NTNU

Norwegian University of
Science and Technology

An MINLP Approach Integrating Predictive Method for Optimal Load Sharing in Generating Machines

Byung Kyu Jung

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Tor Arne Johansen, ITK
Co-supervisor: David Anisi, ABB

Norwegian University of Science and Technology
Department of Engineering Cybernetics

PREFACE

The work presented in this master thesis was conducted in the Department of Engineering Cybernetics of the Norwegian University of Science and Technology (NTNU) as a part of a Masters of Science Engineering degree in Cybernetics under the supervision of Professor Tor Arne Johansen. This thesis was completed in the cooperation with ABB under the supervision of Dr. David Anisi, a principal R&D engineer at ABB industrial automation Oil, Gas, and Chemicals (IAOG). This work was conducted in parallel with system modeling [2] and [3].

In order to run the simulation using the algorithm developed in this work, data from a customer of ABB was required. However, due to the limited access to the ABB database, the simulation was delayed approximately by 1 month. This resulted losing the valuable time to improve the quality of this master thesis. Moreover, the computer provided by ABB had to be shared with a group of members from Oslo Metropolitan University writing their bachelor thesis for ABB. It required more waiting time than expected and was an obstacle for performing the necessary simulations.

I would like to express appreciation to my supervisors for the encouraging discussions and the helpful advice. I gratefully appreciate the time spent on the meetings for the assistance for this work through the semester. I am also grateful for the cooperation of Elene Marie Espeland for discussions and sharing necessary knowledge. I would also like to extend thanks to ABB for providing all the necessary data for this work. A special thanks to the R&D engineers from ABB sitting at K4 (4^{th} floor) for creating a great working environment, and for the support and giving me valuable inputs.

This work was made primarily for readers with the engineering background and was carried out during Spring 2018.

Trondheim, 2018-06-04

Byung Kyu Jung

SUMMARY

Gas turbines and diesel engines have generally been used in the Oil & Gas industry. As the amount of energy required for the operation is large, the efficiency of power plant operation has been subject to many works conducted recently. Moreover, the interest of integrating alternative sources of energy is increasing as the alternative energy sources do not cause any undesirable consequences to the environment. This paper proposes a method for operating the power and energy plant in an optimal manner. The main focus is achieving cost efficient operation of the plant. As the power plant on a petroleum site will require large volumes of fuel, reducing a small fraction of the fuel usage will have a huge impact on the environment in terms of reducing the greenhouse gas emissions as well as the total operating cost. The current practice of using the generating machines is based on the principle of "equal share". The main idea of "equal share" is to share the total load equally between the generating machines. However, this practice is not optimal considering the efficiency of the fuel usage. To perform load sharing in a more cost optimal manner, energy management system algorithm based on mixed integer non-linear programming is proposed. The future power demand is predicted using a machine learning method. The predicted power demands are given as a set of the parameters that are used by the energy management system algorithm. The optimal load distribution is scheduled based on the predicted future power demand, in a predictive control manner. An important performance parameter in order to obtain the optimal load sharing will be Brake Specific Fuel Consumption. The optimal load computation system proposed in this work is integrating gas turbines, grid electricity, and energy storage system. The possibility of integrating other energy sources, e.g., windmills and solar panels is of future consideration. Energy management system algorithm is implemented using the proposed model to perform simulation and the result will be compared and discussed. The findings from the simulations yield total cost saving of 2-2.48 %.

CONTENTS

ACRONYMS

AC Alternating Current

ANN Artificial Neural Network

BNB Branch and Bound

BSFC Brake Specific Fuel Consumption

DED Dynamic Economic Dispatch

EA Evolutionary Algorithm

ED Economic Dispatch

EMS Energy Management System

ESN Echo State Network

ESS Energy Storage System

FV Fitness value

GT Gas Turbines

HPS Hybrid Power System

ILP Integer Linear Programming

LSTM Long Short-Term Memory

LP Linear Programming

MFE Maximum Forecast Error

MPC Model Predictive Control

MSE Mean Squared Error

MILP Mixed Integer Linear Programming

MINLP Mixed Integer Non-Linear Programming

OCDD Optimal Control Dynamic Dispatch

PSO Particle Swarm Optimization

RNN Recurrent Neural Network

SFOC Specific Fuel Oil Consumption

SOS2 Specially Ordered Sets of type 2

STHTC Short Term Hydro Thermal Coordination

UC Unit Commitment

CHAPTER 1
INTRODUCTION

1.1 Background and motivation

The operations in the petroleum industry such as exploration, extraction, refining and transporting require a high amount of energy. The current practice of generating energy for the operation is based on the burning of fossil fuels. In another word, the power generating machines in the petroleum industry are mainly fueled by gas and diesel. The process of producing energy based on fossil fuels leads to greenhouse gas emissions, which is believed to be the main cause of the current global warming. Hence, it is worth looking into the ways of reducing fuel consumption.

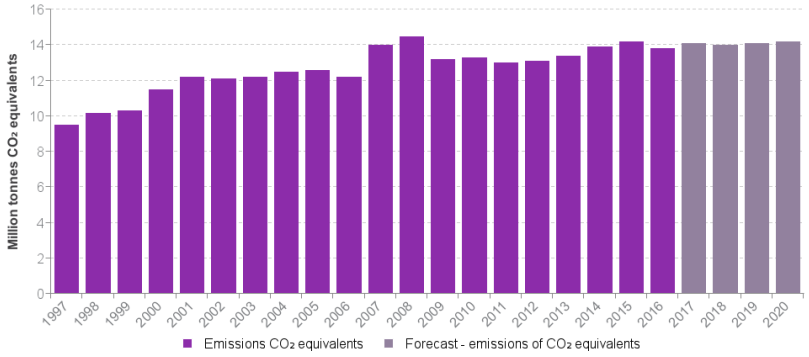


Figure 1.1.1: Greenhouse gas emissions from the petroleum sector in Norway [4]

.

The interest of saving not only the operational cost but reducing greenhouse gas emission is increasing. As the importance of minimizing emission of greenhouse gases increases to prevent the global climate changes. Figure 1.1.1 from [4] shows greenhouse gas emissions from the petroleum sectors in Norway. The numbers exceed multiple millions of tonnes. Therefore, reducing a small fraction of the total emission will have a huge impact on the environment.

The potential of cost-saving and reduction of greenhouse gas emission is significant by operating a petroleum site in a fuel/cost/energy optimal manner. The current practice of using the generating machines is based on the principle "equal share". The main idea of "equal share" is to share the total load equally between the generating machines. However, this practice is not optimal considering the efficiency of the fuel usage. The idea of optimal operation is based on exploiting the instantaneous energy efficiency of generating units. The optimal load sharing among the diesel/gas turbines can be achieved by managing load distribution using optimization algorithms with, e.g., the total fuel consumption as the objective to be minimized. As the total fuel usage is minimized, the emission of greenhouse gases will be reduced. Minimized fuel usage is also correlated with the cost-efficient operation.

This thesis is motivated by an interest of cost efficient operation of the Oil & Gas industry. Experimental studies will be performed using data from power generating units and simulation will be performed by employing machine learning approaches such as Artificial Neural Network (ANN) for forecasting the future power demand and power/energy management applications such as Mixed-Integer Linear Programming (MILP) and Mixed-Integer Non-Linear Programming (MINLP) for optimal load sharing. Moreover, to achieve realistic operational scenarios of the power generating units at a site in the process industry, different operational constraints will be added to the model. It is well known that Energy Storage System (ESS) will help increasing robustness of the system. If the grid or power generating units trip, ESS can kick in immediately and work as an alternative power source. In addition to obtaining robustness of the system, it will be experimented how ESS can help facilitating optimal power generation in terms of optimal load sharing of power generating units.

1.2 Goal and method

The goal of this thesis is to investigate the improvement potential of fuel/cost/energy savings using optimization methods. An important performance parameter is Brake Specific Fuel Consumption (BSFC) that describes

how much fuel is consumed generating one energy unit (MWh). BSFC will vary significantly between different diesel/gas turbines. BSFC curves using curve-fitting method as well as the methods presented in [2] will be used for this work. Offline BSFC estimation is covered in [2]. The further improvements embedding online methods updating BSFC is covered in [3]. The main objective of this thesis is designing and integrating machine learning approaches with energy management system (EMS) for optimal load share scheduling purpose. The detailed procedure of finding the BSFC parameters will not be covered in this work.

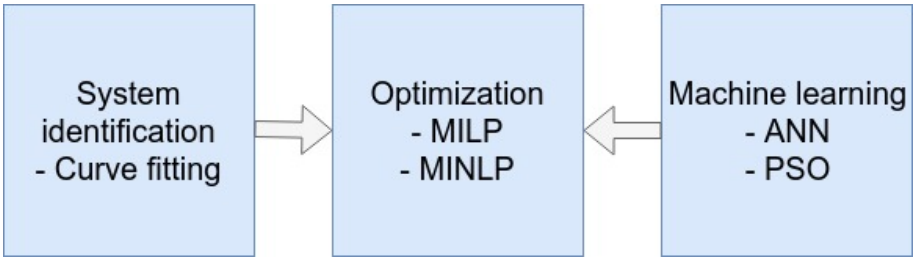


Figure 1.2.1: Sub-problems of the work

Figure 1.2.1 shows the sub-problems to be solved in this work. The main focus of this work will be employing a machine learning method for predictive planning and optimization problem modeling based on the studies about diesel/gas turbines, electrical grid and ESS. BSFC graphs will be estimated using curve-fitting method. The optimization model will be used to perform experiments that include soundness test of the algorithm when run in realistic industrial problems. The result of the simulations using the developed algorithm based on real site data will be discussed.

Literature studies about how machine learning and optimization approaches can be exploited in the Oil & Gas industry will be necessary and previous case and feasibility studies at ABB will also be used to ensure the non-overlapping outcome of the project. Different types ANN models and the solvers of optimization needs to be studied to find the most suitable one that can be adopted to the problem introduced.

The results of this study are based on 3-month data recorded from a site. The sampling rate of the data is 10 seconds. However, the EMS will only be used to find the optimal set-point with 100 seconds of the interval, which needs to be reached by a low-level power control system [5] that considers the dynamic of the power generating units.

1.3 Outline of report

The first part of this thesis will cover the relevant background information as well as some of the fundamental theoretical field necessary for understanding the outcome (Chapter 2). Chapter 3 will give a brief overview of the previous studies related to the optimization and machine learning. The machine learning approach for power demand forecast (time-series forecast) for predictive planning will be covered in Chapter 4. The optimization problem model will be presented in Chapter 5, which is divided into different parts with the detailed explanation about each part of the model.

The non-commercial tools adequate for solving optimization problem will be presented in Chapter 6. Some high-level explanation about implementations using different tools will be given. Moreover, Chapter 6 presents the overall structure of the algorithm developed in this work.

The result will be presented in Chapter 7. Different scenarios will be covered for the comparison and discussion that will show how the different choices can impact the result. Chapter 8 provides the conclusion and the potential cost saving will be stated. Finally, suggestions about the further improvements of this work will be discussed in Chapter 9.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Butterworth filter

Data pre-processing is considered as one of the most important phases of a machine learning task. If there is much irrelevant and redundant information present from noisy and unreliable data, then obtaining the reliable result becomes significantly more difficult. One of the most common data pre-processing methods is filtering noise from a raw data obtained directly from the real world. For example, in a time series sensor data, high-frequency part of the data can be considered as noise and in order to increase the performance of a machine learning task such as forecasting future sensor values, an adequate low-pass filter can be used to process the raw data in order to filter out the high-frequency signals.

