'IDCinv3' 'IDCinv4' 'UDCinv1' 'UDCinv3' 'UDCinv4' 'IACinv1' 'IACinv3' 'IACinv4'
'UACinv1' 'UACinv3' 'UACinv4' 'PTeoa1' 'PTmoa1' 'PTeoa3' 'PTeoa4' 'PTmoa4' 'PTmoa3'
'airtemp' 'time'};
```

A.6 Data Filtering - MATLAB Script

Inv1 = Suntech

Inv3 = Sharp

Inv4 = REC

```
% Data filtering using Kjøita PV system maintanance log

% Author: Camilla Lie
% Date: 24.05.2019
```

Import Data

```
data_time = readtable('alldata.csv');
```

Include Day of year

```
DV  = datevec(data_time.time);  % [N x 6] array
DV  = DV(:, 1:3);   % [N x 3] array, no time
DV2 = DV;
DV2(:, 2:3) = 0;    % [N x 3], day before 01.Jan
doy = cat(2, DV(:, [2 3]), datenum(DV) - datenum(DV2));
data_time.doy = doy(:,3);
tz = 1;
for i=1:height(data_time)
    tz(i)=1;
end
data_time.tz = tz';
```

Remove time to create array

```
dataa = data_time;

dataa.time = [];
```

IACinv4 values from start to 29.10.2012 kl. 14:30 changed to nan

```
dataa.IACinv4(1:144186)=nan;

dataa=table2array(dataa(:,1:end));

idxnan = isnan(dataa(:,14));
```

Replacing Power inv4 with NaN corresponding to previous section

```
for i=1:144186
    if idxnan(i,:)==1
        dataa(i,27)=nan;
    end
end
```

Timerange. Invalid GHI.

```
for i=1:length(dataa(:,1))
if dataa(i,1)==-999
    dataa(i,1)=nan;
end
end
```

Timerange. Poweroutage (Strømstans)

```
first = '25-May-2012 12:55:00'; % This will be the first taken
second = '29-May-2012 08:14:00'; % Time before is the last
LogicalArr = ( data_time.time >= first  & data_time.time< second );
```

Removing IAC/UAC all inverters between 25.may kl 12:55 to 29.May.2012 kl 08:13

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. DHI not valid before this

```
first = '22-May-2012'; % This will be the first taken
second = '23-Jun-2012'; % Time before is the last
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing DHI not valid between 22.May.2012 to 22.Jun.2012

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,2) = nan;
    end
end
```

Timerange. Downtime inv 1 and 3.

```
first = '04-Jul-2012 07:56:00';
second = '04-Jul-2012 09:27:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC inv1 and 3 from 04-Jul-2012 07:56 - 09:26

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:13,15:16]) = nan;
    end
end
```

Timerange. Update software all inv.

```
first = '29-Oct-2012 12:19:00';
second = '29-Oct-2012 12:31:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 29-Oct-2012 12:19 - kl. 12:31

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:17]) = nan;
    end
end
```

Timerange. Update software all inv.

```
first = '29-Oct-2012 12:53:00';
second = '29-Oct-2012 13:10:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 29-Oct-2012 12:53 - kl 13:09

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:17]) = nan;
    end
end
```

Timerange. Bad conntector inv1

```
first = '29-Oct-2012 07:34:00';
second = '06-Dec-2012 15:40:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC inv1 from 29-Oct-2012 07:34 - 06-Dec-2012 15:40

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12,15]) = nan;
    end
end
```

Timerange. Inverter calibration, pkt. 40

```
first = '11-Jan-2013'; % This will be the first taken
second = '19-Jan-2013'; % Time before is the last
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inverters between 11.jan.2013 to 18.jan.2013

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Missing data (=0) for all inv, pkt. 40

```
first = '15-May-2013 19:04:00'; % This will be the first taken
second = '16-May-2013 09:30:00'; % Time before is the last
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inverters between 15.May.2013 kl 19:05 and 16.May.2013 kl 09:30

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Cleaning pyranometers.

```
first = '31-May-2013 10:52:00';
second = '31-May-2013 11:24:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing GHI/DHI/GTI from 31-May-2013 10:52 - kl. 11:23

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,1:3) = nan;
    end
end
```

Timerange.downtime all inv

```
first = '31-May-2013 13:14:00';
second = '31-May-2013 13:22:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC all inv from 31-May-2013 13:14 - 13:22

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:17]) = nan;
    end
end
```

Timerange.downtime inv3

```
first = '06-Jul-2013 04:34:00';
second = '11-Jul-2013';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC inv3 from 06-Jul-2013 04:34 - 11-Jul-2013

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[13,16]) = nan;
    end
end
```

Timerange.downtime inv4

```
first = '20-Jun-2013 13:27:00';
second = '20-Jun-2013 13:21:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC inv4 from 20-Jun-2013 13:27 - 13:21

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[14,17]) = nan;
    end
end
```

Timerange. Cleaning pyranometers.

```
first = '12-Feb-2014 11:56:00';
second = '12-Feb-2014 12:23:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing GHI/GTI/DHI from 12.Feb 2014 kl. 11:56 - 12:22

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,1:3) = nan;
    end
end
```

Timerange.downtime inv1

```
first = '17-Feb-2014 12:21:00';
second = '17-Feb-2014 12:30:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC inv1 from 17-Feb-2014 12:21 - 12:30

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12,15]) = nan;
    end
end
```

Timerange.downtime inv3

```
first = '17-Feb-2014 11:55:00';
second = '17-Feb-2014 12:11:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC inv3 from 17-Feb-2014 11:55 - 12:11

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[13,16]) = nan;
    end
end
```

Timerange.downtime inv4

```
first = '17-Feb-2014 12:06:00';
second = '17-Feb-2014 15:02:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC inv4 from '17-Feb-2014 12:06 - 15:02

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[14,17]) = nan;
    end
end
```

Timerange. Inv1 stopped, lost data

```
first = '07-Apr-2014 13:40:00';
second = '08-Apr-2014 12:55:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IACinv1 from 07.April 2014 kl. 13:40 - 08.April 2014 kl.12:55

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12 15]) = nan; %IACinv1 and UACinv1
    end
end
```

Timerange. Changed inv1

```
first = '09-Apr-2014 12:42:00';
second = '10-Apr-2014 11:37:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UACinv1 from 09.April 2014 kl. 12:42 - 10.April 2014 kl.11:37

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12 15]) = nan;
    end
end
```

Timerange. Removed inv4 for a while. IS THIS WHEN INV4 IS CHANGED?

```
first = '14-Apr-2014 12:03:00';
second = '15-Apr-2014 08:55:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UACinv4 from 14.April 2014 kl. 12:03 - 15.April 2014 kl.08:54

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[14 17]) = nan;
    end
end
```

Timerange. Downtime, Removed all inv. Lightening?

```
first = '02-Aug-2014 10:53:00';
second = '06-Aug-2014 08:20:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 02.August 2014 kl. 10:53 -

```
     06.August 2014 kl.08:19. Lightning?
```

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Downtime for all inv.

```
first = '26-Aug-2014 16:46:00';
second = '29-Aug-2014';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 26.August 2014 kl.16:46-28.Aug(including)

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange.downtime all inv

```
first = '22-Oct-2014 08:19:00';
second = '24-Oct-2014';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC all inv from 22-23 Oct-2014

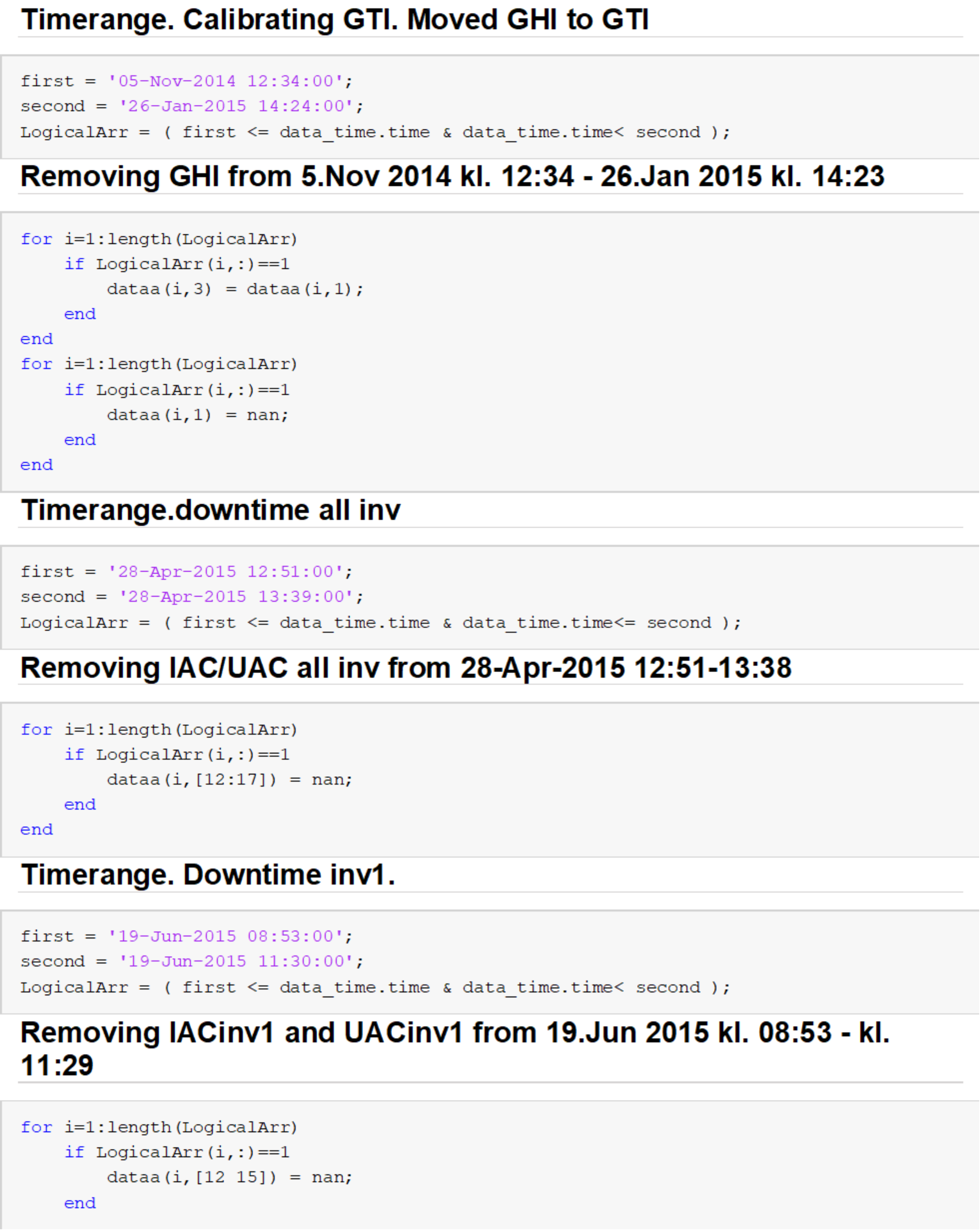
```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:17]) = nan;
    end
end
```

Timerange. Calibrating DHI.

```
first = '05-Nov-2014 12:11:00';
second = '16-Mar-2015 13:40:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing DHI from 5.Nov 2014 kl. 12:11 - 16.Mar 2015 kl. 13:40.

```
    if LogicalArr(i,:)==1
        dataa(i,2) = nan;
    end
end
```



Timerange. Calibrating GTI. Moved GHI to GTI

```
first = '05-Nov-2014 12:34:00';
second = '26-Jan-2015 14:24:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing GHI from 5.Nov 2014 kl. 12:34 - 26.Jan 2015 kl. 14:23

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,3) = dataa(i,1);
    end
end
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,1) = nan;
    end
end
```

Timerange.downtime all inv

```
first = '28-Apr-2015 12:51:00';
second = '28-Apr-2015 13:39:00';
LogicalArr = ( first <= data_time.time & data_time.time<= second );
```

Removing IAC/UAC all inv from 28-Apr-2015 12:51-13:38

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12:17]) = nan;
    end
end
```

Timerange. Downtime inv1.

```
first = '19-Jun-2015 08:53:00';
second = '19-Jun-2015 11:30:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IACinv1 and UACinv1 from 19.Jun 2015 kl. 08:53 - kl. 11:29

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,[12 15]) = nan;
    end
```

```
end
```

Timerange. Downtime inv1.

```
first = '27-Aug-2015 19:00:00';
second = '03-Sep-2015 06:00:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC and UAC all inv from 27.Aug 2015 kl. 19:00 - 3.Sep 2015

```
%kl. 06:00
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Downtime all inv.

```
first = '09-Jun-2016 19:30:00';
second = '20-Jun-2016 12:11:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 09.Jun 2016 kl. 19:30 - 20.Jun2016kl.12:10

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Downtime all inv.

```
first = '17-Nov-2016 11:30:00';
second = '21-Nov-2016 12:50:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 17. Nov 2016 kl. 11:3 - 21.Nov2016kl.12:50

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
```

```
end
```

Timerange. Downtime all inv.

```
first = '23-May-2017 21:04:00';
second = '24-May-2017 09:15:00';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 23.May 2017 kl. 21.04 - 23.May2017kl.09:14

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
```

Timerange. Downtime all inv.

```
first = '01-Sep-2017';
second = '02-Sep-2017';
LogicalArr = ( first <= data_time.time & data_time.time< second );
```

Removing IAC/UAC all inv from 01.Sep 2017 - 02. Sep 2017

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,12:17) = nan;
    end
end
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,1:3) = nan;
    end
end
```

TimeZone

```
r1 = '22-May-2012';
s1 = '27-Oct-2012';
r2 = '31-Mar-2013';
s2 = '26-Oct-2013';
r3 = '30-Mar-2014';
s3 = '25-Oct-2014';
r4 = '29-Mar-2015';
s4 = '24-Oct-2015';
r5 = '27-Mar-2016';
s5 = '29-Oct-2016';
```

```
r6 = '26-Mar-2017';
s6 = '28-Oct-2017';
LogicalArr = ( r1 <= data_time.time & data_time.time<= s1 | r2 <=...
    data_time.time & data_time.time<= s2 | r3 <= data_time.time & ...
    data_time.time<= s3 | r4 <= data_time.time & data_time.time<= s4 |...
    r5 <= data_time.time & data_time.time<= s5 | r6 <= data_time.time & ...
    data_time.time<= s6);
```

Correcting TimeZone

```
for i=1:length(LogicalArr)
    if LogicalArr(i,:)==1
        dataa(i,32) = 2;
    end
end
```

A.7 AIR MASS – MATLAB SCRIPT

```
% Author: Camilla Lie
% Date: 24.05.2019
```

Load data

```
data = readtable('clean_hour.txt');      % Hourly data May 2012-Feb. 2018
alltime = data.time;                     % Time
doy = data.doy;                          % Day of year
```

Time

```
date_time = datenum(alltime);

date = floor(date_time);

time = date_time-date;
```

Constants

```
latitude = 58.1535345;

longitude = 8.0020142;

delta = -23.45 * cosd(360.*(doy+10)./365);    % Solar declination angle
altitude = 20;
```

Time Zone

```
r1 = '22-May-2012';      % UTC+2
s1 = '27-Oct-2012';      % UTC+1
r2 = '31-Mar-2013';      % UTC+2
s2 = '26-Oct-2013';      % UTC+1
r3 = '30-Mar-2014';      % UTC+2
s3 = '25-Oct-2014';      % UTC+1
r4 = '29-Mar-2015';      % UTC+2
s4 = '24-Oct-2015';      % UTC+1
r5 = '27-Mar-2016';      % UTC+2
s5 = '29-Oct-2016';      % UTC+1
r6 = '26-Mar-2017';      % UTC+2
s6 = '28-Oct-2017';

% TimeZone
LogicalArr = ( r1 <= alltime & alltime<= s1 | r2 <= alltime & alltime<=...
    s2 | r3 <= alltime & alltime<= s3 | r4 <= alltime & alltime<= s4 | r5...
    <= alltime & alltime<= s5 | r6 <= alltime & alltime<= s6);

dst = LogicalArr';
```

```
dst_tz = double(dst);       % Time zone
```

Get correct timeZone

```
for i=1:length(dst_tz)
    if dst_tz(i) == 1
        tz(i) = 2;
    elseif dst_tz(i) == 0
        tz(i) = 1;
    end
end

tz =tz';                % Time zone
time = time';
delta = delta';         % Solar declination angle
B = 360*(doy-81)/365;

% ET = difference between Appearent solar time and Mean solar time
ET = (9.87*sin(2*B)-7.53*cosd(B)-1.5*sind(B))';      % Equation of Time
solarTime = ((time.*24-double(dst)).*60+4.*(longitude-1.*15)+ET)./60./24;
t_h = (solarTime*24-12)*15;             % Hours angle
```