Butterworth filter is one of the most commonly used low-pass filters because it is designed to have a frequency response as flat as possible in the passband without ripples. It is also referred to as a maximally flat magnitude filter. The first order version of this filter rolls-off down to zero after the cut-off frequency at the rate of 20 dB per decade. As the order of the filter increases, the roll-off rate increases. For example, 2^{nd} order filter rolls-off at a rate of 40 dB per decade and 3^{rd} order filter at a rate of 80 dB per decade and so forth.

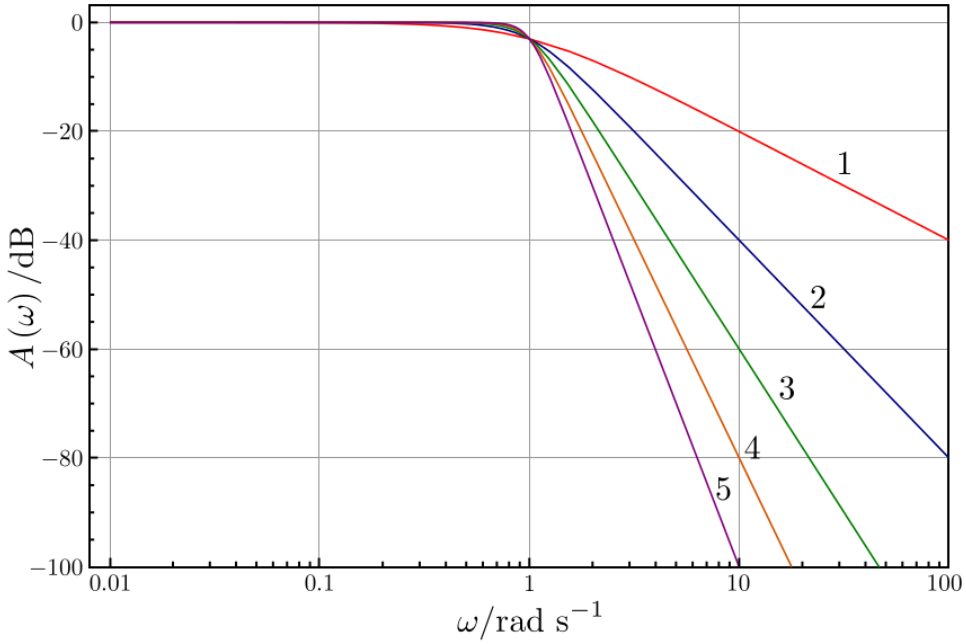


Figure 2.1.1: The gain of low-pass Butterworth filter with increasing order

As shown in Figure 2.1.1 the passband of the filter has a flat frequency response without ripples. This property allows the high-frequency noise signals to be filtered completely without having unwanted side-effects. Ripples can amplify the amplitude of signals at a certain frequency and can lead to different types of undesirable behaviors. E.g., if the ripples are considerably big and a sound wave is passed through the filter, the unwanted sound might be included in the filtered sound wave.

2.2 Artificial neural network

Artificial neural network is designed inspired by the biological neural networks and solve problems in the same way that a human brain would. ANN is designed based on connected units called artificial neurons.

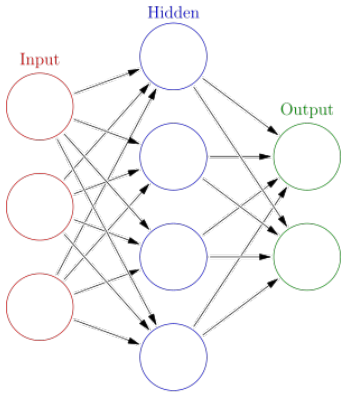


Figure 2.2.1: An example of artificial neural network

Figure 2.2.1 shows an ANN and interconnection between the group of nodes. The connection of the neurons makes it possible to transmit signals to each other and receive signals from each other. The output signal of each neuron is computed by a non-linear function using the sum of its inputs. Moreover, each connection has a weight, which can be adjusted depending on the learning task. ANN does not require any task-specific configuration and is able to learn tasks by considering training samples. E.g., ANN can be trained to identify a specific figure from an image. The input signals are processed through the hidden layers to the output layer. The output signals of the output layer contain the useful information. E.g., in an image recognition task, the output signal will tell if an image contain the target figure or not.

2.3 Recurrent neural network

Recurrent Neural Network (RNN) is a class of ANN. RNN has shown promising results in different tasks such as time series prediction, classification, and natural language processing. The common characteristic of these problems is that historical data plays a crucial role. For example, humans do not start their thinking from scratch every time they want to start talking. In a conversation, context needs to be understood to share ideas and their thinking to each other. RNN is able to address the issue of understanding the context by using feedback input to individual neurons.

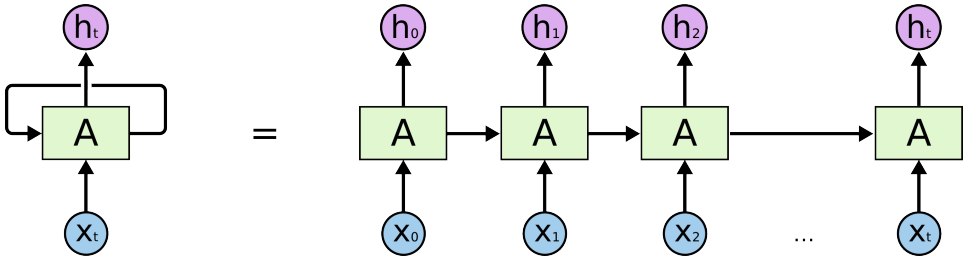


Figure 2.3.1: A simple feedback connection of RNN

As shown in Figure 2.3.1, input x_t results output h_t that is also dependent on the previous result h_{t-1}. The loopback connection can be unrolled to a normal neural network with t connections. The neurons are sequentially connected to each other. The final output h_t will depend on all the sequential data.

There are some drawbacks using RNN. If RNN is used for generating data that requires long-term dependency, which means that t is a big number, several problems can be introduced. In the training process of such network, the gradient that is used for back-propagation will suffer from the vanishing gradient problem. In another word, h_t will not be able to generate the result based on the long-term history. E.g., if the outcome of an event at the end of a movie is needed to be predicted, and the most important information was given at the beginning of the movie, RNN is required to "clearly" remember the accident from several hours ago, which will be hard. It is also expensive to maintain all the state. Let's again think about the movie example, it requires a lot of memory for the computer to remember all the states and the events from the beginning of the movie to the end of the movie.

2.3.1 Long short-term memory

Long Short-Term Memory (LSTM) is an RNN architecture that solves the problems introduced to the standard RNN network, e.g., vanishing gradient. LSTM is designed to be able to have long-term memory. All the figures used in this section are from [23]

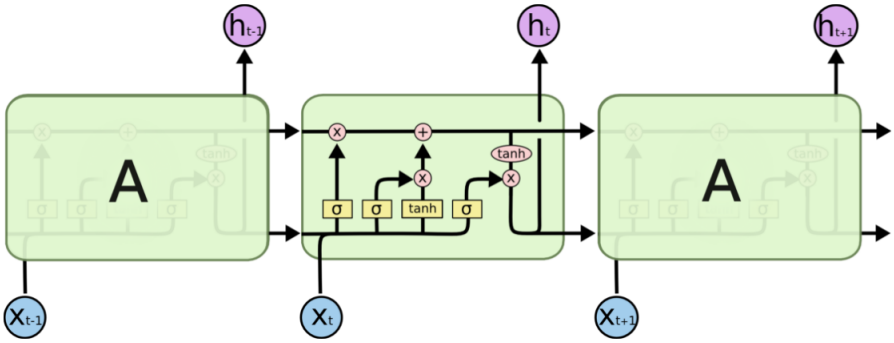


Figure 2.3.2: LSTM network [23]

An LSTM network contains multiple LSTM units. Each LSTM unit has the ability to remember states for long or short time periods. An LSTM unit contains several units. As illustrated in Figure 2.3.2 there are several activation units and math operation units such as point-wise multiplication and addition. There are two values that flow between LSTM units; state, and output. The output h is concatenated with the input x and the cell state C is the information flowing through the "top" line within an LSTM unit. Each line from the diagram carries an entire vector. There are three different gates presented within an LSTM unit: input, forget and output gates.

The forget gate layer controls what information to be thrown away from the cell state passed from the previous LSTM unit.

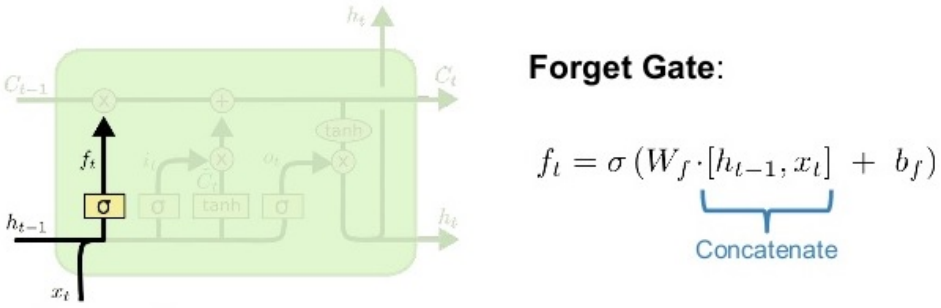


Figure 2.3.3: LSTM forget gate [23]

The output h_{t-1} from the previous LSTM unit is concatenated with the input x_t and is sent through a sigmoid function, which will output a number between 0 and 1. Cell state C_{t-1} is element wisely multiplied with the output from this gate. Depending on the output of the gate, cell state can be kept or removed (1 or 0).

The input gate layer will decide how the cell state will be updated.

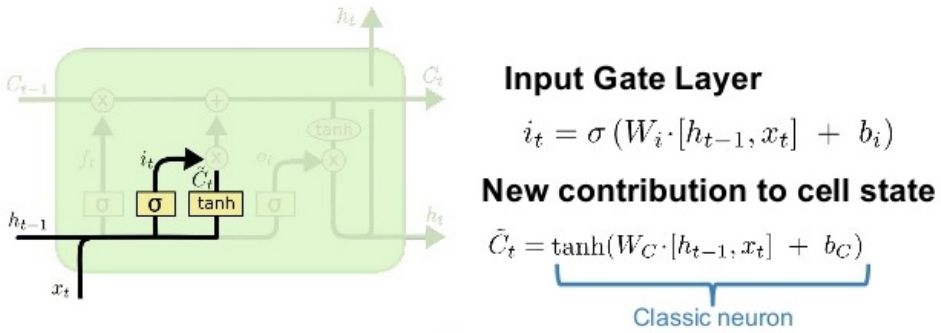


Figure 2.3.4: LSTM input gate [23]

h_{t-1} and x_t will be sent through the sigmoid and hyperbolic tangent activation functions. The outputs will be combined together by point-wise multiplication to create an update to the state. In the end, the update cell state will be combined with the old cell state;

$$\mathbf{C}_t = \mathbf{f}_t * \mathbf{C}_{t-1} + \mathbf{i}_t * \tilde{\mathbf{C}}_t.$$

The output gate layer is the last gate, which will decide the output of LSTM units. The output will be based on the new cell state C_t, output from the previous LSTM unit h_{t-1} and the input x_t.

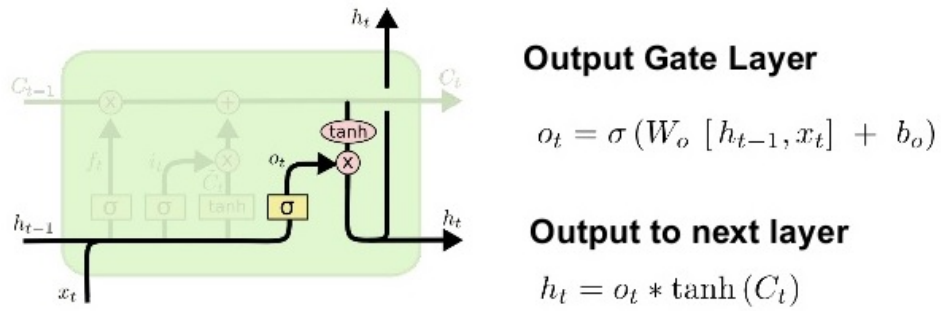


Figure 2.3.5: LSTM output gate [23]

The vector $[\mathbf{h}_{t-1} \; \mathbf{x}_t]$ will be sent through the last sigmoid function and combined with the output of the hyperbolic tangent function, which takes in the new cell state C_t.

2.3.2 Echo state network

Echo State Network (ESN) [32] provides an architecture and supervised learning principle for RNN. ESN is mainly divided into three different parts. Input layer, reservoir layer, and output layer. A significant difference of ESN from another type of NN is that the weights of the input layer and the reservoir layer are generated randomly at the initial state of the algorithm and is kept unchanged. The only trainable weights are the output weights. Given that the ESN will have K input units, N internal network units and L output units the dimension of the weight matrices will be as follows: $\boldsymbol{W}_{in} \in \mathbb{R}^{N \times K}$, $\boldsymbol{W}_{reservoir} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{W}_{out} \in \mathbb{R}^{L \times (1+K+N)}$. The recurrent property of RNNs is provided by the reservoir layer. This layer will also provide the non-linear dynamic between the input signals and the output signals.

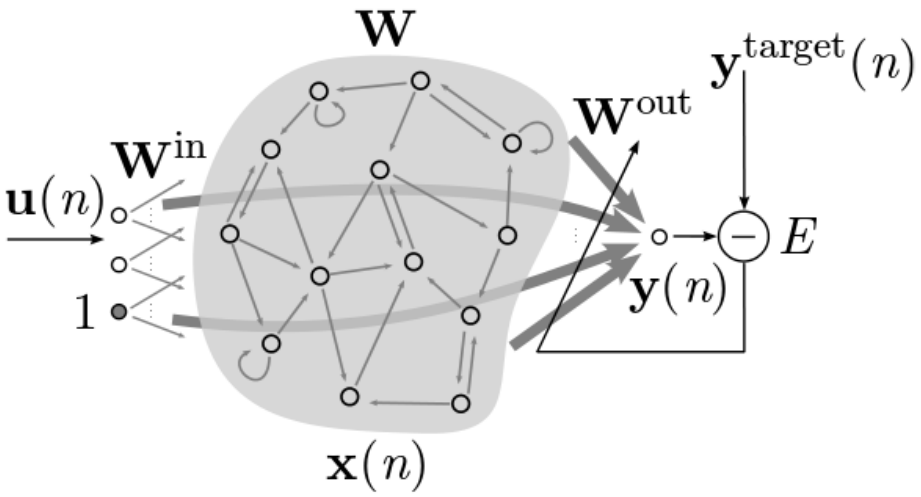


Figure 2.3.6: Illustration of the standard Echo State Network architecture

As it can be seen from Figure 2.3.6 the K dimensional input unit $\boldsymbol{u}(n)$ is provided to the ESN model. The input $\boldsymbol{u}(n)$ is then multiplied with randomly generated $\boldsymbol{W}_{in}$. The reservoir units also called state units $\boldsymbol{x}(n)$ are computed based on the output from the input layer, randomly generated $\boldsymbol{W}_{reservoir}$ and the previous state units $\boldsymbol{x}(n-1)$. $\boldsymbol{b}$ represents a bias term.

$$\boldsymbol{x}(n) = \boldsymbol{f}_{state}\left[\boldsymbol{W}_{in}\boldsymbol{u}(n) + \boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1) + \boldsymbol{b}\right]. \tag{2.3.1}$$

The activation function $\boldsymbol{f}_{state} \in \mathbb{R}^N$ is task-dependent. One of the most commonly used activation function is sigmoid. A sigmoid function is a mathematical function having a characteristic "S"-shaped curve. An example of

a sigmoid function is the logistic function,

$$f = \frac{1}{1 + e^{-x}}.$$

The size of state unit $\boldsymbol{x}$ can be chosen freely and should be bigger than the number of input units. The reservoir serves as a memory and provides temporal context. In general, the bigger the reservoir, the better the obtainable performance. However, this will also increase computational complexity and memory. Moreover, a higher risk for over-fitting. The size of the reservoir should not be bigger than the size of the samples that can be used for training.

A disadvantage of using the standard sigmoid network for updating state units is that they do not contain a time constant. Their dynamic cannot be slowed down like the dynamics of the differential equations. In order to add this property to the model, the state update term can be modified to use leaky integrator [32]. This modifies (2.3.1) into

$$\boldsymbol{x}(n) = (\boldsymbol{I} - \boldsymbol{A})\boldsymbol{x}(n-1) + \boldsymbol{f}_{state}\left[\boldsymbol{W}_{in}\boldsymbol{u}(n) + \boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1) + \boldsymbol{b}\right], \tag{2.3.2}$$

where $\boldsymbol{A}$ is the leaky matrix. The value of the leaky coefficient can be chosen between 0 and 1. A value close to 1 means that the state units do not retain any information about its previous state. A value close to 0 means that the state units retain most of its previous state. Practically, a number close to 0 means that the dynamic is slowed down significantly compared to the standard sigmoid network. As mentioned, the feedback term $\boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1)$ provides the recurrent property. $\boldsymbol{b}$ represents a bias term.

The output is computed according to

$$\boldsymbol{y}(n) = \boldsymbol{f}_{out}(\boldsymbol{W}_{out}\left[\boldsymbol{1}; \boldsymbol{u}(n); \boldsymbol{x}(n)\right]) \tag{2.3.3}$$

The output activation function $\boldsymbol{f}_{out} \in \mathbb{R}^L$ is task-dependent. E.g., for the time series forecast, this function is typically a linear function. The bias term is included by adding an extra vector containing ones. As illustrated in Figure 2.3.6, the output $\boldsymbol{y}(n)$ is compared with the ground truth vector $\boldsymbol{y}_{target}(n)$ to compute the output weights. To recall, in ESN algorithm the input weights and the reservoir weights are kept unchanged.

The connectivity between state units within the reservoir layer needs to be chosen when designing an ESN model. The less connectivity leads to high degree of sparsity of the state units. High sparsity means that the computational effort decreases, however, this can also lead to dynamical decoupling. In general, the sparsity of the reservoir does not affect the performance much.

Choosing the appropriate reservoir weight matrix, $\boldsymbol{W}_{reservoir}$, is crucial considering echo state property. In order to secure echo state property, the spectral radius of the reservoir connection needs to be less than one in most of the cases. The spectral radius of a matrix is the eigenvalue with the biggest amplitude. This means that the input history and the state will have an impact on the future state, and gradually tend to disappear. The reservoir layer will gradually forget the information from the initial condition and reach an asymptotically stable state. However, the spectral radius being less than one does not provide a sufficient condition for the echo state, as this spectral radius being bigger than one is only a sufficient condition for the non-existence of echo states [32]. One additional parameter that needs to be checked is the largest singular value. This value needs to be less than one in order to prove the existence of echo states [32]. However, in most of the practical cases, it is enough to check the spectral radius in order to secure the echo state property. Scaling of the reservoir weights can be used to keep the spectral radius less than one.

The only trainable output weights can be found using regression method that minimizes the squared error between $\boldsymbol{y}_{target}(n)$ and $\boldsymbol{y}(n)$.

$$Minimize \left[\boldsymbol{y}_{target}(n) - \boldsymbol{y}(n)\right], \quad \forall n \in [1, ..., N].$$

Given that the number of training samples N exceeds the summation of the number of inputs and the reservoir size, the system becomes over-determined. An over-determined system is almost always inconsistent, in another word, it has no solution. Hence, a solution of $\boldsymbol{W}_{out}$ needs to be approximated using the least-square method, which yields

$$\underset{\boldsymbol{W}_{out}}{\text{minimize}} \quad \left|\left|\boldsymbol{y}_{target} - \boldsymbol{W}_{out}\boldsymbol{X}\right|\right|,$$

where

$$\boldsymbol{X} = [\boldsymbol{1};\ \ \boldsymbol{u};\ \ \boldsymbol{x}].$$

The solution of which can be presented by the following equation [40],

$$\boldsymbol{W}_{out} = \boldsymbol{y}_{target}\boldsymbol{X}^{\top}\left(\boldsymbol{X}\boldsymbol{X}^{\top}\right)^{-1}. \tag{2.3.4}$$

In [41], an alternative way of finding $\boldsymbol{W}_{out}$ is presented, so called Tikhonov regularization method. Using a regularization coefficient, the solution from (2.3.4) is transformed to

$$\boldsymbol{W}_{out} = \boldsymbol{y}_{target}\boldsymbol{X}^{\top}\left(\boldsymbol{X}\boldsymbol{X}^{\top} + \beta\boldsymbol{I}\right)^{-1}. \tag{2.3.5}$$

Equation (2.3.5) is a solution of

$$\underset{\boldsymbol{W}_{out}}{\text{minimize}} \quad \left|\left|\boldsymbol{W}_{out}\boldsymbol{X} - \boldsymbol{y}_{target}\right|\right| + \beta\left|\left|\boldsymbol{W}_{out}\right|\right|$$

$\beta \boldsymbol{I}$ is a regularization term, which limits the $\boldsymbol{W}_{out}$ to become large. Large weights indicate that the system amplifies tiny differences among the dimensions of $\boldsymbol{x}$, and can be very sensitive to deviations from the exact conditions in which the network has been trained. The regularization coefficient β, penalizes $\boldsymbol{W}_{out}$ and prevents $\boldsymbol{W}_{out}$ from becoming too large. This regularization term is used to prevent over-fitting as well as instability.

A simple and intuitive method of finding $\boldsymbol{W}_{out}$ is using pseudo-inverse, which yields

$$\boldsymbol{W}_{out} = \boldsymbol{y}_{target}\boldsymbol{X}^{+}.$$

$\boldsymbol{X}^{+}$ is a pseudo-inverse matrix of $\boldsymbol{X}$. The direct pseudo-inverse computing method is highly stable, however, it requires large memory for the computation.