Theta calculations (zenith angle)

```
theta = acosd(sind(latitude)*sind(delta)+...    % Zenith angle
    cosd(latitude)*cosd(delta).*cosd(t_h));

theta1 = theta;         % For scaled AM
theta2 = theta;         % For theta <= 89 degrees

% Scaling
for i=1:length(theta1)
    if theta1(i)>=(90)
        theta1(i)= theta1(i)- theta1(i)*0.03;
    end
end

for i=1:length(theta1)
    if theta1(i)>=(90)
        theta(i)= theta1(i)- theta1(i)*0.03;
    end
end

for i=1:length(theta1)
    if theta1(i)>=(90)
        theta1(i)= 89.8;
    end
end
```

```
% Maximum theta <=89
for i=1:length(theta2)
    if theta2(i)>=89
        theta2(i)= 89;
    end
end
```

Air mass calculations

```
AM1= ((exp(-0.0001184*altitude))./(cosd(theta1)+0.5057.*...
    (96.080+theta1).^(-1.634)))';                            % Scaled AM
AM2= ((exp(-0.0001184*altitude))./(cosd(theta2)+0.5057.*...
    (96.080+theta2).^(-1.634)))';                            % theta <=89

figure
subplot(2,1,1)
plot(alltime(14134:40436),AM1(14134:40436),'LineWidth',1)
grid
title('\fontsize{13}(a)')
ylabel('Air mass [-]')
set(gca,'FontSize',18)

subplot(2,1,2)
plot(alltime(14134:40436),AM2(14134:40436),'LineWidth',1)
grid
title('\fontsize{13}(b)')
% ylabel('Air mass [-]')
set(gca,'FontSize',18)
```

A.8 PRINCIPAL COMPONENT ANALYSIS (PCA) – MATLAB SCRIPT

```
% Author: Camilla Lie
% Date: 24.05.2019
```

Import data

```
data_inv1 = readtable('inv1.txt');
```

Variables

```
data1 = [data_inv1.GHIavg,data_inv1.VS,data_inv1.airtemp];
```

Standardize

```
norm_all = normalize(data1);       % normalize in MATLAB = standardize (mean=0, STD=1)
```

Principal Component Analysis (PCA)

MATLAB function

```
[coeff,score,latent,tsquared,explained] = pca(norm_all,'Rows','complete'); % If there
is missing data

figure;
pareto(explained)
xlabel('Principal Component')

figure;
[handlesPareto, axesPareto] = pareto(explained);
yticks = get(axesPareto(2),'YTick');
RightYLabels = cellstr(get(axesPareto(2),'YTickLabel'));
xl = xlim;
set(axesPareto(2),'YTickLabel',[])

xlabel('Principal Component','fontsize',20)
ylabel('Variance Explained (%)','fontsize',20)
set(gca,'FontSize',20)
```

A.9 AVERAGE MUTUAL INFORMATION (AMI) – MATLAB SCRIPT

```
% Reference:
 % R. D. Thomas, N. C. Moses, E. A. Semple, and A. J. Strang,
 % "An efficient algorithm for the computation of average mutual
 % information: Validation and implementation in Matlab," Journal of
 % Mathematical Psychology, vol. 61, pp. 45-59, 2014/08/01/ 2014.

function AMI = average_mutual_information(data)
% function AMI = average_mutual_information(data)
% Calculates average mutual information between
% two
% columns of data. It uses kernel density
% estimation,
% with a globally adjusted Gaussian kernel.
%
% Input should be an n-by-2 matrix, with data sets
% in adjacent
% column vectors.
%
% Output is a scalar.
n = length(data);
X = data(:,1);
Y = data(:,2);
% Example below is for normal reference rule in
% 2 dims, Scott (1992).
hx = std(X)/(n^(1/6));
hy = std(Y)/(n^(1/6));
% Compute univariate marginal density functions.
P_x = univariate_kernel_density(X, X, hx);
P_y = univariate_kernel_density(Y, Y, hy);
% Compute joint probability density.
JointP_xy = bivariate_kernel_density(data, data, hx, hy);
AMI = sum(log2(JointP_xy./(P_x.*P_y))/n);

end
function y = bivariate_kernel_density(value, data, Hone, Htwo)
% function y = bivariate_kernel_density(value,
% data, Hone, Htwo)
% Calculates bivariate kernel density estimates
% of probability.
% Inputs are: value (m x 2 matrix), where density
% is estimated;
% data (n x 2 matrix), the data used to
% estimate the density;
% Hone (scalar) and Htwo (scalar) to use
% for the widths of density estimation.
% Output is an m-vector of probabilities estimated
% at the values in ’value’.
s = size(data);
n = s(1);
t = size(value);
```

```
number_pts = t(1);
rho_matrix = corr(data);
rho = rho_matrix(1,2);
% The adjusted covariance matrix:
W = [Hone^2 rho*Hone*Htwo; rho*Hone*Htwo Htwo^2];
Differences = linear_depth(value,-data);
prob = mvnpdf(Differences,[0 0],W);
Cumprob = cumsum(prob);
y(1) = (1/n)*Cumprob(n);
    for i = 2:number_pts
        index = n*i;
        y(i) = (1/(n))* (Cumprob(index)-Cumprob(index - n));
        i = i + 1;
    end
y = y';
end
function y = Extended(vector,n)
% Takes an m-dimensional row vector and outputs an
% n-by-m matrix with n-many consecutive repeats of
% the vector. Similarly, it takes an
% m-dimensional column vector and outputs an
% m-by-n matrix.
% Else, it returns the original input.
M = vector;
    if size(vector,1) == 1
        M = zeros(n,length(vector));
        for i = 1:n
            M(i,:) = vector;
            i = i + 1;
        end
    end
    if size(vector,2) == 1
        M = zeros(length(vector),n);
        for i = 1:n
            M(:,i) = vector;
            i = i + 1;
        end
    end
y = M;
end
function y = linear_depth(feet, toes)
% linear_depth takes a matrix ‘feet’ and lengthens
% it in blocks, takes a matrix ‘toes’ and lengthens
% it in Extended repeats, and then adds the% lengthened ‘feet’ and ‘toes’ matrices to
achieve
% all sum combinations of their rows.
% feet and toes have the same number of columns
    if size(feet, 2) == size(toes, 2)
        a = size(feet, 1);
        b = size(toes, 1);
        Blocks = zeros(a*b, size(toes, 2));
        Bricks = Blocks;
        for i = 1:a
            Blocks((i-1)*b + 1: i*b,:) = Extended(feet(i,:),b);
            Bricks((i-1)*b + 1: i*b,:) = toes;
            i = i + 1;
        end
    end
y = Blocks + Bricks;


end
function y = univariate_kernel_density(value,data, window)
% function y = univariate_kernel_density(value,
% data, window)
% Estimates univariate density using kernel
% density estimation.
% Inputs are: value (m-vector), where density is
% estimated;
% data (n-vector), the data used to
% estimate the density;
% window (scalar), used for the width of
% density estimation.
% Output is an m-vector of probabilities.
h = window;
n = length(data);
m = length(value);
% We use matrix operations to speed up computation
% of a double-sum.
Prob = zeros(n, m);
G = Extended(value, n);
H = Extended(data', m);
Prob = normpdf((G - H)/h);
fhat = sum(Prob)/(n*h);
y = fhat';
% end
```

Code Below:

```
% Author: Camilla Lie
% Date: 24.05.2019
```

Import data

```
data1 = readtable('inv1.txt');
```

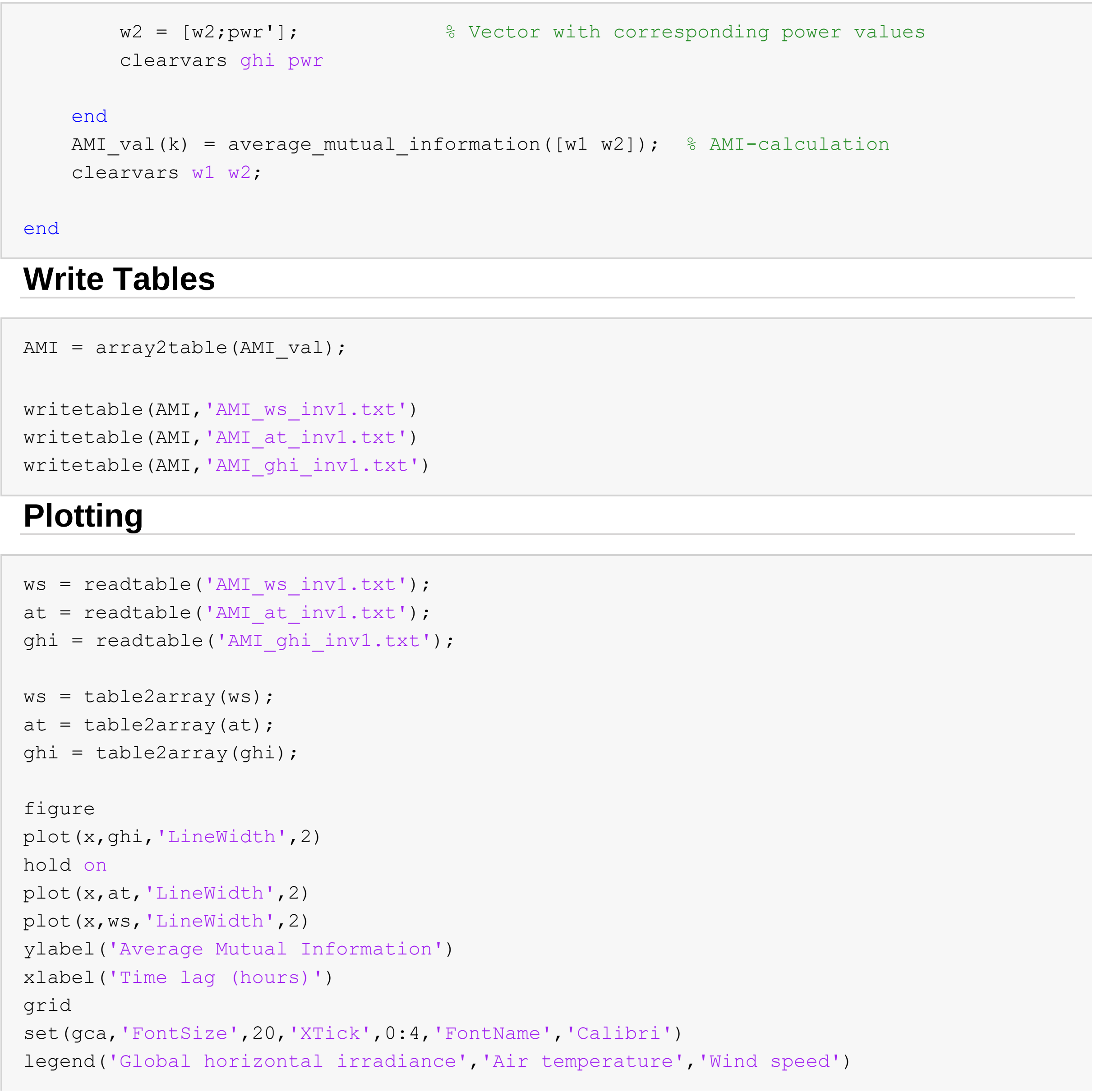
Adapting data matrix

This is performed to get a matrix [hours x days] for all data. Matrix will be as long as the longest day of the year (number of hours) but days with fewer hours will then have "NaN" for corresponding hour. This is performed to be able to calculate AMI over a continuous dataset as time-lag is involved (e.g. a time lag of 2 hours will compare hour x to hour x+2, but if there is no more hours for relevant day, this cannot be calculated. Thus, the model will go to the next day instead to calculate AMI with corresponding time lag.

```
timee = data1.time;                         % Time

% Variables
jourreference = [1 1];                      % Month/day
reference = datetime([timee(1).Year jourreference]);
njour = fix(days(timee - reference) + 1);
[~, ~, jourindex] = unique(njour);
colonnesjour = accumarray(jourindex, data1.VS, [], @(c) {c(:)});    % Variable = GHI,
WS or AT
maxhauteur = max(cellfun(@numel, colonnesjour));
colonnesjour = cellfun(@(c) [c; nan(maxhauteur - numel(c), 1)], colonnesjour,
'UniformOutput', false);
colonnesjour = [colonnesjour{:}];

% Power
colonnesjour1 = accumarray(jourindex, data1.PACinv1, [], @(c) {c(:)});  % Power
variable
maxhauteur1 = max(cellfun(@numel, colonnesjour1));
colonnesjour1 = cellfun(@(c) [c; nan(maxhauteur1 - numel(c), 1)], colonnesjour1,
'UniformOutput', false);
colonnesjour1 = [colonnesjour1{:}];

var = colonnesjour;         % Vector with variable (GHI, WS or AT)
power = colonnesjour1;      % Vector with Power(AC)
```

AMI Time-lag Calculations

```
for k = 1:4                                 % Lags (hour)
    w1=[];                                  % Vector with Variable
    w2=[];                                  % Vector with power (AC)
    for i = 400:900                         % Days
        j = 1;                              % Hours
        while (j+k-1<=19)&&(~isnan(var(j+k-1,i)))    % Avoid discontinuity in AMI
calculations
            variable(j) = var(j,i);                 % Vector with continuous varibale
values
            pwr(j) = power(j+(k-1),i);              % Vector with corresponding power
values

            j = j+1;
        end
        w1 = [w1;variable'];       % Vector with continuous varibale values
```



```
        w2 = [w2;pwr'];            % Vector with corresponding power values
        clearvars ghi pwr

    end
    AMI_val(k) = average_mutual_information([w1 w2]);  % AMI-calculation
    clearvars w1 w2;

end
```

Write Tables

```
AMI = array2table(AMI_val);

writetable(AMI,'AMI_ws_inv1.txt')
writetable(AMI,'AMI_at_inv1.txt')
writetable(AMI,'AMI_ghi_inv1.txt')
```

Plotting

```
ws = readtable('AMI_ws_inv1.txt');
at = readtable('AMI_at_inv1.txt');
ghi = readtable('AMI_ghi_inv1.txt');

ws = table2array(ws);
at = table2array(at);
ghi = table2array(ghi);

figure
plot(x,ghi,'LineWidth',2)
hold on
plot(x,at,'LineWidth',2)
plot(x,ws,'LineWidth',2)
ylabel('Average Mutual Information')
xlabel('Time lag (hours)')
grid
set(gca,'FontSize',20,'XTick',0:4,'FontName','Calibri')
legend('Global horizontal irradiance','Air temperature','Wind speed')
```

A.10 DATA DIVISION – MATLAB SCRIPT

```
% Author: Camilla Lie
% Date: 24.05.2019

% This is an example from Suntech PV modules (inv1), but was also done with
% Sharp and REC. Each of the PV modules had their sub-sets divided
% separately, meaning the three sets (training, validation and test) are
% not divided equally between the three types of modules. They are divided
% randomly to obtain the lowest percentage difference for the coefficent of
% variation within each of the variables.
```

Load data

```
data1 = readtable('inv1_am89.txt');      % Using Air Mass theta <= 89 degrees

% Remove variables that will not be used
data1 = removevars(data1,{'DHIavg','GTIavg','doy','tz'});
```

Choose time-interval

```
    %INV1
time_find1= datevec(data1.time);
year1 = time_find1(:,1:3);
year1 = datetime(year1);
year1 = datetime(year1,'InputFormat','dd/MMM/yyyy','Format','MMM-yyyy');
start1 = find(year1=='01-Jan-2014',1,'first');
stop1 = find(year1==' 31-Dec-2016',1,'last');
```

Divide into datasets

```
data_inv1 = data1(start1:stop1,:);




% Removal of NaNs
data_inv1 = rmmissing(data_inv1);
```