If a recurrence dynamic between the reservoir and the trained readout is required, (2.3.1) can be modified to include a feedback term

$$\boldsymbol{x}(n) = \boldsymbol{f}\left[\boldsymbol{W}_{in}\boldsymbol{u}(n) + \boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1) + \boldsymbol{W}_{back}\boldsymbol{y}(n-1)\right] \tag{2.3.6}$$

The feedback term enables reservoirs to achieve universal computational capabilities [37].

Instead of feeding back the readouts $\boldsymbol{y}(n)$, the desired output $\boldsymbol{y}_{target}(n)$ can be used

$$\boldsymbol{x}(n) = \boldsymbol{f}\left[\boldsymbol{W}_{in}\boldsymbol{u}(n) + \boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1) + \boldsymbol{W}_{back}\boldsymbol{y}_{target}(n-1)\right]$$

This method is called teacher forcing, which speeds up convergence or even may be necessary to achieve convergence at all [28].

2.4 Particle swarm optimization

Particle Swarm Optimization (PSO) [9] is a computational method used for optimization trying to improve the candidate solution with regard to quality. The quality is represented by a numerical value, which is comparable. Denoted as the fitness value (FV). The algorithm explores the search space $\mathbb{R}^n$ of a given problem using spread particles. The dimension of the search space is defined by the number of free parameters that can be tuned by the algorithm. The search space is therefore often called as the parameter space and feasible region in optimization. Each unique combination of the parameters gives a FV and this value is compared to find the optimal parameter set that minimizes or maximizes the FV. The function that is used to generate the FV given a candidate solution is different depending on the problems to be solved. However, this function is often not directly known.

Therefore, a black box approach that takes a candidate solution and gives a FV can be used.

PSO algorithm starts by placing particles within a random $\mathbb{R}^n$ high dimensional search space. The particles are being moved iteratively and the candidate solution of each particle is evaluated using either an objective function or a black box approach. Each particle maintains its position, velocity, evaluated FV and the candidate solution. The evaluated FV of the particles is compared at each iteration and the particle with the best FV is referred as the individual best FV and is remembered by the algorithm. The individual best FV is then compared with the global best FV, which refers to the best FV found during the execution of the algorithm. The position of the particles are updated simply by using the following equation:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (\hat{x}_i(t) - x_i(t)) + c_2 \cdot r_2 \cdot (g(t) - x_i(t)). \quad (2.4.1)$$

The particle number is denoted by i. The parameters w, c_1 and c_2 are user defined coefficients. They affects how the particles are updated. The parameters r_1 and r_2 are a value between 0 and 1, randomly chosen by the algorithm. These random parameters prevent the algorithm from converging to the particle with the first found individual best FV. $\hat{x}$ represents the particle position with the individual best FV and g is the particle position with the global best FV. The terms $w \cdot v_i(t)$, $c_1 \cdot r_1 \cdot (\hat{x}_i(t) - x_i(t))$ and $c_2 \cdot r_2 \cdot (g(t) - x_i(t))$ are called inertia component, cognitive component and social component, respectively [9].

The position is updated according to the equation:

$$x_i(t+1) = x_i + v_i(t+1) \quad (2.4.2)$$

In order to prevent the particles from moving outside of the search space $\mathcal{X} \in \mathbb{R}^n$, a technique called velocity clamping is used [10].

2.5 MILP

Mixed-integer linear programming is an extension of Linear Programming (LP) [47]. LP is a method used for the optimization of a linear objective function subject to linear constraints on the free variables. The constraints may include both equalities and inequalities. Linear programs are usually stated and analyzed in the following standard form

$$\begin{aligned} \underset{\mathbf{x}}{\text{maximize}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}. \end{aligned}$$

In MILP some of the variables are restricted to be integers. MILP is introduced with the following standard form; given the constraints and the objective function, the integer variable $\mathbf{x}_I$ and the continuous variable $\mathbf{x}_C$ should be found, which maximizes the objective function

$$\begin{aligned} \underset{\mathbf{x}}{\text{maximize}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} = (\mathbf{x}_C, \mathbf{x}_I) \\ & \mathbf{x}_I \in \mathbb{Z}^n \\ & \mathbf{x}_C \in \mathbb{R}^n. \end{aligned} \tag{2.5.1}$$

Where $\mathbb{Z}^n$ denotes set of integers. The decision variables are a mix of continuous variables and integer variables. MILP is a non-convex and NP-hard problem. There exists, however, algorithms that can be used to solve MILP such as *Branch and Bound* and *Cutting plane*. Today's MILP/MIQP solvers typically use a combination of those two.

The integer term will help to describe the physical process in a convenient way. For example, in the manufacturing and production industry, the number of units produced needs to be an integer. The integer variables are also often restricted to be binary. The binary variables can be used to describe yes/no or on/off situations. For examples, a binary variable can be used to define if a machine is on or off.

2.5.1 Branch and bound

The Branch and Bound (BnB) is one of the most widely used methods to solve MILP. Conceptually, BnB is a divide and conquer strategy for MILP. The problem P is divided into sub-problems and the solutions of the sub-problems are used to obtain the optimal solution of P. The division is performed iteratively, such that the sub-problems are easier to solve. The sub-problems are discarded as long as it can be proved that it cannot produce the optimal solution.

Explicit enumeration is a method used to divide the main problem P into sub-problems by dividing the search space S into different regions, such that

$$\begin{aligned} S_0 &= \{x \in S : x_1 = 0\} \\ S_1 &= \{x \in S : x_1 = 1\} \\ S &= S_0 \cup S_1. \end{aligned}$$

S_0 and S_1 can be divided further into the smaller regions. However, it is not always feasible for practical problems because explicit enumeration will result in infinitely many sub-regions.

Implicit enumeration is a method decomposing the search-space S into K different subsets

$$S = S_1 \cup ... \cup S_k.$$

Given a subset k of the spaces, an upper bound and a lower bound for the solution is found. The subsets are discarded iteratively using the upper and the lower bounds of the other subsets. If the lower bound of the subset k is bigger than the upper bound of the subset *k+1*, the subset *k+1* is discarded, since it is found that the subset *k+1* does not contain the optimal solution. The subsets with in-feasible solutions are also discarded from the full search-space. This continues until the optimal solution is found.

Equation (2.5.2) shows an example of how the full search-space can be divided into two subsets.

$$\begin{aligned} S_0 &= \{x \in S : x_1 \leq 2\} \\ S_1 &= \{x \in S : x_1 \geq 3\} \\ S &= S_0 \cup S_1 \end{aligned} \tag{2.5.2}$$

2.5.2 Cutting plane

The cutting plane algorithm is the other widely used method for solving the MILP problems. The main idea of the algorithm is assuming the integer variables as continuous and adding valid inequalities until the integer solution is obtained. The feasible sets or the objective function will be refined iteratively by means of linear inequalities.

The feasible sets $X \in P \cap \mathbb{Z}_+^n$ can be updated to include new inequalities that cuts the feasible plane $P^t \in P \cap \{x : \pi_i^\top x \leq \pi_{i0}, i = 1, 2, ..., t\}$ and P^t will be a tighter formulation than the initial formulation P, resulting $X \in P^t \cap \mathbb{R}_+^n$.

The ILP problem

$$\max\{c^\top x : Ax \leq b, x \in \mathbb{Z}_+^n\} \tag{2.5.3}$$

can be reformulated as LP, introducing new additional valid inequalities

$$\max\{c^\top x : \hat{A}x \leq \hat{b}, x \in \mathbb{R}_+^n\}, \tag{2.5.4}$$

where $\hat{A}$ and $\hat{b}$ includes A and B from (2.5.3) and some additional constraints.

Notice that the solution of LP from (2.5.4) will be the solution of ILP from (2.5.3), if the additional constraints covers convex hull of (2.5.3). The main issue of the problem is choosing the useful inequalities. Finding the most useful inequalities requires convex hull analysis. However, there are efficient

ways of finding such inequalities and one of the most used methods for cutting is called Chvátal-Gomory Cutting [14].

2.6 MINLP

Mixed integer non-linear programming is an extension of MILP where some of the constraints or the objective function are non-linear. MINLP is an NP-hard problem and there is no algorithm that guarantees the problem to be solved in a polynomial time. Fortunately, there are some approximation techniques that can be used such as McCormick Envelops and piece-wise linearization for simplification of MINLP.

2.6.1 McCormick envelops

McCormick Envelops is a convex relaxation method used to convert bilinear terms to linear terms so that the non-linear problem only containing bilinear terms becomes a linear problem.

Considering the general class of MINLP where the non-linear term is bilinear

$$w = xy$$

with the constraints

$$x^L \leq x \leq x^U$$
$$y^L \leq y \leq y^U$$

Introducing the new constraints

$$\begin{aligned} x &- x^L \geq 0 \\ x^U &- x \geq 0 \\ y &- y^L \geq 0 \\ y^U &- y \geq 0 \end{aligned}$$

gives

$$\begin{aligned} &(x - x^L)(y - y^L) \geq 0 \\ &\Rightarrow \\ &w \geq xy^L + yx^L - x^L y^L. \end{aligned}$$

All the terms become linear so that the MINLP problem becomes MILP.

2.6.2 Piecewise linear approximation

MINLP is an NP-hard problem and solving MINLP can be very hard, which can takes hours, days or forever to solve. However, MINLP can be transformed to MILP by using piece-wise linear approximation techniques. Specially Ordered Sets of type 2 (SOS2) is one of the most used piece-wise linearization methods.

SOS2 picewise-linear model is using an ordered set of variables, $\{\lambda_0, ..., \lambda_n\}$, which represent special ordered set of variables type 2. The characteristic is that at most two variables are bigger than zero. In addition, if two variables are bigger than zero, then they are consecutive in the ordered set: λ_i and λ_{i+1}. The model based on SOS2 variables require several additional constraints. Given $\{(x_0, f(x_0)), (x_1, f(x_1)), ..., (x_n, f(x_n))\}$ and $\{\lambda_i\}$ is SOS2.

$$\begin{aligned} x &= \sum_{i=0}^{n} \lambda_i x_i \\ y &= \sum_{i=0}^{n} \lambda_i f(x_i) \\ 1 &= \sum_{i=0}^{n} \lambda_i \\ \lambda_i &\geq 0, \;\; i = 0, ..., n \end{aligned} \tag{2.6.1}$$

The non-linear $f(x)$ can be swapped with the linear y presented in (2.6.1).

Some of the alternative models that can be used for the linearizations are:

1. Convex combination
2. Incremental
3. Disaggregated Convex Combination
4. Logarithmic Convex Combination
5. Disaggregated Logarithmic Convex Combination

2.7 Big-M

The Big-M method can be used to relax constraints and can be combined with binary variables to formulate the conditional constraints by adding a large positive penalty constant M to the inequality constraints. For instance, an inequality can be ensured if a binary variable takes on a value, and the

inequality can be relaxed if this binary variable takes on its opposite value. For example, given two conditional constraints,

$$\begin{aligned} if \quad u = 1 \quad &\Rightarrow \quad \boldsymbol{a}_1^\top \boldsymbol{x} \leq b_1 \\ if \quad u = 0 \quad &\Rightarrow \quad \boldsymbol{a}_2^\top \boldsymbol{x} \leq b_2. \end{aligned}$$

Using a big positive constant M, the conditional constraints can be transformed to

$$\begin{aligned} \boldsymbol{a}_1^\top \boldsymbol{x} &\leq b_1 + M \cdot (1 - u) \\ \boldsymbol{a}_2^\top \boldsymbol{x} &\leq b_2 + M \cdot u \\ u &\in \{0, 1\} \end{aligned}$$

which ensures $\boldsymbol{a}_1^\top \boldsymbol{x} \leq b_1$ if and only if u = 1 and $\boldsymbol{a}_2^\top \boldsymbol{x} \leq b_2$ if and only if u = 0.