Normalize data

```
inv1_norm = normalize(data_inv1(:,2:end));
```

Choose set for INV1 (SUNTECH)

```
    %$$$$$$$$$$$$$$ SHOULD NOT BE CHANGED AFTER DIVISION $$$$$$$$$$$$$$$
% p = 0.70;
%
%     %INV1
% [m1,n1] = size(inv3_norm);
% trainInd1 = (idx1((1:round(p*m1))))';
% val = idx1((round(p*m1)+1:end));
% valInd1 = (val(1:round(length(val)/2)))';
% testInd1 = (val(round(length(val)/2)+1:end))';

% x1 = data_inv3(trainInd1,:);
% y1 = data_inv3(valInd1,:);
% z1 = data_inv3(testInd1,:);

    %$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

    % TRAINING
% Mean
avgen = nanmean(x1.GHIavg);
avgto = nanmean(x1.VS);
avgtre = nanmean(x1.airtemp);
avgfem = nanmean(x1.AM2);
avgseks = nanmean(x1.PACinv1);

% Standard Deviation
stden = nanstd(x1.GHIavg);
stdto = nanstd(x1.VS);
stdtre = nanstd(x1.airtemp);
stdfem = nanstd(x1.AM2);
stdseks = nanstd(x1.PACinv1);

% Coefficient of Variation
cven = avgen./stden;
cvto = avgto./stdto;
cvtre = avgtre./stdtre;
cvfem = avgfem./stdfem;
cvseks = avgseks./stdseks;

% Descriptve statistics training set SUNTECH
train_inv1_avg = [avgen stden cven; avgto stdto cvto; avgtre stdtre...
    cvtre; avgfem stdfem cvfem; avgseks stdseks cvseks]

    % VALIDATION
% Mean
avgenval = nanmean(y1.GHIavg);
avgtoval = nanmean(y1.VS);
avgtreval = nanmean(y1.airtemp);
avgfemval = nanmean(y1.AM2);
avgseksval = nanmean(y1.PACinv1);

% Standard Deviation
stdenval = nanstd(y1.GHIavg);
stdtoval = nanstd(y1.VS);
stdtreval = nanstd(y1.airtemp);
```

```
stdfemval = nanstd(y1.AM2);
stdseksval = nanstd(y1.PACinv1);

% Coefficient of Variation
cvenval = avgenval./stdenval;
cvtoval = avgtoval./stdtoval;
cvtreval = avgtreval./stdtreval;
cvfemval = avgfemval./stdfemval;
cvseksval = avgseksval./stdseksval;

% Descriptve statistics validation set SUNTECH
val_inv1_avg = [avgenval stdenval cvenval; avgtoval stdtoval cvtoval;...
    avgtreval stdtreval cvtreval; avgfemval stdfemval cvfemval;...
    avgseksval stdseksval cvseksval]

    % TESTING
% Mean
avgentest = nanmean(z1.GHIavg);
avgtotest = nanmean(z1.VS);
avgtretest = nanmean(z1.airtemp);
avgfemtest = nanmean(z1.AM2);
avgsekstest = nanmean(z1.PACinv1);

% Standard Deviation
stdentest = nanstd(z1.GHIavg);
stdtotest = nanstd(z1.VS);
stdtretest = nanstd(z1.airtemp);
stdfemtest = nanstd(z1.AM2);
stdsekstest = nanstd(z1.PACinv1);

% Coefficient of Variation
cventest = avgentest./stdentest;
cvtotest = avgtotest./stdtotest;
cvtretest = avgtretest./stdtretest;
cvfemtest = avgfemtest./stdfemtest;
cvsekstest = avgsekstest./stdsekstest;

% Descriptve statistics test set SUNTECH
test_inv1_avg = [avgentest stdentest cventest; avgtotest stdtotest...
    cvtotest; avgtretest stdtretest cvtretest ; avgfemtest stdfemtest...
    cvfemtest; avgsekstest stdsekstest cvsekstest]


train_val1 = [train_inv1_avg,val_inv1_avg test_inv1_avg];
train_val11 = array2table(train_val1);
train_val11.Properties.VariableNames = {'AVG_train' 'STD_train' 'CV_train' 'AVG_val'
'STD_val' 'CV_val' 'AVG_test' 'STD_test' 'CV_test'};

% Percentage difference
```

```
CV1_diff1 = 100.*[(train_val11.CV_train-train_val11.CV_val)./...
    train_val11.CV_val, (train_val11.CV_train-train_val11.CV_test)./...
    train_val11.CV_test, (train_val11.CV_val-train_val11.CV_test)./...
    train_val11.CV_test ];
```

INDEX FOR INV1

idx1 = idx1'; x1.idx1 = trainInd1; y1.idx1 = valInd1; z1.idx1 = testInd1;

```
idx1_train = x1.idx;                        % Training indices
idx1_val = y1.idx;                          % Validation indices
idx1_test = z1.idx;                         % Test indices

time = data_inv1.time;
inv1_norm.time = time;
inv1.time = time;
inv1_norm_t = movevars(inv1_norm,'time','before','GHIavg');
inv1_t = movevars(data_inv1,'time','before','GHIavg');
train_inv1_norm = inv1_norm_t(idx1_train,:);    % Extracting only indeces from training
train_inv1_norm.idx = idx1_train;               % Including indices in table
train_inv1 = inv1_t(idx1_train,:);              % Extracting only indices from training
train_inv1.idx = idx1_train;                    % Including indices in table
val_inv1_norm = inv1_norm_t(idx1_val,:);        % Extracting only indices from
validation
val_inv1_norm.idx = idx1_val;                   % Including indices in table
val_inv1 = inv1_t(idx1_val,:);                  % Extracting only indices from
validation
val_inv1.idx = idx1_val;                        % Including indices in table
test_inv1_norm = inv1_norm_t(idx1_test,:);      % Extracting only indices from testing
test_inv1_norm.idx = idx1_test;                 % including indices in table
test_inv1 = inv1_t(idx1_test,:);                % Extracting only indices from testing
test_inv1.idx = idx1_test;                      % Including indices in tabl
```

Write table SUNTECH

writetable(inv1_norm_t,'inv1_norm.txt'); writetable(inv1_t,'inv1.txt'); writetable(train_inv1_norm,'train_inv1_norm.txt'); writetable(train_inv1,'train_inv1.txt'); writetable(val_inv1_norm,'val_inv1_norm.txt'); writetable(val_inv1,'val_inv1.txt'); writetable(test_inv1_norm,'test_inv1_norm.txt'); writetable(test_inv1,'test_inv1.txt');

A.11 INPUT SELECTION

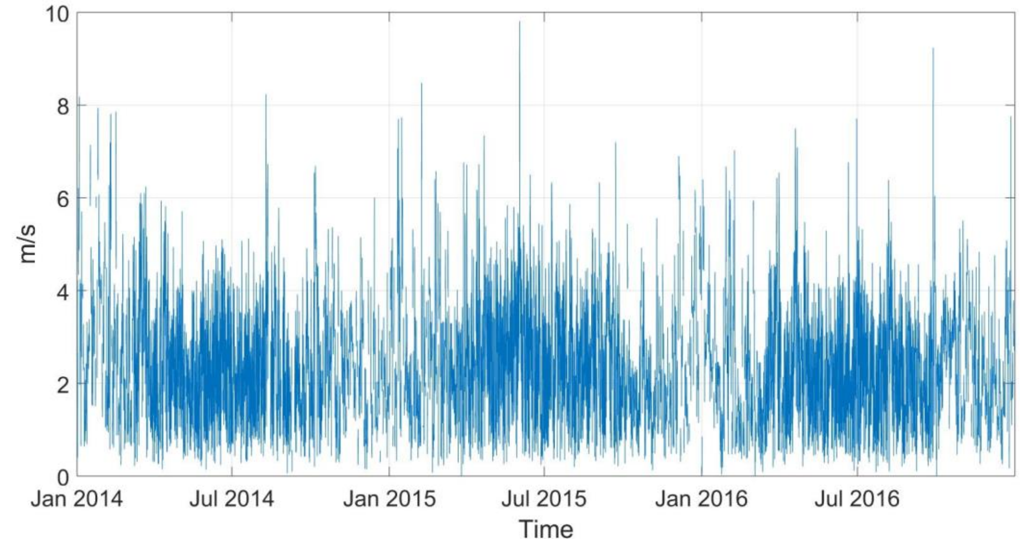


Figure 55: Hourly wind speed for relevant years

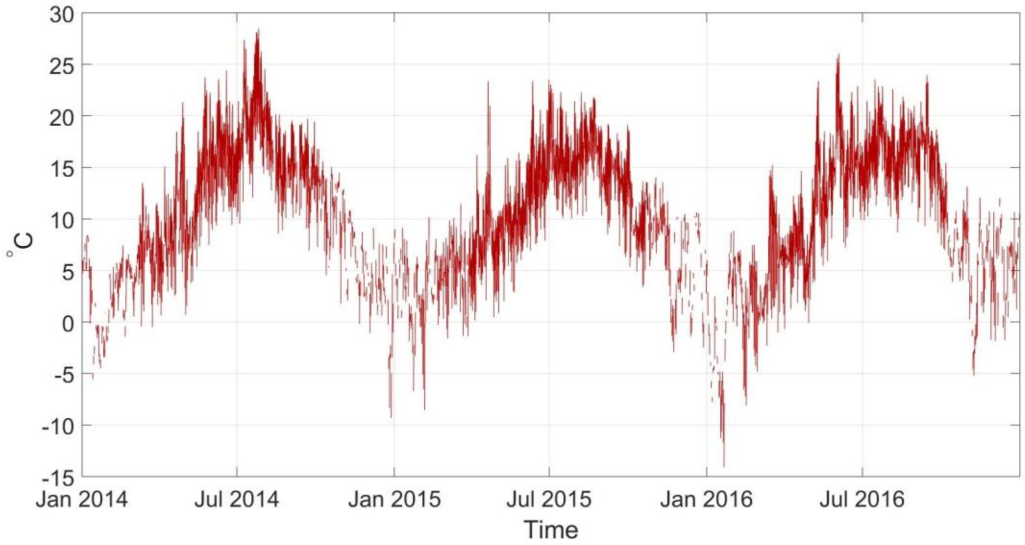


Figure 56: Hourly air temperature for relevant years

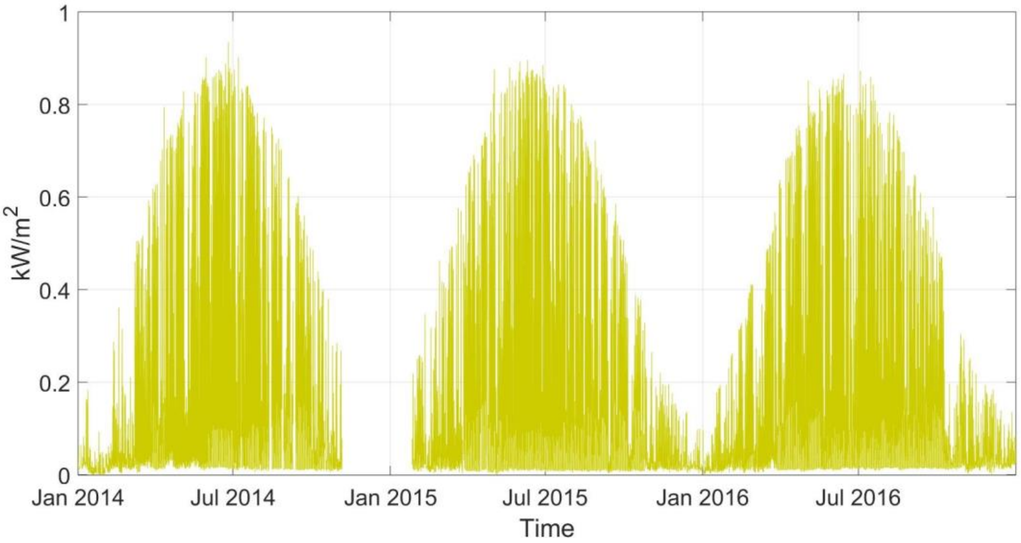


Figure 57: Hourly global horizontal irradiance for relevant years

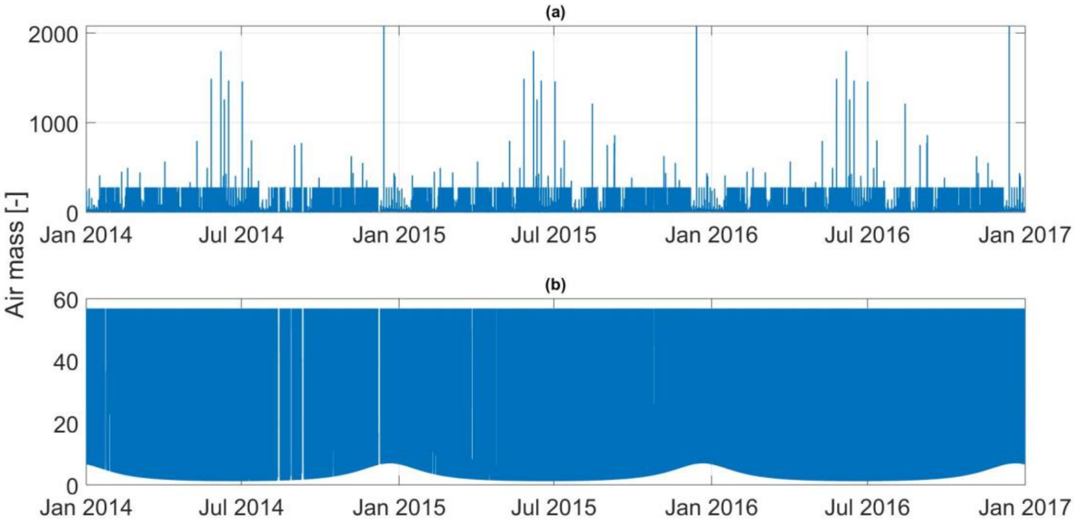


Figure 58: Air mass using scaling (a) and maximum zenith angle (Θ_z) of 89° (b)

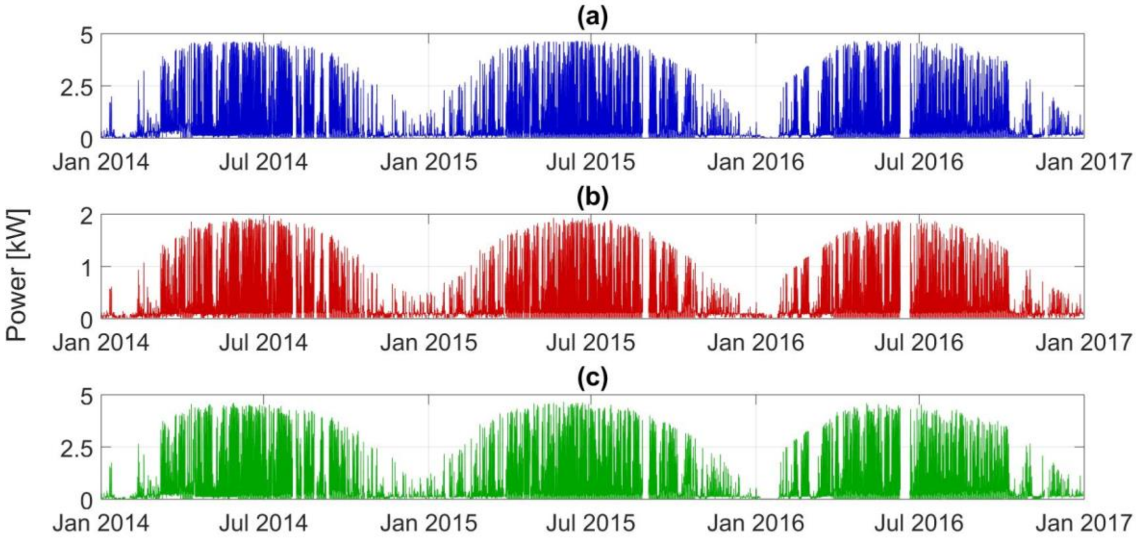


Figure 59: Hourly power production of (a) Suntech, (b) Sharp and (c) REC

A.12 VALIDATION CHECK

Table 23: Performance parameters Suntech modules using 6 and 1000 validation checks

Performance parameters	6 validation checks			1000 validation checks		
	4	10	20_2	4	10	20_2
MSE train	0.0645	0.0538	0.0510	0.0644	0.0515	0.0479
MSE validation	0.0590	0.0504	0.0490	0.0589	0.0485	0.0470
MSE test	0.0579	0.0505	0.0480	0.0578	0.0485	0.0464
MAE train	0.1598	0.1459	0.1402	0.1595	0.1427	0.1367
MAE validation	0.1541	0.1437	0.1403	0.1539	0.1410	0.1380
MAE test	0.1520	0.1428	0.1382	0.1516	0.1402	0.1362
MARE train	1.0875	0.9121	0.9621	1.0656	-0.3081	-0.3054
MARE validation	0.8084	0.7323	0.7563	0.8181	-0.0624	-0.0706
MARE test	0.6880	0.6094	0.6041	0.6838	0.1088	0.1323
MBE train	-0.0005	0.0019	0.0009	-0.0002	0.3081	0.3054
MBE validation	-0.0017	-0.0020	-0.0017	-0.0012	0.0654	0.0706
MBE test	-0.0062	-0.0056	-0.0071	-0.0062	0.1088	0.1323
R^2 train	0.9360	0.9466	0.9494	0.9361	0.9741	0.9760
R^2 validation	0.9397	0.9485	0.9499	0.9398	0.9749	0.9757
R^2 test	0.9411	0.9486	0.9512	0.9412	0.9751	0.9762