M needs to be chosen carefully, such that it is "large enough". However, too large M can also affect the convergence rate of the algorithm to the optimal solution [46].

CHAPTER 3

LITERATURE REVIEW

3.1 ESN for wireless communication

In [29], ESN has been used for prediction of the Mackey-Glass system, which is a benchmark system for time series prediction. An ESN model with 1000 reservoir units and one output unit with output feedback to the reservoir has been designed. The designed ESN model is then trained using 3000-step teach sequence. The output connection weight w_{out} is computed using the last 2000 steps of the stored reservoir states. The first 1000 steps have been removed since these steps introduce the transient error. 84-step ahead prediction has been performed for testing. 100 independent trials have been used to find a normalized root mean square error and the result yields $10^{-4.2}$. Another ESN model has been designed for the equalization task for wireless communication. A noise possessed signal needs to be filtered using an equalizing filter when it is received. The intention of using a filter is to recover the original signal, which is converted into a symbol sequence. Performance of the ESN with 46 reservoir units has been compared with the standard equalization methods such as linear decision feedback equalizer, Volterra decision feedback equalizer, and bilinear decision feedback equalizer. The prediction result yields an improvement of two magnitudes for high signal-to-noise ratios.

3.2 ESN with EA for identification of highly non-linear motion

In the work presented in [36], ESN has been combined with the Evolutionary Algorithm (EA). The ESN model has been used for estimation of the motion of an underwater robot, which is highly non-linear. The limit cycle test,

which turns the robot to the right until the yawing rate exceeds the limit velocity has been performed and the resulting data is used as the training set of ESN. The input of the ESN model is yawing rate and torque. The output is yawing acceleration. The coupling relationship between those variables is highly non-linear and hard to find an analytic model. EA is used to find the most appropriate global ESN parameters such as the number of reservoir units, global contraction and the average density of interconnection. The EA use MSE as the error measurement and the fitness criterion is defined as lowering the MSE error. After finding the best combination of the global parameters, EA has been applied again for adjusting the internal reservoir weights. Two different approaches have been tested for tuning the weights. One of them yields using EA on the entire reservoir weight matrix and the other method yields applying a mask matrix to keep all the zeros of the sparse reservoir weight matrix. In another word, only the non-zero elements of the reservoir weight matrix have been adjusted using EA. The most promising result has been obtained using the full mask version, which yields tuning the global ESN parameters and the full reservoir weight matrix using EA. The next best approach found is using EA on the masked sparse reservoir matrix. The error difference between the best and the next best approach is not that significant and those two approaches showed better performance compared to the brute force method and the method only using EA for tuning the global ESN parameters. Conclusively, the approaches including the EA outperformed the brute force method.

3.3 Electrical load forecasting using ESN with PSO optimization

ESN for electrical load forecasting and optimization using PSO is discussed in [43]. The global parameters of designing ESN such as reservoir size, connectivity of the reservoir units, spectral radius of the reservoir weight matrix and input scaling factor have been optimized using PSO. Prediction error of ESN has been used for the fitness measurement and the objective is to minimize this fitness value. The algorithm continues until the termination condition of the PSO algorithm is reached. After the termination of the PSO algorithm, the output of the PSO algorithm, which is the optimal combination of the global ESN parameters has been applied for making an ESN model. 24000 sets of data for electrical load has been used for training ESN and the result has been compared with Support Vector Machine, Back Propagation and ESN with manually tuned parameters. The comparison result yields better performance using ESN-based approaches compared to Support Vector Machine and Back Propagation. The lowest prediction error was achieved using the ESN model with PSO optimization.

3.4 Time series classification using ESN for speech recognition

Employing ESN for automatic speech recognition using isolated utterance of the English digits "zero" through "nine" has been discussed in [38]. 26 utterances of each digit from 4 male and 4 female speakers were collected. Frame-based speech features, using human factor cepstral coefficients were extracted from the utterances and used as the training set. The test set consisted of 26 utterances from the 8 different speakers. Each digit was trained using multi-filter readout, meaning that a portion of the speech signal for each digit was segmented, which reduces non-stationarity. Practically, for each digit, the input to the ESN classifier is a given size of the batch of the training samples and K number of W_{out} were trained by segmenting reservoir states into K equal-length sections along with the desired signals. The readout weight was trained using a dynamic programming similar to the Viterbi algorithm presented in [39]. A different number of readout filters and reservoir size was tested and the classification accuracy was compared with each other. Conclusively, the classification result obtained using a hidden Markov model was compared with the classification result for the ESN classifier.

3.5 Online readout weight update of ESN using an adaptive algorithm

The adaptive algorithm can be conjoined with a classic ESN to catch the non-stationary dynamic. Finding the optimal readout weight is a linear task, a standard recursive least square algorithm for Mean Squared Error (MSE) minimization can, therefore, be applied to online ESN estimation. In [31] the author is applying recursive least square algorithm together with ESN for prediction of a 10^{th}-order system. ESN is slightly modified by using a squared version of the input and the network state. Originally, the readout weight is computed based on a vector; $[u(n), x_1(n), ..., x_N(n)]$, but in order to increase the modeling power of ESN, the readout weight is computed using a augmented vector; $\big[u(n), x_1(n), ..., x_N(n), u^2(n), x_1^2(n), ..., x_N^2(n)\big]$. Hence, the length of the readout weight is increased from N + 1 to 2N + 2. In the case study conducted in [31], the squared version of the readout weight is updated with a forgetting factor = 0.95 and 202 network states were used. The goal was to estimate a 10^{th} order system with randomly varying parameters. The 10^{th} order system was assigned to new parameters every 2000 steps. A 10 000 sequence training data is used to test the performance of online ESN. The result yields lower NMSE using online ESN after convergence compared with offline trained ESN. The readout weight size of a standard offline training of ESN depends on the amplitude size of the

noise inserted into the network during training [32], similarly, the readout weight size using online ESN was affected by the size of the noise inserted to the network. The readout weight entered a region of numerical instability without noise insertion. Thus, the noise term is crucial here for numerical stability.

3.6 EMS based on MILP and logic algorithm

In [13], MILP-based and logic-based algorithms are applied to control power distribution in diesel-electric marine vessels such as ferry, platform supply vessel and seismic survey vessel. The objective regards fuel efficiency and maintenance of the gensets. MILP considers fixed speed genset, variable speed genset, and ESS. Leading to three different types of the configurations. The objective function includes power capability for the gensets, number of running hours as well as the number of starts/stops of the gensets. Service and maintenance of the diesel engines are required after 1000 running hours, by including the running hour term to the objective function, running hours of all the gensets can be synchronized, which can save maintenance cost and downtime. The decision variables are genset on/off decisions, which is binary, power loading of the variable speed genset and the power flow of ESS. The performance of MILP-based EMS and logic-based EMS are compared for each type of the vessel.

3.7 EMS based on ILP

In [15], The Specific Fuel Oil Consumption (SFOC) is estimated using a recursive estimation method. The objective regards fuel efficiency. The objective function includes the fuel consumption of each diesel generator. The decision variables of the objective function represent the discrete operating points of the diesel generators. Meaning that the power of each diesel generator is divided into a finite number of operating points. The decision variables are binary. Each generator is only allowed to operate at one specific operating point, and the binary variable representing this operating point is set to one, meanwhile, the rest are set to zero, in another word, the binary variables are Specially Ordered Sets of type 1. Leading to optimization problem model only consisting of the binary decision variables. The number of decision variables will increase or decrease depending on the number of operating points of the diesel generators. The performance of ILP-based EMS is tested by comparing the optimized fuel consumption with the original fuel consumption.

3.8 EMS based on MILP and prediction method

The MILP model presented in [16] is using power output of the diesel generators as the decision variables. Moreover, the on/off decision variables of the diesel generators are embedded into the model. The objective function regards the optimal load sharing in terms of the optimal fuel saving. The model is expanded further to include a certain window of time. The on/off decision variables in the model are transformed to the number of start-ups given a window of time, which are integers. However, embedding the time into the model introduces an additional requirement. The power demand for a given time t needs to be considered, which means that the power demand needs to be predicted for the certain window of time used for the model. Therefore, the algorithm introduced in this paper combines MILP with some prediction methods. The performance of such model depends on the accuracy of the prediction. Embedding the future decision variables in the model will potentially increase the fuel-saving since the model also considers the future operating points. Furthermore, energy storage technology is investigated to achieve higher efficiency. The idea is to control the total load using ESS in the most beneficial way with regard to engine efficiency.

3.9 MILP with renewable energy resources

The MILP model presented in [17] includes different renewable resources as well as gas turbines and fuel cells. The model is designed to find the optimal combination of distributed generation units. The objective regards annual total cost including investment cost and running cost. The integer decision variables represent the on-off status of the energy resources and the existence of energy storage devices. The continuous decision variables express the power flow from the system components. The model considers energy balance, supply-demand and performance characteristics of the system components such as photovoltaic system, wind turbine, and energy storage etc. The algorithm is simulated for every hour in a whole year. The output is the optimal combination of on-site generation and heat recovery, the capacity of each technology to be installed and an alternative power generation schedule of the system components. Since the model is used for scheduling, the algorithm needs to take account of variations of the load demand and fuel price over the simulation period. The assumption made for this work is that all the varying parameters are known with complete certainty, i.e., energy load demands and fuel prices for the duration of the test year are assumed to be given.

3.10 MILP for solving STHTC

The optimal Short Term Hydro-Thermal Coordination (STHTC) is discussed in [18]. In this work, MILP has been applied for solving STHTC problem. The non-linear behavior of the power generating units is transformed to piece-wise linear equations and used for designing a MILP model. The objective function regards cost-efficient hourly power generation schedule for a given time horizon. The objective function includes thermal cost, thermal unit start-up cost, and future cost. The future cost is related to the water volume stored in the reservoir and is estimated using stochastic dynamic programming recursion. The constraints of the MILP model consists of hydro constraints, thermal constraints, and system constraints. The hydro constraints related to water turbine regards water balance equation, irrigation, limits on storage, power production as a function of water discharge, constant and variable head operation and spill characteristics. The constant/variable operating head behaviors and the spill characteristic of the water turbines are piece-wisely linearized and embedded to the MILP model as a part of the constraints. The thermal constraints regard to power production of each thermal unit, minimum downtime, and uptime, ramping in power, the maximum number of start-ups, etc. The system constraints regard power demand, import/export of power and spinning reserve. The decision variables are power from the thermal units, thermal unit on-off binary variables, turbine water flow, which is coupled with the power produced by the water turbines and on-off binary variables of the hydro units.