A.13 DISCARDED TESTS – 4 AND 120 NEURONS

Table 24: Performance parameters with 4 and 120 neurons, Suntech modules. Averaged 50 runs

Suntech	4 neurons			120 neurons		
	Training	Validation	Test	Training	Validation	Test
MSE	0.0644	0.0589	0.0578	0.0451	0.0504	0.0511
MARE	1.0656	0.8181	0.6838	1.0958	0.7935	0.6105
MAE	0.1595	0.1539	0.1516	0.1315	0.1406	0.1392
MBE	-0.0002	-0.0012	-0.0062	0.0027	0.0005	-0.0060
R^2	0.9361	0.9398	0.9412	0.9552	0.9485	0.9480
Epoch	418			110		
Time	11			423		

A.14 Example of Model Training 50 runs – Suntech Modules

Suntech (20-2-1)															
Runs	MSE train	MSE valida	MSE test	MAE train	MAE valid	MAE test	MARE trai	MARE vali	MARE test	MBE train	MBE valid	MBE test	R^2 train	R^2 valida	R^2 test
1	0.0446	0.0450	0.0430	0.1300	0.1352	0.1298	0.3492	0.0761	0.1322	0.0009	-0.0047	-0.0071	0.9557	0.9540	0.9563
2	0.0445	0.0454	0.0443	0.1299	0.1332	0.1313	0.2157	0.0528	0.1199	0.0022	-0.0018	-0.0031	0.9558	0.9536	0.9549
3	0.0460	0.0462	0.0455	0.1335	0.1352	0.1343	0.6762	0.0891	0.1589	-0.0004	-0.0037	-0.0065	0.9544	0.9528	0.9538
4	0.0482	0.0456	0.0444	0.1345	0.1349	0.1312	0.3968	0.0619	0.1341	0.0001	-0.0032	-0.0093	0.9522	0.9534	0.9548
5	0.0466	0.0449	0.0436	0.1332	0.1332	0.1313	0.3559	0.0742	0.1485	0.0017	-0.0042	-0.0064	0.9538	0.9541	0.9557
6	0.0474	0.0458	0.0448	0.1345	0.1351	0.1329	0.4881	0.0587	0.1526	-0.0003	-0.0024	-0.0084	0.9530	0.9531	0.9544
7	0.0484	0.0473	0.0461	0.1365	0.1384	0.1351	0.3603	0.1028	0.1212	0.0005	-0.0038	-0.0088	0.9519	0.9516	0.9532
8	0.0443	0.0440	0.0449	0.1291	0.1325	0.1308	0.3820	0.0655	0.1312	0.0001	-0.0048	-0.0061	0.9560	0.9550	0.9544
9	0.0442	0.0445	0.0440	0.1293	0.1334	0.1303	0.3272	0.0625	0.1247	-0.0002	-0.0050	-0.0065	0.9561	0.9545	0.9552
10	0.0481	0.0456	0.0456	0.1330	0.1335	0.1311	0.3173	0.1263	0.1479	0.0008	-0.0033	-0.0073	0.9523	0.9534	0.9536
11	0.0453	0.0459	0.0435	0.1294	0.1346	0.1295	0.3309	0.0418	0.1102	0.0012	-0.0044	-0.0054	0.9551	0.9531	0.9557
12	0.0440	0.0444	0.0439	0.1288	0.1325	0.1297	0.2841	0.0443	0.1091	-0.0001	-0.0040	-0.0063	0.9563	0.9547	0.9554
13	0.0473	0.0475	0.0467	0.1358	0.1391	0.1381	0.3669	0.0761	0.0899	0.0006	-0.0037	-0.0085	0.9531	0.9514	0.9525
14	0.0467	0.0463	0.0456	0.1330	0.1363	0.1335	0.4170	0.0677	0.1514	-0.0008	-0.0030	-0.0068	0.9536	0.9527	0.9536
15	0.0464	0.0458	0.0454	0.1309	0.1332	0.1310	0.3435	0.0763	0.1385	0.0001	-0.0036	-0.0081	0.9540	0.9532	0.9538
16	0.0460	0.0458	0.0477	0.1334	0.1362	0.1361	0.4165	0.1011	0.1307	0.0005	-0.0017	-0.0061	0.9544	0.9532	0.9515
17	0.0448	0.0454	0.0451	0.1290	0.1317	0.1315	0.5408	0.0746	0.1313	-0.0006	-0.0050	-0.0074	0.9555	0.9536	0.9541
18	0.0445	0.0446	0.0454	0.1299	0.1339	0.1318	0.3085	0.1104	0.1584	-0.0005	-0.0059	-0.0078	0.9559	0.9544	0.9539
19	0.0460	0.0450	0.0445	0.1333	0.1339	0.1328	0.4199	0.0316	0.1281	0.0011	-0.0016	-0.0057	0.9544	0.9540	0.9548
20	0.0469	0.0468	0.0464	0.1337	0.1363	0.1340	0.3174	0.0584	0.1173	0.0012	-0.0033	-0.0052	0.9535	0.9522	0.9528
21	0.0440	0.0435	0.0439	0.1280	0.1304	0.1302	0.3151	0.0694	0.1510	-0.0002	-0.0044	-0.0051	0.9563	0.9555	0.9554
22	0.0458	0.0474	0.0470	0.1332	0.1378	0.1357	0.2838	0.0756	0.1588	-0.0003	-0.0050	-0.0065	0.9546	0.9515	0.9522
23	0.0458	0.0458	0.0437	0.1314	0.1341	0.1310	0.4987	0.0381	0.1287	0.0008	-0.0021	-0.0074	0.9545	0.9532	0.9556
24	0.0496	0.0471	0.0468	0.1384	0.1379	0.1373	0.2671	0.0712	0.1166	-0.0017	-0.0057	-0.0087	0.9508	0.9519	0.9525
25	0.0466	0.0455	0.0458	0.1334	0.1349	0.1339	0.4524	0.1030	0.1424	0.0006	-0.0023	-0.0070	0.9537	0.9535	0.9534
26	0.0452	0.0453	0.0513	0.1308	0.1340	0.1349	0.2830	0.0537	0.1212	0.0000	-0.0049	-0.0103	0.9551	0.9537	0.9478
27	0.0491	0.0467	0.0525	0.1377	0.1368	0.1389	0.3186	0.0492	0.1067	-0.0002	-0.0051	-0.0101	0.9513	0.9523	0.9466
28	0.0462	0.0456	0.0457	0.1331	0.1364	0.1337	0.4100	0.0992	0.1879	-0.0001	-0.0033	-0.0094	0.9542	0.9534	0.9536
29	0.0456	0.0456	0.0445	0.1304	0.1340	0.1313	0.4728	0.0775	0.1531	0.0010	-0.0028	-0.0056	0.9547	0.9534	0.9548
30	0.0457	0.0445	0.0445	0.1317	0.1338	0.1310	0.4410	0.0326	0.1768	0.0006	-0.0021	-0.0061	0.9547	0.9545	0.9547
31	0.0446	0.0440	0.0442	0.1303	0.1326	0.1313	0.2476	0.0123	0.1409	0.0014	-0.0016	-0.0044	0.9557	0.9551	0.9551
32	0.0443	0.0461	0.0450	0.1292	0.1351	0.1314	0.2739	0.0850	0.1510	0.0000	-0.0054	-0.0066	0.9560	0.9529	0.9543
33	0.0452	0.0446	0.0444	0.1305	0.1325	0.1304	0.3450	0.0387	0.1361	0.0007	-0.0060	-0.0050	0.9552	0.9544	0.9548
34	0.0440	0.0435	0.0452	0.1294	0.1320	0.1316	0.4108	0.0679	0.1616	-0.0001	-0.0047	-0.0071	0.9563	0.9556	0.9540
35	0.0483	0.0457	0.0448	0.1351	0.1351	0.1314	0.3072	0.0888	0.1556	0.0006	-0.0022	-0.0077	0.9521	0.9533	0.9545
36	0.0453	0.0449	0.0446	0.1317	0.1338	0.1326	0.2963	0.0824	0.1431	0.0017	-0.0026	-0.0034	0.9551	0.9541	0.9547
37	0.0475	0.0476	0.0473	0.1367	0.1392	0.1374	0.2443	0.1241	0.1008	-0.0001	-0.0047	-0.0073	0.9529	0.9514	0.9520
38	0.0456	0.0441	0.0434	0.1303	0.1323	0.1296	0.3894	0.0860	0.1337	0.0010	-0.0018	-0.0073	0.9548	0.9549	0.9559
39	0.0454	0.0456	0.0444	0.1306	0.1341	0.1311	0.4147	0.0487	0.1510	-0.0014	-0.0051	-0.0088	0.9550	0.9534	0.9549
40	0.0443	0.0453	0.0472	0.1295	0.1345	0.1340	0.3428	0.0719	0.1441	0.0002	-0.0043	-0.0074	0.9560	0.9538	0.9521
41	0.0466	0.0458	0.0451	0.1334	0.1349	0.1323	0.2951	0.0761	0.1396	0.0009	-0.0012	-0.0073	0.9538	0.9532	0.9542
42	0.0461	0.0458	0.0435	0.1329	0.1331	0.1312	0.3452	0.0389	0.1473	-0.0013	-0.0058	-0.0101	0.9543	0.9532	0.9558
43	0.0455	0.0460	0.0452	0.1312	0.1363	0.1339	0.3051	0.0899	0.1545	0.0000	-0.0049	-0.0085	0.9549	0.9530	0.9540
44	0.0542	0.0502	0.0484	0.1436	0.1405	0.1379	0.0554	0.0161	0.1681	0.0021	-0.0006	-0.0067	0.9463	0.9487	0.9508
45	0.0447	0.0437	0.0439	0.1299	0.1314	0.1308	0.3930	0.0336	0.1178	0.0007	-0.0059	-0.0059	0.9556	0.9553	0.9553
46	0.0450	0.0451	0.0455	0.1312	0.1339	0.1335	0.4720	0.1073	0.1382	0.0009	-0.0028	-0.0053	0.9553	0.9539	0.9537
47	0.0460	0.0449	0.0451	0.1311	0.1324	0.1317	0.2466	0.0617	0.1374	0.0002	-0.0035	-0.0067	0.9544	0.9541	0.9542
48	0.0460	0.0473	0.0460	0.1341	0.1383	0.1363	0.2325	0.1029	0.1391	0.0002	-0.0057	-0.0056	0.9544	0.9517	0.9532
49	0.0460	0.0456	0.0453	0.1330	0.1350	0.1340	0.3595	0.0903	0.1708	-0.0011	-0.0057	-0.0094	0.9544	0.9534	0.9540
50	0.0463	0.0450	0.0439	0.1315	0.1337	0.1306	0.3207	0.0636	0.1512	0.0000	-0.0042	-0.0071	0.9541	0.9540	0.9553

A.15 MODEL TRAINING RESULTS – EXAMPLE SUNTECH

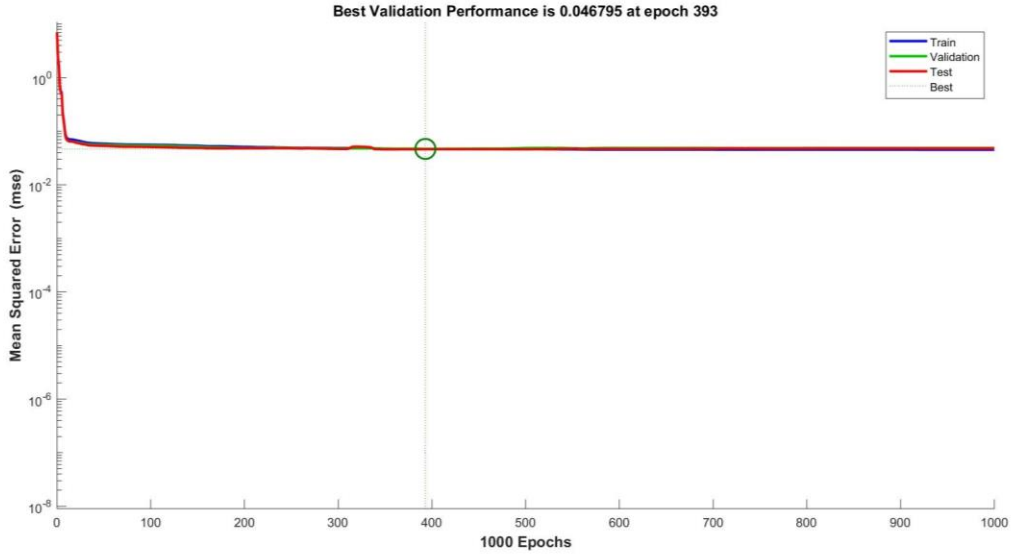


Figure 60: Performance of 20-2-1 model Suntech

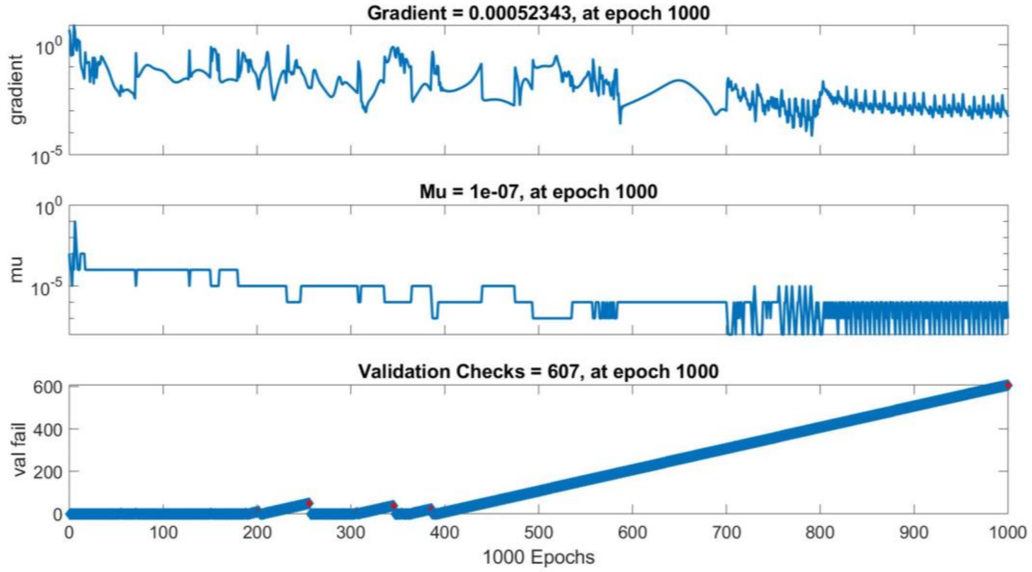


Figure 61: Training state 20-2-1 model Suntech

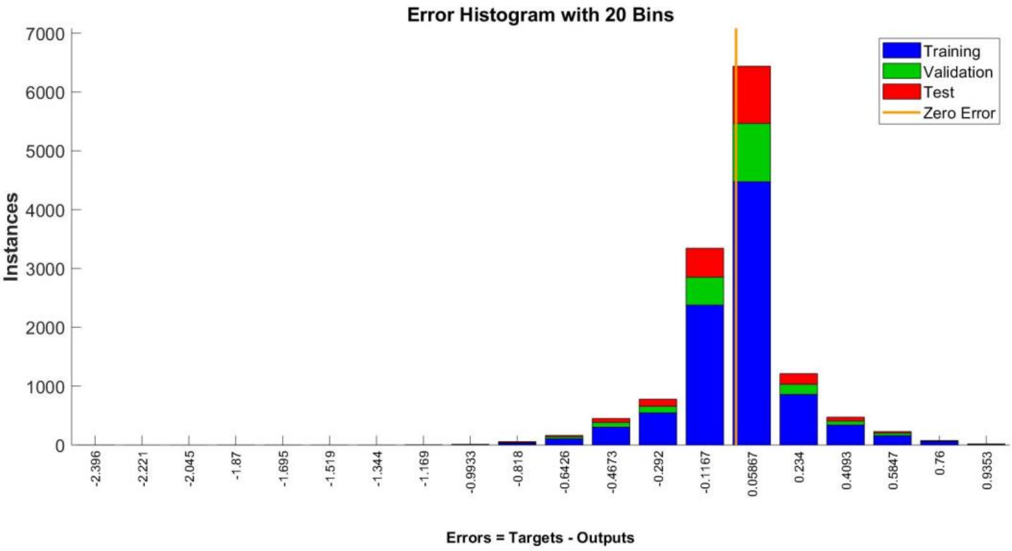


Figure 62: Error histogram 20-2-1 model Suntech

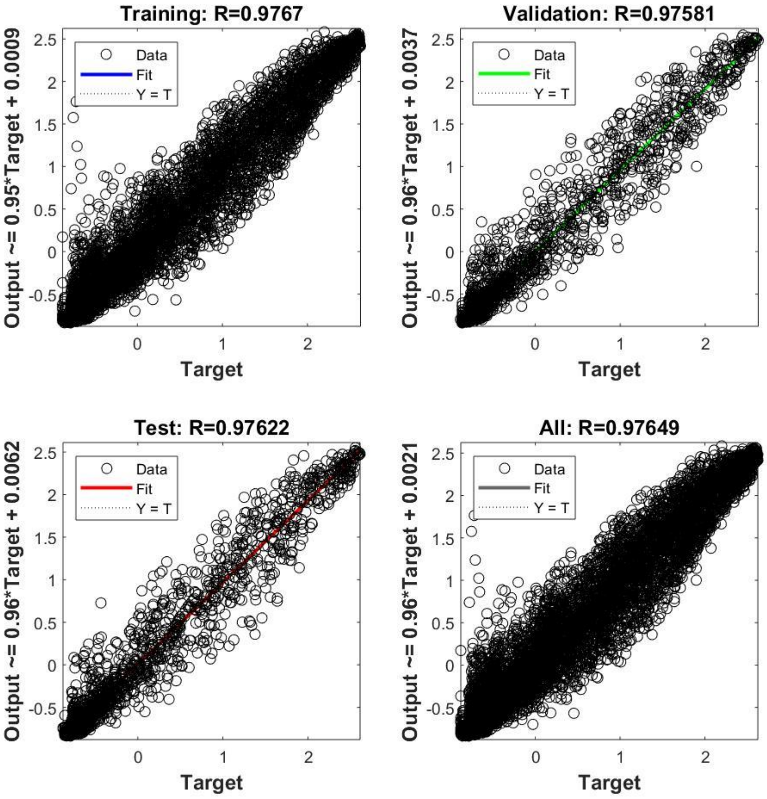


Figure 63: Regression plot 20-2-1 model Suntech

A.16 PERFORMANCE PARAMETERS – SCALED AIR MASS

Table 25: Performance parameters Suntech PV modules using scaled air mass

Suntech		MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0623	0.1697	0.1538	0.0007	0.9382	443	19
	Validation set	0.0578	0.0584	0.1500	-0.0005	0.9410		
	Test set	0.0556	0.1614	0.1469	-0.0064	0.9435		
10	**Train set**	0.0615	0.1382	0.1527	0.0012	0.9390	520	19
	Validation set	0.0574	0.0447	0.1492	0.0003	0.9414		
	Test set	0.0553	0.1623	0.1463	-0.0064	0.9438		
20	**Train set**	0.0586	0.1315	0.1488	0.0017	0.9418	542	35
	Validation set	0.0567	0.0379	0.1484	0.0009	0.9420		
	Test set	0.0548	0.1645	0.1453	-0.0061	0.9443		
30	**Train set**	0.0574	0.1410	0.1472	0.0016	0.9431	551	60
	Validation set	0.0570	0.0340	0.1481	0.0020	0.9417		
	Test set	0.0557	0.1627	0.1457	-0.0056	0.9433		
50	**Train set**	0.0549	0.1563	0.1448	0.0023	0.9456	522	104
	Validation set	0.0576	0.0268	0.1489	0.0027	0.9412		
	Test set	0.0570	0.1564	0.1467	-0.0051	0.9420		
120	**Train set**	0.0599	0.1154	0.1506	0.0013	0.9406	48	478
	Validation set	0.0603	0.0946	0.1521	0.0024	0.9384		
	Test set	0.0596	0.1630	0.1509	-0.0056	0.9394		
10_2	**Train set**	0.0588	0.1851	0.1493	0.0002	0.9417	737	25
	Validation set	0.0552	0.0337	0.1465	-0.0014	0.9436		
	Test set	0.0533	0.1601	0.1437	-0.0077	0.9459		
20_2	**Train set**	0.0568	0.1864	0.1470	0.0006	0.9436	650	36
	Validation set	0.0557	0.0394	0.1470	0.0001	0.9430		
	Test set	0.0548	0.1672	0.1446	-0.0075	0.9443		

Table 26: Performance parameters Sharp PV modules using scaled air mass

Sharp		MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0343	0.7801	0.1190	-0.0001	0.9656	555	17
	Validation set	0.0337	0.0173	0.1191	0.0019	0.9664		
	Test set	0.0396	0.0659	0.1278	0.0010	0.9606		
10	**Train set**	0.0339	0.7231	0.1172	0.0005	0.9661	619	24
	Validation set	0.0334	0.0172	0.1177	0.0023	0.9667		
	Test set	0.0393	0.0679	0.1263	0.0016	0.9609		
20	**Train set**	0.0325	0.6440	0.1134	0.0010	0.9675	684	25
	Validation set	0.0332	0.0294	0.1162	0.0024	0.9669		
	Test set	0.0390	0.0638	0.1243	0.0026	0.9612		
30	**Train set**	0.0318	0.5922	0.1113	0.0012	0.9681	745	70
	Validation set	0.0336	0.0245	0.1156	0.0022	0.9665		
	Test set	0.0393	0.0608	0.1234	0.0026	0.9609		
50	**Train set**	0.0315	0.7098	0.1109	0.0011	0.9685	546	112
	Validation set	0.0345	0.0152	0.1172	0.0016	0.9656		
	Test set	0.0404	0.0564	0.1252	0.0019	0.9598		
120	**Train set**	0.0317	1.0081	0.1134	0.0024	0.9682	218	433
	Validation set	0.0362	0.0194	0.1219	0.0029	0.9639		
	Test set	0.0426	0.0441	0.1309	0.0013	0.9576		
10_2	**Train set**	0.0336	0.6576	0.1159	0.0001	0.9663	808	26
	Validation set	0.0333	0.0200	0.1167	0.0018	0.9668		
	Test set	0.0391	0.0712	0.1250	0.0011	0.9611		
20_2	**Train set**	0.0327	0.7324	0.1140	0.0004	0.9672	724	27
	Validation set	0.0335	0.0228	0.1165	0.0014	0.9666		
	Test set	0.0394	0.0653	0.1246	0.0013	0.9608		

Table 27: Performance parameters REC PV modules using scaled air mass

REC		MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0547	0.0976	0.1349	0.0010	0.9459	511	17
	Validation set	0.0565	0.0714	0.1310	0.0013	0.9404		
	Test set	0.0588	0.7558	0.1380	-0.0021	0.9412		
10	**Train set**	0.0540	0.0951	0.1337	0.0010	0.9466	481	18
	Validation set	0.0561	0.0719	0.1301	0.0015	0.9409		
	Test set	0.0583	0.7622	0.1370	-0.0023	0.9417		
20	**Train set**	0.0528	0.0804	0.1325	0.0018	0.9477	414	51
	Validation set	0.0563	0.0824	0.1306	0.0025	0.9406		
	Test set	0.0578	0.6797	0.1364	-0.0018	0.9422		
30	**Train set**	0.0526	0.0793	0.1324	0.0024	0.9480	342	76
	Validation set	0.0567	0.0858	0.1314	0.0031	0.9402		
	Test set	0.0580	0.6227	0.1366	-0.0010	0.9420		
50	**Train set**	0.0523	0.0773	0.1325	0.0031	0.9482	237	80
	Validation set	0.0571	0.0971	0.1329	0.0039	0.9397		
	Test set	0.0586	0.5461	0.1377	-0.0002	0.9415		
120	**Train set**	0.0519	0.0719	0.1326	0.0033	0.9486	115	474
	Validation set	0.0583	0.0983	0.1355	0.0041	0.9385		
	Test set	0.0595	0.4389	0.1390	0.0003	0.9405		
10_2	**Train set**	0.0532	0.0832	0.1327	0.0009	0.9473	771	24
	Validation set	0.0558	0.0704	0.1294	0.0012	0.9412		
	Test set	0.0576	0.7121	0.1359	-0.0024	0.9424		
20_2	**Train set**	0.0522	0.0788	0.1317	0.0014	0.9484	519	37
	Validation set	0.0559	0.0802	0.1300	0.0019	0.9410		
	Test set	0.0577	0.5891	0.1356	-0.0022	0.9423		

A.17 Performance Parameters – Air Mass ($\Theta_Z \leq 89°$)

Table 28: Performance parameters Suntech PV module using air mass with $\Theta_Z \leq 89°$

Neurons	Set	MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0543	0.2892	0.1469	0.0018	0.9462	313	19
	Validation set	0.0506	0.0678	0.1443	-0.0027	0.9483		
	Test set	0.0503	0.1167	0.1434	-0.0054	0.9489		
10	**Train set**	0.0515	0.3081	0.1427	0.0025	0.9489	333	25
	Validation set	0.0485	0.0624	0.1410	-0.0026	0.9504		
	Test set	0.0485	0.1088	0.1402	-0.0047	0.9507		
20	**Train set**	0.0479	0.3054	0.1367	0.0028	0.9525	381	34
	Validation set	0.0470	0.0706	0.1380	-0.0021	0.9520		
	Test set	0.0464	0.1323	0.1362	-0.0043	0.9529		
30	**Train set**	0.0458	0.3568	0.1335	0.0026	0.9545	415	27
	Validation set	0.0470	0.0700	0.1377	-0.0021	0.9520		
	Test set	0.0472	0.1388	0.1360	-0.0048	0.9521		
50	**Train set**	0.0444	0.4102	0.1311	0.0026	0.9559	522	104
	Validation set	0.0474	0.0485	0.1376	-0.0009	0.9515		
	Test set	0.0481	0.1405	0.1366	-0.0050	0.9511		
10_2	**Train set**	0.0494	0.3259	0.1375	0.0005	0.9510	595	22
	Validation set	0.0469	0.0738	0.1370	-0.0043	0.9521		
	Test set	0.0468	0.1225	0.1357	-0.0071	0.9524		
20_2	**Train set**	0.0461	0.3531	0.1323	0.0003	0.9543	584	23
	Validation set	0.0456	0.0695	0.1347	-0.0038	0.9534		
	Test set	0.0454	0.1392	0.1327	-0.0070	0.9539		

Table 29: Performance parameters Sharp PV module using air mass with $\Theta_Z \leq 89°$

Neurons	Set	MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0299	0.3593	0.1122	0.0006	0.9700	528	20
	Validation set	0.0289	0.0506	0.1114	0.0031	0.9712		
	Test set	0.0351	0.0567	0.1223	-0.0006	0.9650		
10	**Train set**	0.0289	0.2337	0.1092	0.0006	0.9711	456	20
	Validation set	0.0278	0.0564	0.1083	0.0029	0.9723		
	Test set	0.0343	0.0529	0.1195	-0.0012	0.9659		
20	**Train set**	0.0272	0.2362	0.1030	0.0008	0.9728	482	20
	Validation set	0.0268	0.0674	0.1038	0.0021	0.9733		
	Test set	0.0328	0.0561	0.1140	-0.0003	0.9673		
30	**Train set**	0.0265	0.3269	0.1015	0.0009	0.9735	387	30
	Validation set	0.0270	0.0683	0.1035	0.0017	0.9731		
	Test set	0.0328	0.0585	0.1138	0.0001	0.9673		
50	**Train set**	0.0252	0.3477	0.0987	0.0008	0.9747	365	58
	Validation set	0.0273	0.0732	0.1034	0.0009	0.9728		
	Test set	0.0330	0.0584	0.1133	0.0004	0.9671		
10_2	**Train set**	0.0280	0.3430	0.1056	0.0000	0.9719	550	26
	Validation set	0.0273	0.0559	0.1055	0.0016	0.9727		
	Test set	0.0333	0.0567	0.1154	-0.0012	0.9668		
20_2	**Train set**	0.0265	0.3394	0.1014	0.0002	0.9735	494	41
	Validation set	0.0268	0.0573	0.1029	0.0007	0.9733		
	Test set	0.0325	0.0576	0.1124	-0.0008	0.9676		

Table 30: Performance parameters REC PV module using air mass with $\Theta_Z \leq 89°$

Neurons	Set	MSE	MARE	MAE	MBE	R^2	Epoch	Time
8	**Train set**	0.0474	0.1033	0.1259	0.0011	0.9531	283	14
	Validation set	0.0508	0.0514	0.1259	0.0004	0.9465		
	Test set	0.0514	0.8000	0.1291	-0.0007	0.9486		
10	**Train set**	0.0464	0.0913	0.1245	0.0018	0.9541	325	16
	Validation set	0.0502	0.0459	0.1249	0.0014	0.9471		
	Test set	0.0511	0.8166	0.1283	-0.0008	0.9489		
20	**Train set**	0.0439	0.0646	0.1206	0.0018	0.9565	380	25
	Validation set	0.0491	0.0458	0.1217	0.0017	0.9482		
	Test set	0.0497	0.7454	0.1253	-0.0017	0.9503		
30	**Train set**	0.0428	0.0599	0.1187	0.0017	0.9576	335	40
	Validation set	0.0488	0.0460	0.1210	0.0014	0.9485		
	Test set	0.0496	0.7055	0.1244	-0.0018	0.9505		
50	**Train set**	0.0423	0.0632	0.1176	0.0025	0.9582	119	81
	Validation set	0.0489	0.0548	0.1209	0.0019	0.9484		
	Test set	0.0503	0.6384	0.1252	-0.0014	0.9497		
10_2	**Train set**	0.0448	0.0823	0.1206	0.0004	0.9557	651	21
	Validation set	0.0488	0.0339	0.1208	-0.0001	0.9485		
	Test set	0.0500	0.8118	0.1245	-0.0026	0.9500		
20_2	**Train set**	0.0424	0.0724	0.1173	0.0007	0.9580	420	32
	Validation set	0.0480	0.0246	0.1191	0.0002	0.9494		
	Test set	0.0492	0.7549	0.1223	-0.0033	0.9508		