3.11 MINLP for solving ED and UC combined problem

In [19], MINLP is applied for solving the Unit Commitment (UC) problem and the Economic Dispatch (ED) problem as a one combined problem. UC is the problem of determining the number of committed units in the system in advance. ED is the problem of scheduling the output power levels of the committed generating units in a power system to meet the demand at minimum cost. The MINLP model presented in this work includes a generator index and a time index. The objective function regards fuel consumption and start-up cost. The start-up cost considers cold start-ups and hot start-ups. This cost is modeled as an exponential function. However, this non-linear term is linearized by discretization. The fuel consumption is modeled as a quadratic function, which induces non-linearity. The constraints presented in this work is power generation, ramping rates, power balance, minimum up, down-times and spinning reserve. The decision variables are UC decision variables, generator state variables for minimum down-time and up-time, scheduled output power level and scheduled ramp up or down-level.

The generator state variables are used for designing discrete-time dynamic model with UC decision variables as an input control. The power level is considered as a state variable with ramp variables as an input control. The final model that solves the UC and ED combined problem is formulated as MINLP. Quadratic term of the fuel consumption makes the problem MINLP. This term is simplified using a piece-wise linear approximation, which transforms the problem to MILP. The results obtained by solving MINLP and MILP are compared.

3.12 MPC for solving UC problem for wind farms

In [20], a UC problem for the wind farm is introduced, which is solved using the Model Predictive Control (MPC) method. The UC problem is designed using an objective function that regards production costs, start-up costs, shut-down costs and maintaining costs. The constraints, which is included in the model is the maximum available power of the wind farm, the minimum wind farm capability, the coupling of the state and input variables. The state variables represent the on-off transition of each wind turbine, e.g., off $\rightarrow$ on, off $\rightarrow$ off, on $\rightarrow$ off and on $\rightarrow$ on. The input variables represent the on-off status of the wind turbines. The UC problem is then transformed into a standard MPC problem. ED problem is not included in the model and the power from the operating wind turbines are fixed to the maximum available power. The short-term available wind power is obtained from a wind power forecasting center.

3.13 MPC to overcome constraint violation and robustness under certain uncertainties

The work [21] presents two different formulations for solving an ED problem. Namely, the Optimal Control Dynamic Dispatch (OCDD) and the Dynamic Economic Dispatch (DED). OCDD formulation is using a simplified power dynamic model. The future power is recursively estimated based on the initial power and ramp rate. The power output works as the state variables. The optimization result will highly depend on the initial power. The ramp rate is controlled by using a separate constraint. DED does not include any power dynamic model, however, the ramp rate is controlled by using a ramp rate constraint on the power output between each time step. The power output is denoted as the states. The difference is that DED ignores the ramp limit between the initial power and the first optimized power and unlike OCDD, the optimization result does not depend on the initial power. The ramp rate violation will most likely occur for both of the formulations. In this paper, MPC is used to prevent ramp constraint violation. MPC increases robustness and prevents the ramp violation, however, the cost

determined by the optimization solution using MPC is proven to always be greater than or equal to that of the classical OCDD/DED. Using MPC is similar to employing extra constraints to the OCDD/DED models that secure feasibility of the solution. Increased number of constraints will lead to increased cost of optimization result compared to the original model. Convergence and feasibility of MPC are proven in this paper. The MPC algorithm is tested on a six-unit system and shows the convergence and robustness of the MPC solutions. The electricity demand is assumed to be periodic and therefore does not require any prediction method. This assumption comes from the fact that the demand is periodic due to cyclic consumption behavior and seasonal changes.

3.14 An MPC approach for solving ED involving intermittent energy resources

An MPC approach for solving the ED problem in electric power systems involving intermittent resources is presented in [22]. The MPC algorithm is designed to minimize both environmental cost and operating cost. The look-ahead optimal dispatch algorithm includes constraints such as power demand, minimum/maximum power of all the generating units and ramp rate of generators for dynamically scheduling all available resources. The time-varying parameters such as power demand and available power of intermittent energy generators are predicted using a dynamic model for a time horizon N. The dynamic of the varying parameters such as minimum/maximum power of intermittent resources and the total demand is included to the MPC algorithm as a part of constraints. This paper does not cover how these dynamic models are designed. The optimization algorithm is executed using a moving horizon N, which equals to the forecast horizon and the output yields the power output of all the available generators for the time horizon N. However, only the first control signal is implemented [49]. After retrieving a new measurement, the designed algorithm is solved again using this new measurement and the first control signal is implemented. The algorithm is simulated on a 12-bus system and about 7 % cost reduction is achieved.

CHAPTER 4

ESN AND POWER DEMAND FORECAST

4.1 Introduction

Figure 4.1.1 shows example pattern of required power to operate a site. The goal is to catch this pattern and forecast the required power or power demand and use the predicted values for implementing optimal predictive planning algorithm. By integrating the predictive method to the optimal planning algorithm, the power distribution scheduler will be able to catch the dynamic of the power system and find a reliable cost-efficient way to distribute the load throughout the power system.

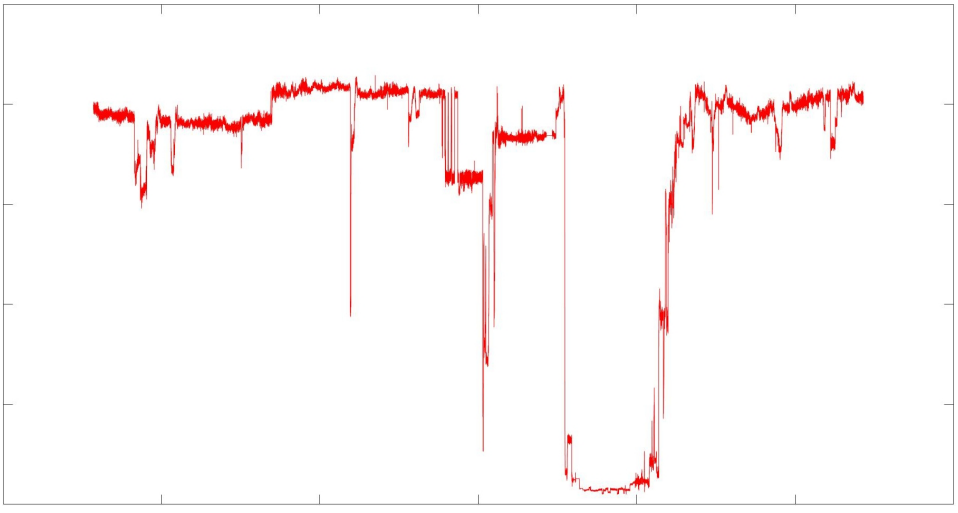


Figure 4.1.1: Required power to operate a site

In order to implement a predictive planning algorithm, forecast of the time-varying parameters such as power demand, the maximum power level of the gas turbines and the electricity price is required to use them as model constraints. The electricity price and the maximum power can be considered constant or known if the time of window to be included in the model is approximately chosen. However, the power demand constraint is fast changing and can't assumed to be fixed. Therefore, prediction of power demand using a machine learning method is considered. power demand changes over time, and is dependent on the multiple factors such as the grid loading, weather, tide, etc.. However, availability of the such tags is often limited and for generality of the algorithm, only historical data will be used to train a neural network model that will forecast the future power demand.

power demand oscillation includes load shifting in order to keep the stability of the grid frequency. The frequency of the grid is the nominal frequency of the oscillations of Alternating Current (AC) in an electric power grid transmitted from a power station to the end-user. Depending on the location, the frequency of the grid can vary. In Europe, this frequency is 50 Hz. If the consumption exceeds the generation, the frequency of the alternating current will fall to a value below 50 Hz. In this case, either more electricity needs to be produced by the producers or load reduction can be requested to the major energy users. If the generation of electricity exceeds the consumption, the frequency will rise to a value above 50 Hz. The grid operator then needs to ensure that the electricity producers reduce the generation of electricity.

ANN explain in Section 2.2 is chosen as the method to forecast power demand. ANN is a good candidate for the prediction task because of its ability to find implicit dependencies and relationship in data without having any prior knowledge about all the relevant influencing factors. It is also possible to adopt changed conditions without having any significant complications.

RNN is a class of ANN where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence data. Basic ideas of RNN is explained in Section 2.3.

ESN is a class of RNN. The basic idea of ESN is explained in Section 2.3.2. Shortly explained, the random reservoir layer is used to influence the input signals $\boldsymbol{u}(\mathrm{t})$ and creates a high-dimensional collection of non-linearly transformed versions $\boldsymbol{x}(\mathrm{t})$. Moreover, $\boldsymbol{x}(\mathrm{t})$ works as a memory of input signals. $\boldsymbol{x}(\mathrm{t})$ is then combined with readout weights to map the output signals with the desired signals. The reservoir states, being an input-driven dynamical system, should provide a rich and relevant enough signal space in $\boldsymbol{x}(\mathrm{t})$, such that the desired signals can be obtained by a linear combination from it.

ESN is proved to be a highly practical approach to RNN training. It is conceptually simple and computationally inexpensive, therefore reinvigorating interest among the researchers and yielded excellent performance. In [29], the performance of ESN has been tested for prediction of a chaotic system called the Mackey-Glass system and use of the equalization of a wireless communication channel. ESN has been used for prediction of 84-step ahead signal. A normalized root mean square error over 100 independent trials is found to be $10^{-4.2}$. For equalization of wireless communication task, ESN showed improvement of signal error rate by two orders of magnitude compared to standard techniques. For training the output weights, an online learning method has been applied. In [36], the motion of an underwater robot is estimated using ESN combined with an Evolutionary Algorithm (EA) to tune the global ESN parameters such as the number of reservoir units, global contraction and the average density of interconnection. Moreover, using the tuned number of reservoir units, the connectivity matrix has been adjusted using an EA. The work presented in [43] is applying PSO together with ESN for electrical load forecasting. The global parameters of ESN model such as reservoir size, connectivity, spectral radius and input scaling factor are tuned using PSO. The prediction error value is used as the FV for PSO. After finishing tuning the global parameters using PSO, an ESN model is made based on the result obtained using PSO. Error evaluation showed PSO-ESN improved the prediction accuracy significantly. Speech recognition using isolated utterances of the English digits "zero" through "nine" in a classification experiment was performed using ESN in [38] and the classification result was compared with a hidden Markov model. [31] describes how recursive least squares algorithm can be used together with ESN to provide online adaptation to the trainable readout weights. The readout weights were updated online by minimizing the error between the prediction and the desired outputs. The algorithm presented in [31] is used to track a 10^{th} order non-linear system.

The work presented in [29] shows the potential of using ESNs for tasks such as time-series prediction. However, the Mackey-Glass system has a clear pattern with a clear dynamic, which simplifies the task. As shown in Figure 4.7.1, power demand does not contain a clear dynamic nor the pattern. The performance proved in [29] is not expected to be achievable for power demand forecast. The motion of an underwater robot represents a highly non-linear dynamic and [36] shows the potential of using ESNs for capturing the highly non-linear dynamic. This is an essential property of the RNN that will be used in this work since the non-observable dynamic of power demand might consist of a highly non-linear dynamic. The work presented in [43] shows the possibility of integrating an optimization algorithm with ESNs to optimize the global parameters of ESNs. This will simplify the pa-

rameter tuning process to obtain the best forecast accuracy possible. Also, [31] shows how non-stationary dynamic can be captured by ESNs by integrating an adaptive method to update the readout weights. power demand might contain a non-stationary dynamic, hence adding an adaptive method for updating the readout weight can be considered.

In this work, the raw power demand data will be pre-processed using the Butterworth low-pass filter and standardized according to the formula (4.7.1). Moreover, ESN will be designed to forecast power demand and the global parameters of ESN will be optimized using PSO.

4.2 ESN vs. LSTM

There are advantages and disadvantages of ESNs over LSTMs. The most important advantage of using ESNs is the time it takes to train a model. ESNs are especially good at learning fast and online. ESNs can, therefore, be adapted to dynamics that change over time (online training). The fast learning property gives room to integrate ESNs with a genetic algorithm, which can be used to optimize the global parameters to design task-specific ESNs. However, ESNs will require a large amount of memory compared to LSTMs. LSTMs are smaller and more efficient once trained, although the training process will require significantly more time than ESNs.

It is hard to decide the most task suitable model when designing RNN using LSTMs, it requires experience in designing ANN and high degree of knowledge about the task related domain. Moverover, unlike ESNs, LSTMs cannot be integrated with PSO for the optimization of the model, since it does not allow fast and online learning.

For the regression, the reservoir layer of ESNs has more hidden units than inputs so it represents many random independent transformations of the input. The probability that this transformation will contain useful representations of the dynamics that will let the regression perform better than directly on the input itself is high. Hence, depending on the size of the reservoir, ESNs can be a suitable choice when coping with time-series data that contain multiple numbers of dynamics. LSTMs will not be able to catch different types of dynamics at the same time and will, therefore, run into trouble if time-series data contains multiple dynamics.

4.3 Global parameters of ESN

The behavior of ESN is defined by different global parameters such as the size of the reservoir, connectivity of the reservoir states, size of noise added

to the reservoir states and spectral radius of the reservoir weights. These parameters can be chosen differently for different learning tasks. Tuning these parameters require knowledge about how they affect the behavior of ESN.

4.3.1 Size of the reservoir

Reservoir acts as non-linear expansion and a memory of the input signals. Meaning that for highly complex learning tasks a big reservoir is required. In general, the bigger the reservoir, the better obtainable performance. However, without any generalization of training of ESN, the over-fitting problem can be introduced. Since over-fitting can be overcome as explained in Section 4.3.4, it is desired to have a reservoir size as big as possible. The only limitation is the amount of training data available and computational power. The reservoir size should be kept such that the total number of inputs and reservoirs is always less than the total number of training samples. Moreover, a big number of reservoir requires big memory, therefore computational trade-off needs to be considered choosing the reservoir size.

4.3.2 Connectivity of the reservoir

Connectivity of the reservoir can be controlled at the initialization step. Among the randomly generated reservoir weights, all the weight below a specific number can be set to zero. In this way, the sparsity of the reservoir connection can be obtained. The sparsity of the reservoir matrix can speed up the training process. Distribution of the non-zero elements can be controlled by using different types of distributions when generating random reservoir weights.

4.3.3 Spectral radius of the reservoir weights

Spectral radius of the reservoir weight is an important parameter considering the stability of ESN. This parameter has to be carefully chosen to secure that the echo state property is not violated [25]. Spectral radius of the reservoir matrix can be computed by finding the maximal absolute eigenvalue of the matrix. This eigenvalue describes the width of the distribution of the non-zero elements, which is governed by the connectivity parameter and the distribution method. In another word, the spectral radius can be easily controlled by using a scale factor. Firstly, the matrix can be divided by its spectral radius to obtain a matrix with a unit spectral radius. This matrix is then multiplied by a scale factor, which becomes the new spectral radius. This scale factor needs to be less than 1 to secure the echo state property in most of the cases. It is theoretically possible that the echo state property can be violated even though this scale factor is chosen to be less than 1. However, this is unlikely to happen in practice. Spectral radius

of the reservoir matrix needs to be selected to maximize the performance. Intuitively, this parameter determines how fast the influence of an input dies out in a reservoir with time. With small spectral radius, the influence of the historical inputs will die out faster compared to the reservoir weight matrix with greater spectral radius. This indicates that the spectral radius should be chosen greater in tasks requiring longer memory. The spectral radius also defines how stable the reservoir activations are. ESN becomes more stable with smaller spectral radius [26].

4.3.4 Noise on the reservoir neurons

Scaled white noise is used to increase stability and reduce the chance for over-fitting. This scaled white noise is added directly to the reservoir states from (2.3.1).

$$\boldsymbol{x}(n) = \boldsymbol{f}\left[\boldsymbol{W}_{in}\boldsymbol{u}(n) + \boldsymbol{W}_{reservoir}\boldsymbol{x}(n-1) + \boldsymbol{b}\right] + \boldsymbol{w}_{\boldsymbol{white}}$$

In [32] it is shown that the standard offline training of ESNs yields output weights whose absolute size depends on the noise inserted into the network during training, the larger the noise, the smaller the mean output weights. Extremely large readout weight values may be an indication of a very sensitive and unstable solution. So by increasing the amplitude of noise, ESN can become more robust for different types of feature inputs as well as reducing the chance for over-fitting. A readout training method referred as Tikhonov regularization [27] can be used to obtain a similar regularization effect.

4.3.5 Leaky parameter

Different kind of activation functions is applicable designing ESN, not only the standard sigmoid network. The echo state property can be maintained using any kind of dynamical system. A weakness of sigmoid network is that it does not have any time constant. In another word, the network using sigmoid will not be able to catch the dynamic that is very slow. For instance, it will be very hard for a standard ESN to track a very slow sine-waves using sigmoid. An alternative method is using a continuous time neuron model, leaky integrator neurons [30]. Leaky neurons can be described by a differential equation with a time constant [30].

$$\dot{\boldsymbol{x}} = \frac{1}{\tau}\left[-\boldsymbol{A}\boldsymbol{x} + \boldsymbol{f}(\boldsymbol{W}_{in}\boldsymbol{u} + \boldsymbol{W}_{reservoir}\boldsymbol{x})\right]. \tag{4.3.1}$$

The time constant τ can be chosen freely, and a large time constant should be used for dynamics that are slow and vice versa. $\boldsymbol{A}$ is a decay potential of the neuron. The larger $\boldsymbol{A}$, the faster the decay of the previous state. In another word, the input signals will have more influence on the reservoir neurons. The update scheme from (4.3.1) can be transformed into a discrete time domain [28]. The transformed equation is shown in (2.3.2).

4.3.6 Input weight scale factor

The input matrix connecting the input signals with reservoir layer can be scaled and is a global parameter of ESN. Less scaling value indicates that the input signals have less influence on the reservoir neurons. It is advised in [28] that this scaling factor should be big for non-linear tasks and for the linear tasks, this scaling factor should be small.

4.4 Optimizing the global parameters of ESN using PSO

The forecast performance of ESN depends highly on the global parameters presented in Section 4.3. Moreover, these global parameters are task-specific, so they need to be tuned differently depending on the learning task. Tuning the parameters optimally requires excellent knowledge about the learning task and experience in using ESN.

An alternative approach instead of tuning the global parameters manually is using an evolutionary search algorithm that does not require gradient based calculation. ESN provides a black box method, mapping the inputs to the outputs using a black box, it is therefore not possible to use gradient based optimization. This lack of principles of the design can be overcome by using PSO. PSO does not require the use of the gradient, in another word, the optimization problem does not need to be differentiable. Conceptually, PSO works in a way that a given number of candidate solutions are randomly generated. These candidate solutions are also referred to as particles. Those particles are moving around in the search-space according to a simple mathematical formula as explained in Section 2.4 either until the optimal solution is found or a defined number of iteration is reached. The movement of the particles is influenced by the best-positioned particles.

PSO is used in this work for finding the important parameters designing ESN. A benefit of using PSO is that the optimal combination of the parameters can be found without having deep knowledge about the learning task and being expert in using ESN. ESN does not require a large amount of time for the training process, which gives possibility for tuning parameters using a search algorithm based on the prediction result.

There are six global parameters that will be tuned using PSO. Size of the reservoir, connectivity of the reservoir, spectral radius of the reservoir weights, the amplitude of white noise to be added when updating the reservoir states, leaky parameter, and the input weight scale factor. For each particle that contains a candidate combination of these six parameters will

be used to design ESN and the prediction result (MSE) will be used as the FV and different combinations will be tested and compared using this FV. The algorithm will be terminated after a given number of iterations. A similar procedure using PSO together with ESN is explained in [43]. The total number of iterations and the number of particles to be used to search the optimal solution is depending on the number of the parameters to be optimized. As the number of the parameters to be optimized increases, the number of the particles to be used for search should also increase.

4.4.1 Optimal parameter tuning process

PSO algorithm is used to optimize the global parameters of ESN. The optimization process will contribute increasing the prediction accuracy as well as the robustness of the algorithm. The process is as follows:

1. A given number of particles are initialized and randomly distributed in $\mathcal{X} \in \mathbb{R}^6$. In this case, there are six parameters to be optimized, therefore the search space will be $\mathcal{X} \in \mathbb{R}^6$. Moreover, each particle's velocity is initialized randomly. The velocity of the particles will be used to find the optimal direction and speed of movement of the particles. The randomly generated position and velocity are constrained by upper and lower bounds of the parameters.

2. After the initialization step, each particle is tested using a fitness function. The FV in this work will be the MSE error of prediction. Each set of the parameters stored in a particle is used to train an ESN model and the MSE error of prediction is stored as the FV of the particle. The FV of the particles is compared with each other and the particle with the best FV survives and is stored. This particle is referred as the particle with the best individual FV.

3. This individual best FV is compared with the global best FV and the maximum/minimum (depending on the criteria) FV is stored as the new global best FV.

4. According to the formulae (2.4.1), particles's position and velocity is updated.

5. This continues for a given number of iterations or until the global best FV converges.

PSO can be used to optimize the global parameters of ESN since the training process of an ESN does not require a large amount of time. Given that training an ESN model requires a given time $\mathcal{T}(n)$, the time PSO will spend optimizing the parameters in the worst case will be subjected to the number of iterations to perform and the number of particles. In another word, the

time that will be spent for optimization using PSO can be calculated simply by: $\mathcal{T}(n * iterations * number\ \ of\ \ particles) \in \mathcal{O}(n)$. Using PSO would not be very practical if the training process requires a large amount of time.

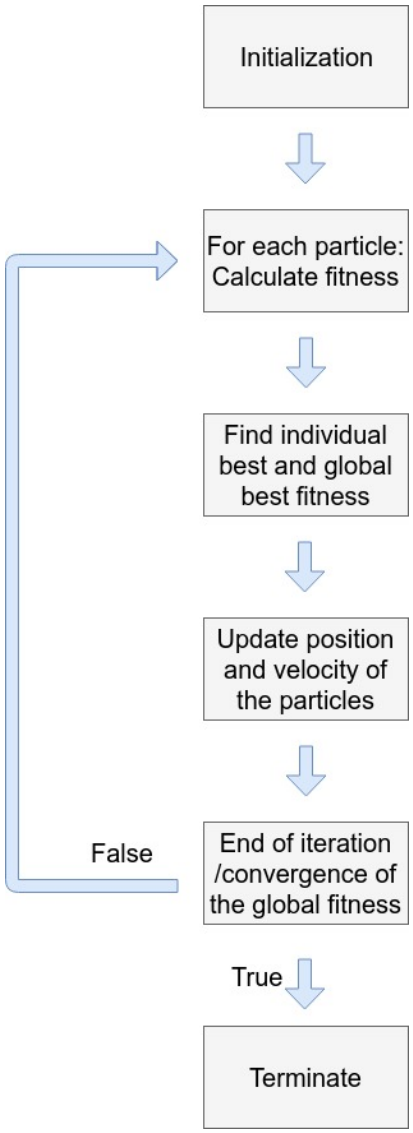


Figure 4.4.1: Flow chart of the PSO algorithm

Figure 4.4.1 shows flow chart of the PSO algorithm. After termination of the algorithm, the optimized parameters are used to make an ESN model. The ESN model designed using the optimized parameters are used for the prediction of the power demand time-series data.