A.18 Model Training Results – Example Sharp

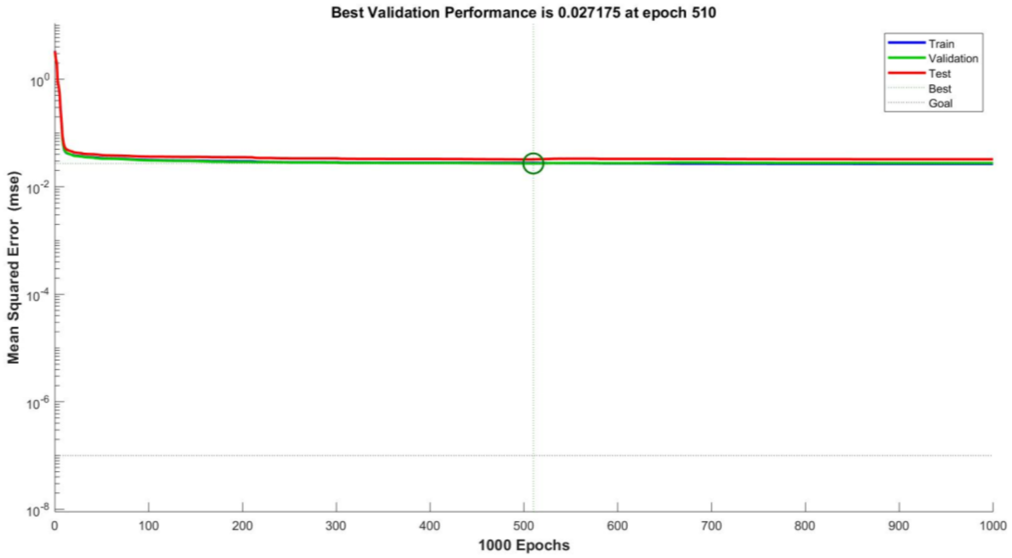


Figure 64: Performance 20-2-1 model Sharp

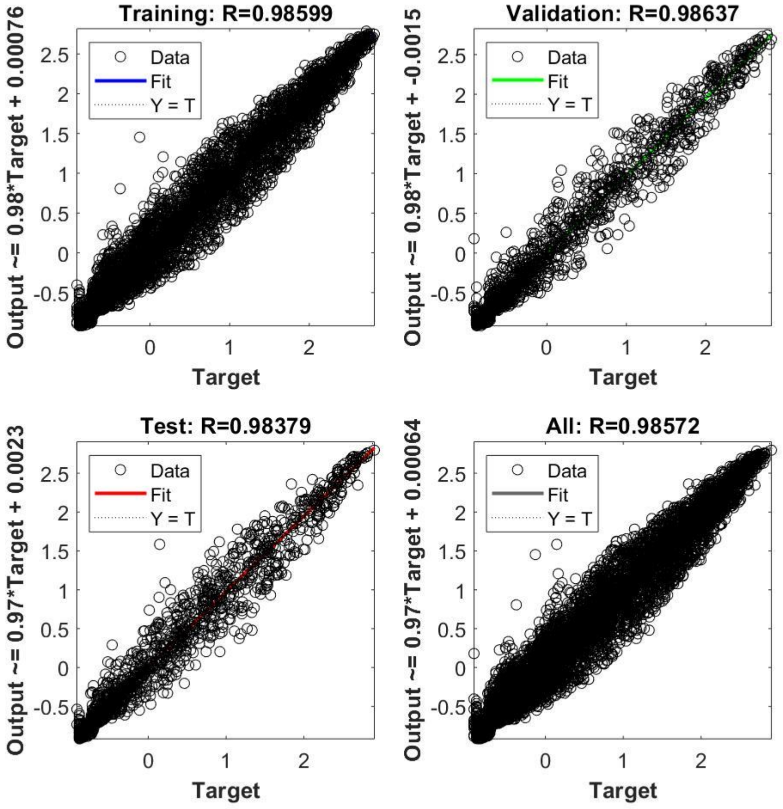


Figure 65: Regression plot 20-2-1 model Sharp

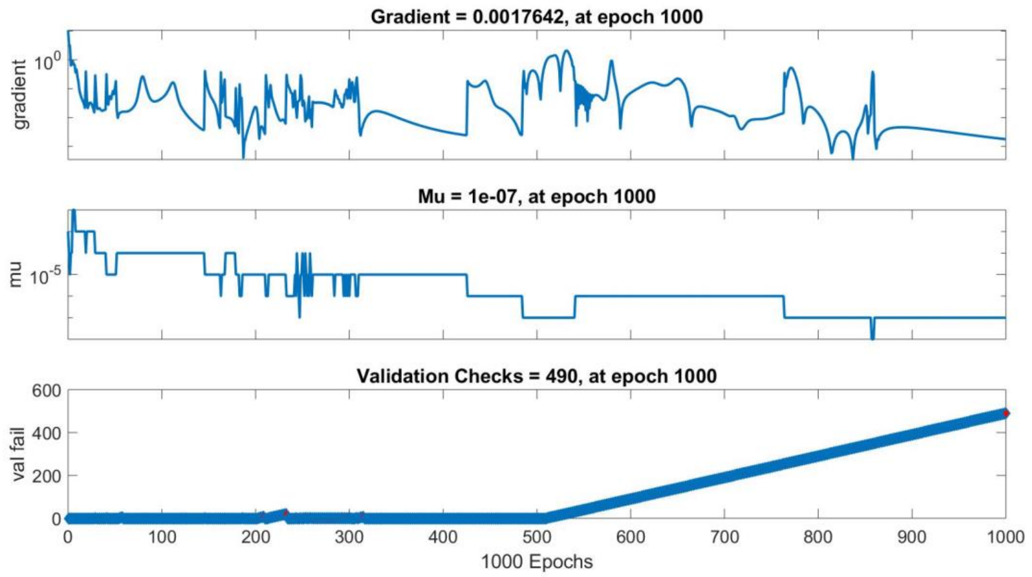


Figure 66: Training state 20-2-1 model Sharp

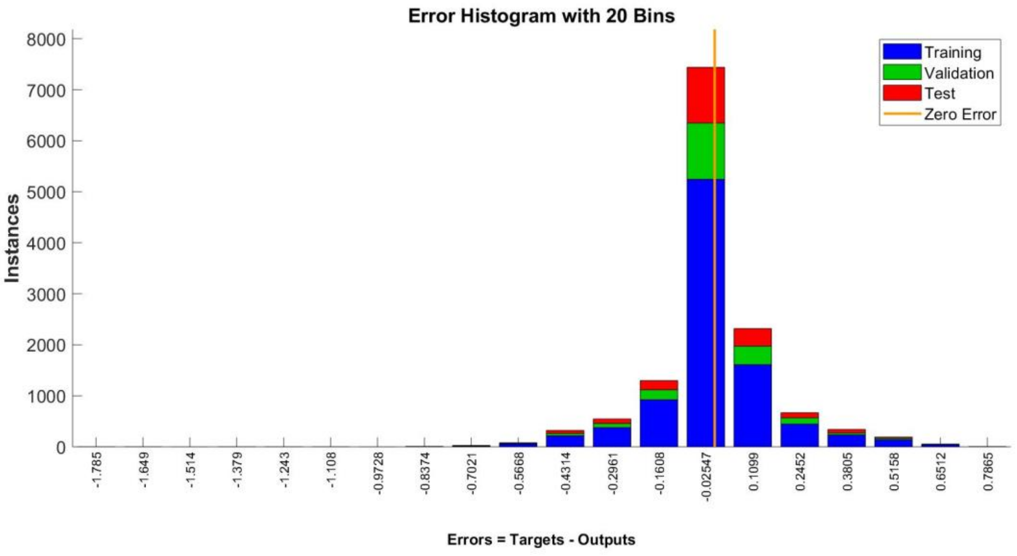


Figure 67: Error histogram 20-2-1 model Sharp

A.19 EXAMPLE OF MODEL TRAINING 50 RUNS – SHARP MODULES

Sharp (20-2-1)															
Runs	**MSE train**	**MSE valida**	**MSE test**	**MAE train**	**MAE valid**	**MAE test**	**MARE trai**	**MARE vali**	**MARE test**	**MBE train**	**MBE valida**	**MBE test**	**R^2 train**	**R^2 valida**	**R^2 test**
1	0.0265	0.0262	0.0322	0.1019	0.1018	0.1133	0.3087	0.0458	0.0516	-0.0005	0.0003	-0.0022	0.9735	0.9739	0.9680
2	0.0269	0.0261	0.0329	0.1018	0.1024	0.1138	0.4083	0.0658	0.0573	0.0018	0.0025	-0.0011	0.9730	0.9740	0.9673
3	0.0273	0.0272	0.0330	0.1001	0.1021	0.1112	0.5038	0.0721	0.0696	0.0002	0.0004	-0.0017	0.9726	0.9729	0.9671
4	0.0264	0.0257	0.0316	0.0998	0.1002	0.1097	0.5123	0.0291	0.0577	0.0003	0.0010	-0.0004	0.9736	0.9744	0.9685
5	0.0258	0.0261	0.0312	0.1013	0.1027	0.1109	0.4583	0.0394	0.0575	0.0020	0.0027	0.0011	0.9742	0.9740	0.9689
6	0.0259	0.0265	0.0315	0.1025	0.1036	0.1122	0.0924	0.0138	0.0482	-0.0013	-0.0023	-0.0019	0.9740	0.9736	0.9686
7	0.0254	0.0260	0.0316	0.0971	0.0996	0.1081	0.1229	0.0689	0.0563	0.0015	0.0007	-0.0002	0.9745	0.9741	0.9686
8	0.0268	0.0263	0.0322	0.1007	0.1021	0.1116	0.1158	0.0597	0.0596	0.0004	0.0004	-0.0012	0.9732	0.9738	0.9680
9	0.0287	0.0286	0.0347	0.1088	0.1092	0.1204	0.2256	0.0441	0.0624	-0.0005	-0.0007	0.0005	0.9712	0.9715	0.9655
10	0.0255	0.0276	0.0319	0.1019	0.1055	0.1126	0.0605	0.0600	0.0583	0.0002	0.0003	-0.0020	0.9745	0.9725	0.9682
11	0.0255	0.0259	0.0315	0.0972	0.0998	0.1080	0.6667	0.0607	0.0576	0.0001	-0.0002	-0.0021	0.9745	0.9742	0.9687
12	0.0260	0.0273	0.0337	0.1013	0.1042	0.1141	0.0550	0.0463	0.0551	0.0002	0.0003	0.0008	0.9740	0.9728	0.9664
13	0.0286	0.0279	0.0344	0.1067	0.1067	0.1185	0.4770	0.0712	0.0647	0.0007	0.0004	0.0004	0.9714	0.9722	0.9657
14	0.0253	0.0264	0.0317	0.1012	0.1033	0.1131	0.0666	0.0010	0.0429	-0.0016	-0.0022	-0.0016	0.9746	0.9737	0.9684
15	0.0268	0.0275	0.0327	0.1047	0.1057	0.1152	0.3314	0.0633	0.0647	0.0001	0.0011	-0.0003	0.9731	0.9725	0.9674
16	0.0267	0.0264	0.0325	0.1033	0.1037	0.1146	0.5625	0.0419	0.0438	-0.0004	0.0002	-0.0020	0.9732	0.9737	0.9677
17	0.0300	0.0299	0.0353	0.1122	0.1130	0.1233	0.5858	0.0606	0.0623	-0.0001	0.0011	-0.0006	0.9700	0.9702	0.9649
18	0.0261	0.0263	0.0318	0.1003	0.1017	0.1102	0.6497	0.0556	0.0600	-0.0003	0.0007	-0.0013	0.9738	0.9738	0.9683
19	0.0253	0.0262	0.0319	0.0960	0.0992	0.1077	0.2629	0.0703	0.0616	0.0001	0.0002	-0.0012	0.9747	0.9739	0.9682
20	0.0267	0.0262	0.0320	0.1007	0.1016	0.1107	0.4548	0.0374	0.0522	0.0012	0.0005	0.0005	0.9733	0.9739	0.9681
21	0.0264	0.0263	0.0327	0.0985	0.1008	0.1104	0.0381	0.0707	0.0636	0.0000	0.0013	-0.0020	0.9736	0.9738	0.9674
22	0.0248	0.0259	0.0312	0.0955	0.0990	0.1077	0.5358	0.0539	0.0518	-0.0002	0.0005	-0.0014	0.9752	0.9741	0.9690
23	0.0263	0.0258	0.0313	0.0986	0.0997	0.1088	0.1745	0.0531	0.0544	-0.0004	0.0009	-0.0025	0.9737	0.9743	0.9688
24	0.0279	0.0279	0.0333	0.1056	0.1063	0.1161	0.1447	0.0781	0.0617	0.0010	0.0004	0.0009	0.9721	0.9722	0.9669
25	0.0263	0.0260	0.0313	0.1023	0.1027	0.1126	0.5108	0.0469	0.0663	0.0005	0.0018	0.0003	0.9737	0.9741	0.9688
26	0.0255	0.0271	0.0326	0.0972	0.1004	0.1094	0.5855	0.0615	0.0720	-0.0015	-0.0003	-0.0029	0.9745	0.9730	0.9675
27	0.0255	0.0260	0.0320	0.0989	0.0997	0.1104	0.5393	0.0412	0.0562	0.0004	0.0002	0.0003	0.9744	0.9741	0.9681
28	0.0262	0.0260	0.0318	0.0987	0.1002	0.1095	0.1452	0.0811	0.0681	-0.0007	0.0007	-0.0010	0.9738	0.9741	0.9684
29	0.0273	0.0272	0.0324	0.1047	0.1055	0.1149	0.9903	0.0654	0.0588	0.0011	0.0022	0.0021	0.9727	0.9729	0.9677
30	0.0269	0.0266	0.0406	0.1049	0.1047	0.1175	0.1547	0.0718	0.0521	0.0010	0.0027	-0.0015	0.9730	0.9735	0.9596
31	0.0274	0.0266	0.0326	0.1056	0.1050	0.1155	0.4318	0.0723	0.0500	0.0006	0.0025	0.0003	0.9725	0.9735	0.9675
32	0.0260	0.0266	0.0311	0.1004	0.1023	0.1103	0.1370	0.0303	0.0496	-0.0002	-0.0006	-0.0008	0.9740	0.9735	0.9691
33	0.0260	0.0266	0.0321	0.1027	0.1038	0.1132	0.2805	0.0475	0.0366	-0.0001	-0.0001	-0.0012	0.9740	0.9735	0.9681
34	0.0266	0.0264	0.0324	0.1033	0.1037	0.1145	0.3722	0.0137	0.0587	0.0003	0.0016	-0.0021	0.9734	0.9737	0.9677
35	0.0254	0.0281	0.0312	0.0979	0.1017	0.1074	0.1739	0.0618	0.0637	0.0001	0.0006	-0.0004	0.9745	0.9720	0.9689
36	0.0262	0.0262	0.0326	0.1038	0.1041	0.1153	0.6889	0.0705	0.0518	0.0000	0.0009	-0.0011	0.9738	0.9738	0.9676
37	0.0262	0.0265	0.0319	0.1002	0.1023	0.1109	0.1613	0.0612	0.0537	0.0014	0.0011	0.0005	0.9738	0.9736	0.9683
38	0.0274	0.0273	0.0336	0.1078	0.1081	0.1204	0.0511	0.1008	0.0504	-0.0009	-0.0005	-0.0014	0.9725	0.9728	0.9665
39	0.0281	0.0274	0.0331	0.1085	0.1082	0.1192	0.1142	0.0744	0.0567	-0.0001	0.0005	-0.0016	0.9719	0.9727	0.9670
40	0.0267	0.0260	0.0320	0.0989	0.0999	0.1082	0.5253	0.1059	0.0561	0.0000	0.0013	-0.0019	0.9733	0.9741	0.9681
41	0.0266	0.0318	0.0319	0.0985	0.1016	0.1091	0.2199	0.0716	0.0576	0.0011	0.0005	-0.0002	0.9733	0.9683	0.9682
42	0.0263	0.0261	0.0317	0.1020	0.1027	0.1124	0.5993	0.0714	0.0638	0.0003	0.0017	0.0006	0.9737	0.9740	0.9685
43	0.0263	0.0263	0.0323	0.0995	0.1009	0.1103	0.4096	0.0465	0.0632	0.0005	0.0018	0.0008	0.9736	0.9738	0.9679
44	0.0275	0.0281	0.0330	0.1010	0.1035	0.1114	0.2869	0.0507	0.0469	0.0002	0.0015	0.0001	0.9725	0.9720	0.9671
45	0.0267	0.0261	0.0325	0.1024	0.1026	0.1135	0.2404	0.0491	0.0622	-0.0012	0.0000	-0.0033	0.9732	0.9740	0.9676
46	0.0259	0.0275	0.0327	0.0986	0.1021	0.1110	0.7332	0.0577	0.0681	0.0005	-0.0001	-0.0017	0.9740	0.9726	0.9674
47	0.0250	0.0266	0.0317	0.0960	0.1004	0.1071	0.3893	0.0615	0.0624	0.0002	0.0001	-0.0009	0.9749	0.9735	0.9685
48	0.0258	0.0263	0.0322	0.0987	0.1004	0.1108	0.1549	0.0746	0.0604	0.0007	0.0003	0.0001	0.9741	0.9738	0.9680
49	0.0253	0.0255	0.0317	0.0977	0.0993	0.1091	0.0653	0.0333	0.0475	0.0004	0.0006	-0.0006	0.9746	0.9746	0.9685
50	0.0280	0.0275	0.0335	0.1022	0.1031	0.1136	0.1972	0.0789	0.0702	0.0018	0.0012	0.0012	0.9719	0.9726	0.9667

A.20 MODEL TRAINING RESULTS – EXAMPLE REC

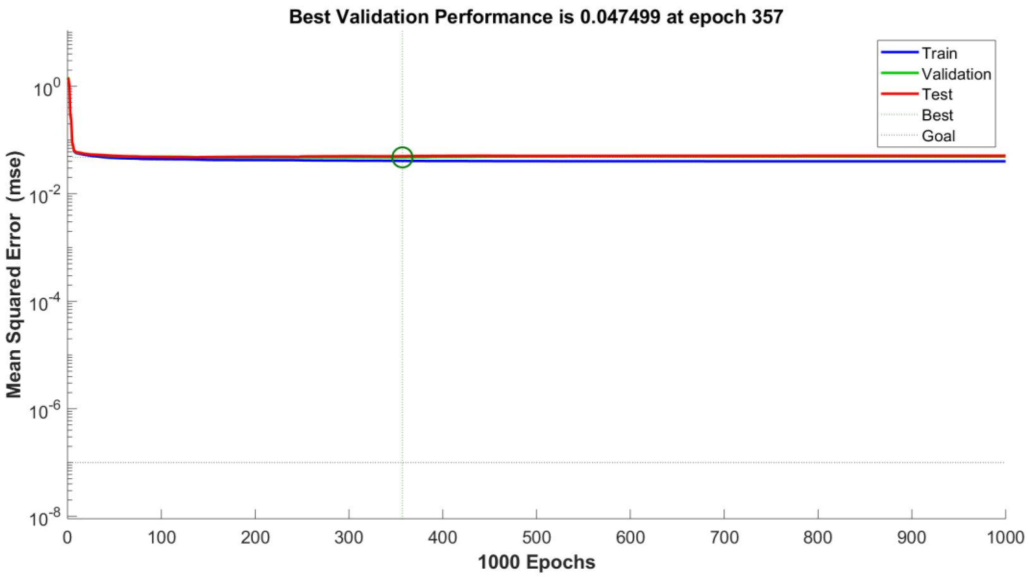


Figure 68: Performance 20-2-1 model REC

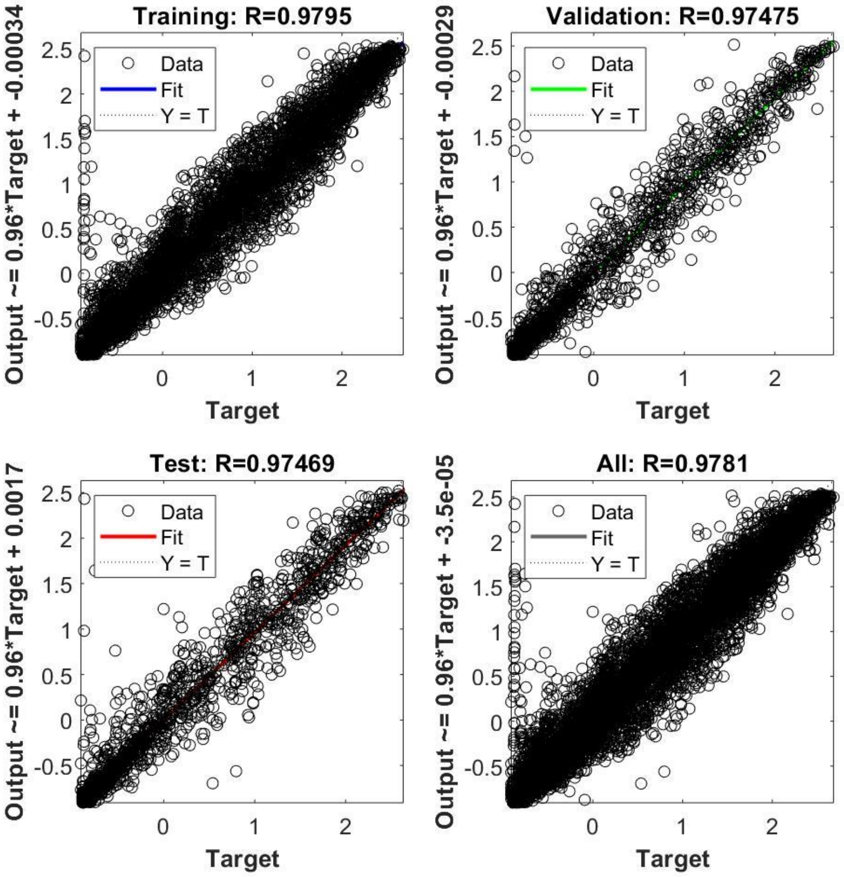


Figure 69: Regression plot 20-2-1 model REC

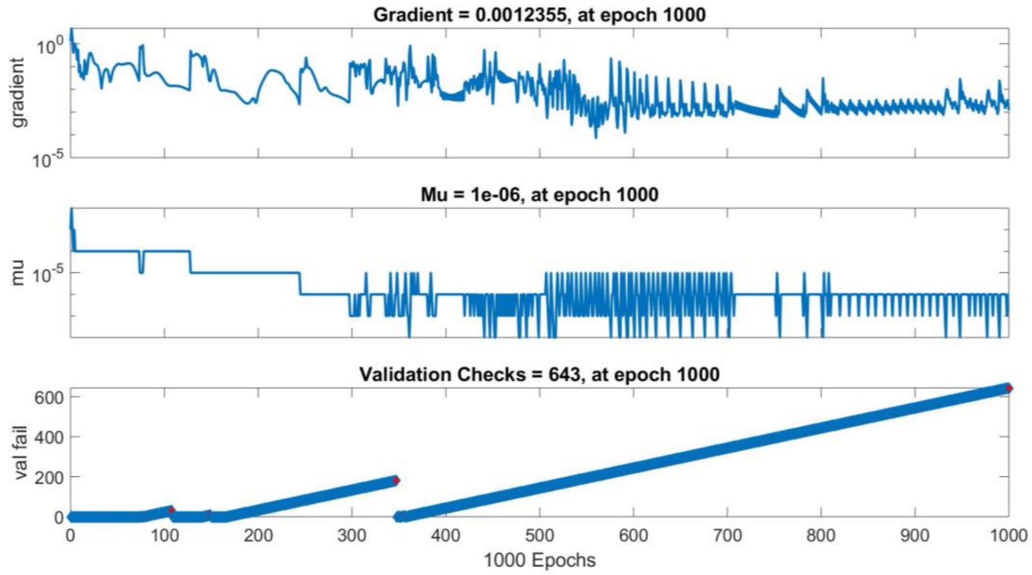


Figure 70: Training state 20-2-1 model REC

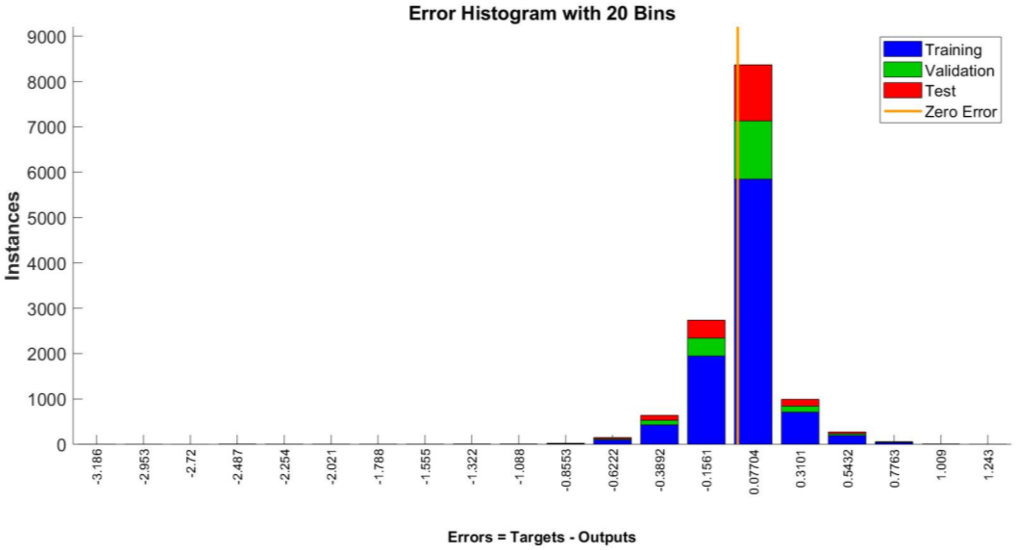


Figure 71: Error histogram 20-2-1 model REC

A.21 EXAMPLE OF MODEL TRAINING 50 RUNS – REC MODULES

REC (20-2-1) Modules															
Runs	**MSE train**	**MSE valida**	**MSE test**	**MAE train**	**MAE valid**	**MAE test**	**MARE trai**	**MARE vali**	**MARE test**	**MBE train**	**MBE valida**	**MBE test**	**R^2 train**	**R^2 valida**	**R^2 test**
1	0.0421	0.0477	0.0483	0.1177	0.1201	0.1223	0.0694	0.0215	0.8048	0.0007	0.0002	-0.0031	0.9583	0.9497	0.9517
2	0.0408	0.0474	0.0492	0.1148	0.1175	0.1219	0.0478	0.0271	0.8069	0.0003	-0.0009	-0.0046	0.9596	0.9500	0.9508
3	0.0413	0.0480	0.0550	0.1151	0.1182	0.1229	0.0883	0.0074	0.7333	-0.0005	-0.0004	-0.0071	0.9591	0.9494	0.9451
4	0.0438	0.0487	0.0503	0.1209	0.1214	0.1260	0.0667	0.0527	0.8213	0.0013	0.0020	-0.0029	0.9566	0.9487	0.9497
5	0.0424	0.0471	0.0482	0.1170	0.1187	0.1220	0.0583	0.0186	0.6778	0.0009	0.0000	-0.0024	0.9581	0.9503	0.9519
6	0.0430	0.0487	0.0483	0.1200	0.1225	0.1249	0.0520	0.0309	0.6178	0.0017	-0.0008	-0.0019	0.9575	0.9487	0.9517
7	0.0423	0.0474	0.0477	0.1160	0.1161	0.1191	0.0751	0.0139	0.7720	-0.0004	-0.0024	-0.0049	0.9581	0.9501	0.9523
8	0.0414	0.0472	0.0493	0.1154	0.1191	0.1216	0.0864	0.0439	0.8289	-0.0006	-0.0017	-0.0042	0.9591	0.9502	0.9507
9	0.0413	0.0473	0.0498	0.1168	0.1198	0.1242	0.0982	0.0234	0.7385	-0.0002	0.0004	-0.0034	0.9592	0.9501	0.9502
10	0.0424	0.0484	0.0498	0.1166	0.1194	0.1222	0.0472	0.0120	0.8194	-0.0002	0.0004	-0.0059	0.9581	0.9489	0.9503
11	0.0413	0.0475	0.0485	0.1157	0.1167	0.1193	0.0730	0.0082	0.7815	-0.0001	0.0000	-0.0044	0.9592	0.9499	0.9516
12	0.0427	0.0482	0.0492	0.1165	0.1187	0.1219	0.0352	0.0273	0.7852	0.0011	0.0012	-0.0039	0.9577	0.9491	0.9508
13	0.0414	0.0475	0.0468	0.1142	0.1172	0.1183	0.0736	0.0175	0.8031	0.0017	0.0020	-0.0021	0.9590	0.9499	0.9533
14	0.0411	0.0473	0.0478	0.1144	0.1168	0.1199	0.0630	0.0049	0.6778	-0.0006	-0.0002	-0.0036	0.9593	0.9501	0.9523
15	0.0427	0.0481	0.0492	0.1165	0.1197	0.1214	0.1006	0.0281	0.7500	0.0029	0.0029	-0.0012	0.9578	0.9493	0.9508
16	0.0445	0.0496	0.0497	0.1211	0.1219	0.1262	0.0656	0.0611	0.7336	-0.0003	-0.0010	-0.0055	0.9560	0.9477	0.9504
17	0.0416	0.0482	0.0487	0.1165	0.1195	0.1215	0.0899	0.0223	0.5713	0.0002	0.0004	-0.0046	0.9589	0.9491	0.9513
18	0.0415	0.0483	0.0487	0.1183	0.1216	0.1245	0.0659	0.0285	0.6854	0.0041	0.0035	0.0003	0.9590	0.9491	0.9514
19	0.0425	0.0480	0.0485	0.1170	0.1190	0.1207	0.0743	0.0353	0.7942	-0.0002	-0.0005	-0.0046	0.9579	0.9493	0.9515
20	0.0377	0.0451	0.0518	0.1157	0.1190	0.1235	0.0960	0.0073	0.7875	0.0019	0.0003	-0.0017	0.9627	0.9525	0.9483
21	0.0432	0.0479	0.0493	0.1176	0.1183	0.1223	0.0918	0.0071	0.7776	0.0061	0.0058	0.0017	0.9572	0.9495	0.9507
22	0.0412	0.0478	0.0490	0.1150	0.1196	0.1219	0.0892	0.0167	0.7525	-0.0006	-0.0001	-0.0047	0.9593	0.9496	0.9511
23	0.0421	0.0482	0.0478	0.1161	0.1182	0.1205	0.0834	0.0095	0.7868	0.0005	0.0009	-0.0033	0.9583	0.9492	0.9523
24	0.0426	0.0491	0.0482	0.1169	0.1193	0.1210	0.0645	0.0051	0.7230	0.0005	-0.0008	-0.0029	0.9578	0.9483	0.9519
25	0.0432	0.0482	0.0499	0.1189	0.1203	0.1237	0.0969	0.0182	0.6322	0.0004	-0.0024	-0.0028	0.9572	0.9492	0.9501
26	0.0420	0.0471	0.0476	0.1154	0.1158	0.1193	0.0876	0.0095	0.6946	0.0010	-0.0002	-0.0030	0.9585	0.9503	0.9525
27	0.0424	0.0483	0.0497	0.1159	0.1172	0.1213	0.0946	0.0081	0.7696	0.0001	0.0004	-0.0039	0.9580	0.9491	0.9504
28	0.0429	0.0492	0.0508	0.1193	0.1219	0.1252	0.0819	0.0486	0.3267	0.0002	-0.0020	-0.0034	0.9575	0.9481	0.9493
29	0.0410	0.0470	0.0498	0.1147	0.1167	0.1203	0.0617	0.0240	0.8536	-0.0004	-0.0014	-0.0067	0.9594	0.9504	0.9503
30	0.0440	0.0480	0.0502	0.1201	0.1201	0.1248	0.0753	0.0463	0.7466	0.0004	-0.0010	-0.0015	0.9564	0.9494	0.9498
31	0.0426	0.0470	0.0500	0.1161	0.1180	0.1229	0.0679	0.0269	0.7078	0.0024	0.0007	-0.0013	0.9579	0.9504	0.9500
32	0.0421	0.0472	0.0472	0.1156	0.1161	0.1196	0.0845	0.0121	0.7869	0.0014	0.0016	-0.0019	0.9583	0.9502	0.9528
33	0.0427	0.0484	0.0488	0.1163	0.1179	0.1213	0.0827	0.0062	0.8270	0.0004	-0.0011	-0.0040	0.9577	0.9490	0.9512
34	0.0430	0.0478	0.0477	0.1174	0.1187	0.1208	0.0395	0.0247	0.7838	0.0003	0.0001	-0.0046	0.9574	0.9496	0.9524
35	0.0432	0.0487	0.0491	0.1213	0.1225	0.1256	0.0841	0.0186	0.8092	0.0040	0.0034	0.0000	0.9572	0.9487	0.9509
36	0.0436	0.0483	0.0490	0.1197	0.1204	0.1237	0.0489	0.0439	0.7216	0.0014	0.0010	-0.0018	0.9568	0.9491	0.9510
37	0.0423	0.0482	0.0492	0.1168	0.1194	0.1225	0.0611	0.0267	0.7423	-0.0003	-0.0010	-0.0034	0.9581	0.9492	0.9509
38	0.0436	0.0487	0.0510	0.1193	0.1206	0.1256	0.0620	0.0507	0.7238	0.0036	0.0028	-0.0006	0.9568	0.9487	0.9490
39	0.0426	0.0481	0.0478	0.1152	0.1171	0.1187	0.0928	0.0068	0.8464	0.0001	0.0005	-0.0028	0.9579	0.9493	0.9523
40	0.0412	0.0476	0.0486	0.1137	0.1165	0.1192	0.0623	0.0165	0.7840	0.0002	-0.0003	-0.0035	0.9593	0.9498	0.9514
41	0.0433	0.0486	0.0491	0.1186	0.1205	0.1234	0.0448	0.0373	0.8600	0.0020	0.0011	-0.0033	0.9571	0.9488	0.9510
42	0.0437	0.0475	0.0478	0.1182	0.1178	0.1207	0.0922	0.0206	0.8116	-0.0062	-0.0075	-0.0100	0.9567	0.9499	0.9522
43	0.0416	0.0477	0.0516	0.1146	0.1170	0.1212	0.0924	0.0031	0.7941	-0.0002	0.0010	-0.0045	0.9588	0.9497	0.9484
44	0.0431	0.0477	0.0514	0.1194	0.1199	0.1252	0.0405	0.0162	0.8295	0.0029	0.0025	-0.0011	0.9573	0.9497	0.9486
45	0.0426	0.0485	0.0483	0.1186	0.1191	0.1224	0.0634	0.0236	0.8224	-0.0015	-0.0008	-0.0069	0.9579	0.9489	0.9518
46	0.0435	0.0484	0.0475	0.1184	0.1201	0.1207	0.0906	0.0206	0.7508	-0.0002	-0.0004	-0.0036	0.9570	0.9489	0.9525
47	0.0426	0.0490	0.0483	0.1182	0.1201	0.1228	0.0761	0.0414	0.6969	0.0015	0.0005	-0.0027	0.9578	0.9484	0.9518
48	0.0443	0.0489	0.0493	0.1214	0.1212	0.1262	0.0638	0.0550	0.7366	-0.0003	-0.0012	-0.0034	0.9562	0.9485	0.9508
49	0.0439	0.0478	0.0507	0.1188	0.1180	0.1230	0.0665	0.0398	0.8116	0.0001	-0.0010	-0.0020	0.9566	0.9496	0.9494
50	0.0440	0.0484	0.0488	0.1215	0.1212	0.1253	0.0527	0.0662	0.7541	0.0042	0.0031	-0.0007	0.9565	0.9490	0.9513

A.22 EXAMPLE OF PROGRAMMING CODE ANN MODEL – MATLAB

```
% Author: Camilla Lie
% Date: 24.05.2019


    %$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
    %$$$$$$$$$$$$$$$$$$$$$$$$$$$$ ANN-MODEL $$$$$$$$$$$$$$$$$$$$$$$$$$
    %$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Tables

```
T_error = [];

T_pred_all = [];

T_pred_test = [];

T_pred_train = [];

T_pred_val = [];

T_train = [];

T_wb = [];

T_state = [];
```

Load all data for TRAINING of model

```
% Training dataset
inv1t = readtable('train_inv1am.txt');
inv3t = readtable('train_inv3am.txt');
inv4t = readtable('train_inv4am.txt');

% Validation dataset
inv1v = readtable('val_inv1am.txt');
inv3v = readtable('val_inv3am.txt');
inv4v = readtable('val_inv4am.txt');

% Test dataset
inv1test = readtable('test_inv1am.txt');
inv3test = readtable('test_inv3am.txt');
inv4test = readtable('test_inv4am.txt');

% Standardized training dataset
inv1t_n = readtable('train_inv1_normam.txt');
inv3t_n = readtable('train_inv3_normam.txt');
inv4t_n = readtable('train_inv4_normam.txt');
```

```
% Standardized validation dataset
inv1v_n = readtable('val_inv1_normam.txt');
inv3v_n = readtable('val_inv3_normam.txt');
inv4v_n = readtable('val_inv4_normam.txt');

% Standardized test dataset
inv1test_n = readtable('test_inv1_normam.txt');
inv3test_n = readtable('test_inv3_normam.txt');
inv4test_n = readtable('test_inv4_normam.txt');
```

Load data for TESTING of model

```
% All data
inv1 = readtable('inv1am.txt');
inv3 = readtable('inv3am.txt');
inv4 = readtable('inv4am.txt');

% Standardized all data
inv1n = readtable('inv1_normam.txt');
inv3n = readtable('inv3_normam.txt');
inv4n = readtable('inv4_normam.txt');

% Input selection for model TRAINING
input_test1n = [inv1n.GHIavg,inv1n.VS,inv1n.airtemp,inv1n.AM2]';    % AM = scaled, AM2
= theta<=89 degrees
input_test3n = [inv3n.GHIavg,inv3n.VS,inv3n.airtemp,inv3n.AM2]';    % AM = scaled, AM2
= theta<=89 degrees
input_test4n = [inv4n.GHIavg,inv4n.VS,inv4n.airtemp,inv4n.AM2]';    % AM = scaled, AM2
= theta<=89 degrees

% Output for model TRAINING
target_test1n = [inv1n.PACinv1]';
target_test3n = [inv3n.PACinv3]';
target_test4n = [inv4n.PACinv4]';

all_test_norm1 = [input_test1n' target_test1n'];


% Input selection for model TESTING
input_testing1n = [inv1test_n.GHIavg,inv1test_n.VS,inv1test_n.airtemp,inv1test_n.AM2]';
input_testing3n = [inv3test_n.GHIavg,inv3test_n.VS,inv3test_n.airtemp,inv3test_n.AM2]';
input_testing4n = [inv4test_n.GHIavg,inv4test_n.VS,inv4test_n.airtemp,inv4test_n.AM2]';

% Target for model TESTING
target_testing1n = [inv1test_n.PACinv1]';
target_testing3n = [inv3test_n.PACinv3]';
target_testing4n = [inv4test_n.PACinv4]';
```

Standardized data

```
inv1a = table2array(inv1(:,2:end));

inv3a = table2array(inv3(:,2:end));

inv4a = table2array(inv4(:,2:end));




% Standard deviation
STD1 = nanstd(inv1a(:,2));
STD3 = nanstd(inv3a(:,2));
STD4 = nanstd(inv4a(:,2));

% Mean
mean1 = nanmean(inv1a(:,2));
mean3 = nanmean(inv3a(:,2));
mean4 = nanmean(inv4a(:,2));

% Standardized
STD1inam = nanstd(inv1a(:,[1,3,4,6]));
mean1inam = nanmean(inv1a(:,[1,3,4,6]));
```

For model purposes

```
% Input variabels (lines shuffled) sorted for training, validation and test
input_train1 = [inv1t_n.GHIavg,inv1t_n.VS,inv1t_n.airtemp,inv1t_n.AM2;...
    inv1v_n.GHIavg,inv1v_n.VS,inv1v_n.airtemp,inv1v_n.AM2;...
    inv1test_n.GHIavg,inv1test_n.VS,inv1test_n.airtemp,inv1test_n.AM2]';
input_train3 = [inv3t_n.GHIavg,inv3t_n.VS,inv3t_n.airtemp,inv3t_n.AM2;...
    inv3v_n.GHIavg,inv3v_n.VS,inv3v_n.airtemp,inv3v_n.AM2;...
    inv3test_n.GHIavg,inv3test_n.VS,inv3test_n.airtemp,inv3test_n.AM2]';
input_train4 = [inv4t_n.GHIavg,inv4t_n.VS,inv4t_n.airtemp,inv4t_n.AM2;...
    inv4v_n.GHIavg,inv4v_n.VS,inv4v_n.airtemp,inv4v_n.AM2;...
    inv4test_n.GHIavg,inv4test_n.VS,inv4test_n.airtemp,inv4test_n.AM2]';

% Target variabels (lines shuffled) sorted for training, validation and test
target_train1 = [inv1t_n.PACinv1;inv1v_n.PACinv1;inv1test_n.PACinv1]';
target_train3 = [inv3t_n.PACinv3;inv3v_n.PACinv3;inv3test_n.PACinv3]';
target_train4 = [inv4t_n.PACinv4;inv4v_n.PACinv4;inv4test_n.PACinv4]';

all_train = [input_train1' target_train1'];
```

Hold-Out Set

```
m1 = readtable('marapr17inv1am.txt');               % Data for March-April 2017
```

```
marapr1 = [m1.GHIavg,m1.VS,m1.airtemp,m1.AM2]';     % Input variables
marapr1n = (marapr1'-mean1inam)./STD1inam;          % Normalize input variables
maraprtarg1 = m1.PACinv1;                           % Output (target)
maraprt1n = (maraprtarg1'-mean1)./STD1;             % Normalize output
```

MEPS forecasted weather

Load either 'twodatforecast' or 'onedayforecast'

```
meps1 = readtable('twodayforecast.txt');               % Forecasted data
MEPS = [m1.GHIavg,m1.VS,m1.airtemp,m1.AM2]';     % Input variables
MEPSn = (MEPS'-mean1inam)./STD1inam;          % Normalize input variables
MEPStarg = meps1.PACinv1;                           % Output (target)
MEPStn = (MEPStarg'-mean1)./STD1;             % Normalize output
```

SPECIFY NETWORK

```
  for i = 1:50                                      % Run network several times
       clearvars net tr
[trainInd,valInd,testInd] = divideind(length(all_train),1:height(inv1t),...
    height(inv1t)+1:height(inv1t)+1+height(inv1v),height(inv1t)+1+height...
    (inv1v)+1:length(all_train));

net = feedforwardnet([20 2],'trainlm');             % Network configuration
net.divideFcn = 'divideind';                        % Data-division
net.divideParam.trainInd = trainInd;                % Training set
net.divideParam.valInd = valInd;                    % Validation set
net.divideParam.testInd = testInd;                  % Test set
net.trainParam.max_fail = 1000;                     % Validation check
net.trainParam.epochs = 1000;                       % Recommended by MATLAB
net.trainParam.min_grad = 1e-1000;                  % Set low to avoid early stopping
net.trainParam.goal = 1e-7;                         % Set low to avoid early stopping
net.trainParam.mu_max = 1e300;                      % Set high to avoid early stopping
net.layers{2}.transferFcn = 'tansig';               % Transfer function hidden layer
net.layers{3}.transferFcn = 'tansig';               % Transfer function output layer
net.performFcn = 'mse';                             % Mean Squared Error
```

TRAIN NETWORK

```
tic;

[net,tr] = train(net,input_train1,target_train1);   % Train network with input/output
ElapsedTime = toc;
```

ANN (feedforward) Model

x and t can be changed to hold-out set or MEPS data

```
x = input_train1;                                   % Input for model evaluation
t = target_train1;                                  % Corresponding target
Predict_for = net(x);                               % Predict using input variables
e1 = gsubtract(t,Predict_for);                      % Error predicted-target
performance = perform(net,t,Predict_for);           % Model performance
```

Model Evaluation

```
% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
testTarg = t(testInd);
valTarg = t(valInd);
trainTarg = t(trainInd);

    % ERRORS
e = gsubtract(t(trainInd),Predict_for(trainInd));
ev = gsubtract(t(valInd),Predict_for(valInd));
et = gsubtract(t(testInd),Predict_for(testInd));

    % MEAN SQUARE ERROR
MSE = immse(t(trainInd),Predict_for(trainInd));
MSEv = immse(t(valInd),Predict_for(valInd));
MSEt = immse(t(testInd),Predict_for(testInd));

    % MEAN ABSOLUTE ERROR
MAE = mae(e);
MAEv = mae(ev);
MAEt = mae(et);

    % MEAN ABSOLUTE RELATIVE ERROR
MARE = mean(abs(e./t(trainInd)));
MAREv = mean(abs(ev./t(valInd)));
MAREt = mean(abs(et./t(testInd)));

    % MEAN BIAS ERROR
MBE = mean(e);
MBEv = mean(ev);
MBEt = mean(et);

    % Rsquared
Rsq_s = 1 - sum((t(trainInd) - Predict_for(trainInd)).^2)/sum(...
    (t(trainInd) - mean(t(trainInd))).^2);
Rsqv_s = 1 - sum((t(valInd) - Predict_for(valInd)).^2)/sum(...
    (t(valInd) - mean(t(valInd))).^2);
Rsqt_s = 1 - sum((t(testInd) - Predict_for(testInd)).^2)/sum(...
    (t(testInd) - mean(t(testInd))).^2);
```

De-standardization

```
% Test set
test_pred = (Predict_for(testInd').*STD1+mean1)';
test_targ = (t(testInd).*STD1+mean1)';

% Validation set
val_pred = (Predict_for(valInd').*STD1+mean1)';
val_targ = (t(valInd).*STD1+mean1)';

% Training set
train_pred = (Predict_for(trainInd').*STD1+mean1)';
train_targ = (t(trainInd).*STD1+mean1)';
```

Network Specifications

```
weights = getwb(net);

ElapsedTime = array2table(ElapsedTime);



j = ones(length(TT),1).*i;

tes = ones(length(testInd),1).*i;

tra = ones(length(trainInd),1).*i;

val = ones(length(valInd),1).*i;

wei = ones(length(weights),1)*i;

o = (1:length(tr.epoch))';



en = {tr.trainFcn};

to = {tr.divideFcn};

tre = {tr.best_perf};

fire = {tr.best_vperf};

fem = {tr.best_tperf};

seks = {tr.best_epoch};
```

```
sju = {tr.time(end)};
```

For Tables

```
T_error = [T_error; MSE MSEv MSEt MAE MAEv MAEt MARE MAREv MAREt MBE MBEv...
    MBEt Rsq_s Rsqv_s Rsqt_s i];
T_pred_all = [T_pred_all; yt' tt' YT TT j];
T_pred_test = [T_pred_test; testPred' testTarg' test_pred test_targ tes];
T_pred_train = [T_pred_train; trainPred' trainTarg' train_pred...
    train_targ  tra];
T_pred_val = [T_pred_val; valPred' valTarg' val_pred val_targ  val];
T_train = [T_train; seks sju i];
T_wb = [T_wb; weights wei];
T_state = [T_state;tr.perf' tr.vperf' tr.tperf' o];
  end
```

Array2table

```
T_error = array2table(T_error);

T_error.Properties.VariableNames ={'MSEtrain','MSEval','MSEtest', 'MAEtrain',
'MAEval','MAEtest', 'MAREtrain','MAREval','MAREtest',
'MBEtrain','MBEval','MBEtest','Rsqtrain_2', 'Rsqval_2', 'Rsqtest_2' 'i'};
T_pred_all = array2table(T_pred_all);
T_pred_all.Properties.VariableNames = {'Predicted_norm', 'Target_norm', 'Predicted',
'Target', 'j'};
T_pred_test = array2table(T_pred_test);
T_pred_test.Properties.VariableNames = {'Predicted_norm', 'Target_norm', 'Predicted',
'Target', 'l'};
T_pred_train = array2table(T_pred_train);
T_pred_train.Properties.VariableNames = {'Predicted_norm', 'Target_norm', 'Predicted',
'Target', 'm'};
T_pred_val = array2table(T_pred_val);
T_pred.val.Properties.VariableNames = {'Predicted_norm', 'Target_norm', 'Predicted',
'Target', 'n'};
T_train = array2table(T_train);
T_train.Properties.VariableNames = {'best_epoch','TimeElapsed','i'};
T_wb = array2table(T_wb);
T_wb.Properties.VariableNames = {'Weights', 'k'};
T_state = array2table(T_state);
T_state.Properties.VariableNames = {'Perf', 'vPerf', 'tPerf','epoch'};
```

Plotting for model performance observation

```
timetest = inv1.time;
```

```
time_findtest = datevec(timetest);

time_test = time_findtest(:,1:3);

time_test = datetime(time_test);

dato = '06-Jun-2015';
time_test = datetime(time_test,'InputFormat','dd/MMM/yyyy','Format','MMM-yyyy');
start_test = find(time_test== dato,1,'first');
stop_test = find(time_test==  dato,1,'last');

dato1 = '05-Feb-2016';
time_test1 = datetime(time_test,'InputFormat','dd/MMM/yyyy','Format','MMM-yyyy');
start_test1 = find(time_test1== dato1,1,'first');
stop_test1 = find(time_test1==  dato1,1,'last');
```

PLOT for model observations

```
figure

plot(timetest(start_test:stop_test),(TT(start_test:stop_test)),'b','LineWidth',1.2)
hold on
plot(timetest(start_test:stop_test),(YT(start_test:stop_test)),'r','LineWidth',1.2)
grid;
legend('Target Output Power','Predicted Output Power','Location','Northwest')
ylabel('W')
title('GHI,WS,airtemp, AM2. LM')
set(gca,'FontSize',18);

figure
plot(timetest(start_test1:stop_test1),(TT(start_test1:stop_test1)),'b','LineWidth',1.2)
hold on
plot(timetest(start_test1:stop_test1),(YT(start_test1:stop_test1)),'r','LineWidth',1.2)
grid;
legend('Target Output Power','Predicted Output Power','Location','Northwest')
ylabel('W')
title('GHI,WS,airtemp, AM2. LM')
set(gca,'FontSize',18);
```

Write Tables

writetable(T_error,'error_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt'); writetable(T_pred_all,
'pred_all_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');
writetable(T_pred_test,'pred_test_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');
writetable(T_pred_train,'pred_train_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');
writetable(T_pred_val,'pred_val_20_2LM1000f_ghi_vs_t_AM_inv1_50.txt');
writetable(T_train,'trainfcn_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');
writetable(T_wb,'wb_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');
writetable(T_state,'state_20_2LM1000f_ghi_vs_t_AM2_inv1_50.txt');