4.5 Readout weight training of ESN

The readout weight can be trained in several different ways. Since the output signal is computed simply by multiplying the readout weights with the concentrated matrix $\boldsymbol{X}$, it is a linear and feed-forward process,

$$\boldsymbol{y} = \boldsymbol{W}_{out}\boldsymbol{X}.$$

The concentrated matrix $\boldsymbol{X}$ is presented using input, reservoir state, and bias column vectors,

$$\boldsymbol{X} = [\mathbf{1};\ \boldsymbol{u};\ \boldsymbol{x}] \tag{4.5.1}$$

$\mathbf{1}$ presents the bias term, $\boldsymbol{u}$ presented the input and $\boldsymbol{x}$ presents the reservoir states. Therefore, the dimensions of the matrices presented in Section 4.5 becomes $\boldsymbol{y} \in \mathbb{R}^{N_y \times T}$, $\boldsymbol{W}_{out} \in \mathbb{R}^{N_y \times (1+N_u+N_x)}$ and $\boldsymbol{X} \in \mathbb{R}^{(1+N_u+N_x)\times T}$, where T is the number of the training samples.

In this work, (2.3.5) is used with a very small regularization factor β. The training process requires using a large data-set. However, as it can be observed from (2.3.5),

$$\boldsymbol{y}_{target}\boldsymbol{X}^\top \in \mathbb{R}^{N_y \times (1+N_u+N_x)}$$

and

$$\boldsymbol{X}\boldsymbol{X}^\top \in \mathbb{R}^{(1+N_u+N_x)\times(1+N_u+N_x)},$$

these terms do not depend on the size of the training data-set. Neither memory usage, nor the time of the training procedure depends on the length of data-set. Moreover, pseudo-inverse requires high memory usage and calculation is memory-wise expensive. Thereby limiting the size of training samples. Therefore, the method presented in (2.3.5) is of the first choice.

4.6 Forecast strategies

RNN can use different types of features as the input, which in this case will be the time series historical data. The RNN model can take D number of the historical power demand and forecast power demand at the sequence number D + 1. The time interval to be predicted can be defined as a window W. Let's say W is defined to be 10. The algorithm will be shifted 10 times starting from predicting the data at D + 1. The algorithm will continue by using the predicted data as one of the inputs for the next prediction sequence. Another method will be forecasting the power demand at D + t ($t \in [1, W]$) directly by using W number of outputs. Multiple RNN models can be trained to make it less complex to catch the temporal dynamic of the time sequential data. Let's say N number of models will be used to forecast W number of future values, the forecasting domain W can be divided

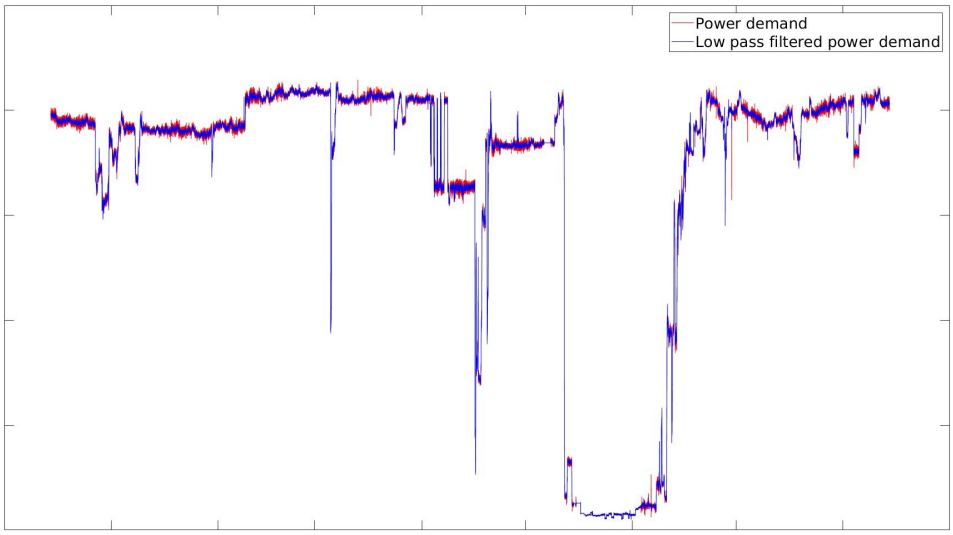
into N number of sub-domains and each model will then be mapped to each time sequential domain having $\frac{W}{N}$ outputs. The accuracy of prediction will depend on W. The accuracy is expected to be reduced as W increases.

The direct approach using multiple dimensional outputs is chosen in this work since the recurrent method with shifting window iteratively will be affected by the prediction error that is being accumulated. This accumulated error will be propagated and the prediction performance can quickly degrade as the prediction horizon increases. As the long-term optimization requires the prediction horizon to be reasonably long, the most adequate method for the problem at hand will be using the direct approach including a model with multiple dimensional outputs.

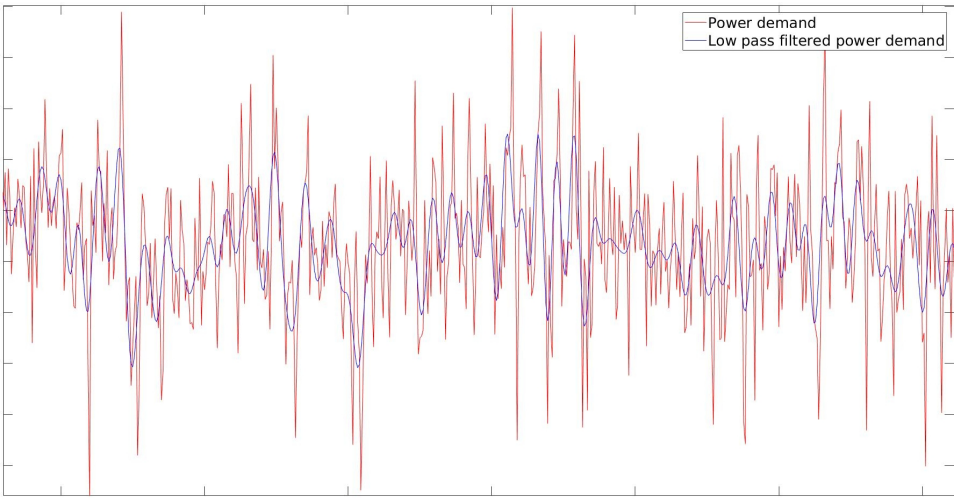
Improvement of the prediction result is expected to be achieved if multiple features are used as the input to the algorithm, not only the historical data. However, considering generality and simplicity of the algorithm as well as minimizing the data size and feature dimension, the only feature that will be used for the prediction is the historical data. It is desired to minimize the feature dimension in order to reduce the dependency of the algorithm from the different types of features. The symbol denoted as $\boldsymbol{u}(\mathrm{t})$ will represent the historical power demand in this Chapter.

4.7 Data pre-processing

Increased performance of prediction can be achieved by applying common data pre-processing methods. For the raw time-series data, using a low-pass filter can be considered in order to remove the high-frequency measurement noise as well as the other type of noise such as sensor readout noise, quantization noise, and photon noise.



(a) Power demand for the entire interval



(b) Power demand for part of the interval

Figure 4.7.1: Raw power demand vs. filtered power demand

4.7.1 Low-pass filtering

The raw power demand data is filtered using a low-pass Butterworth filter. The basic idea of Butterworth filter is briefly explained in Section 2.1. As it can be observed from Figure 4.7.1, high peaks of the measurements are removed, this is achieved without polluting the original data too much.

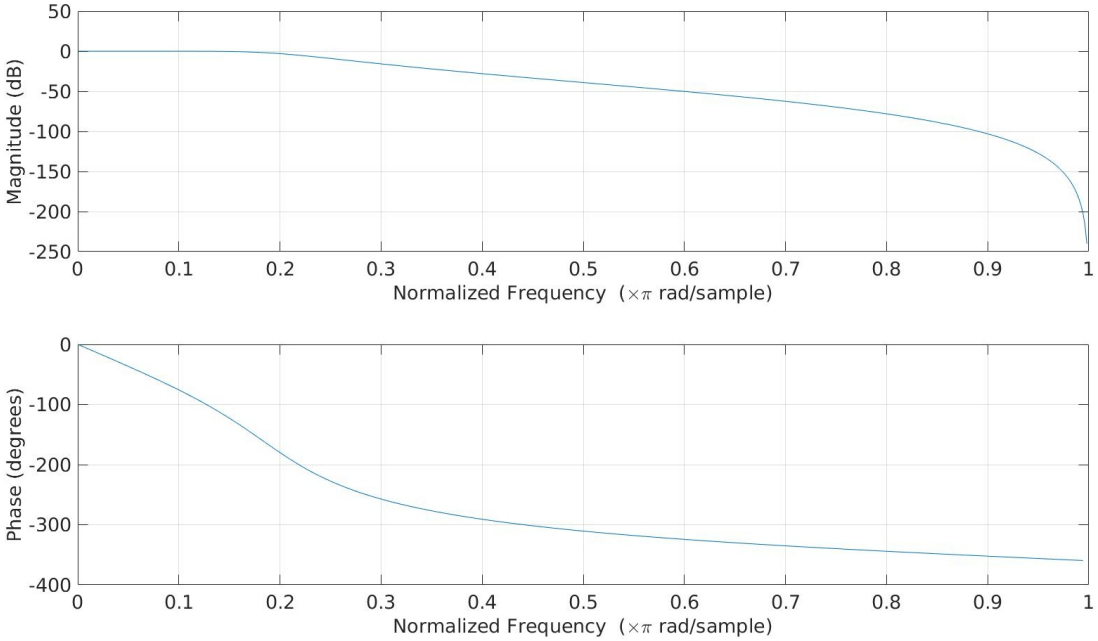


Figure 4.7.2: Frequency response of 4^{th} order low-pass Butterworth digital filter

The signals after the cut-off frequency can be filtered more smoothly by increasing the order of the filter. However, as the order increases so does its size and complexity, also its accuracy declines, therefore, depending on the application of the filter, the number of the order needs to be chosen carefully. In this work, 4^{th} order Butterworth filter has been used, which gives 160 dB per decade. The response of the filter is shown in Figure 4.7.2.

4.7.2 Feature standardization

To improve the quality of the forecasting result, feature standardization can be considered. The procedure makes the values of each feature in the data have zero-mean and unit-variance. In another word, all the features will be within the range $[-1\ 1]$. The simplest way to achieve this is using mean value and standard deviation of the feature vector. The formula yields,

$$\acute{\boldsymbol{d}} = \frac{\boldsymbol{d} - \bar{d} \cdot \mathbf{1}}{\sigma}. \tag{4.7.1}$$

$\acute{\boldsymbol{d}}$ is the normalized feature vector, $\boldsymbol{d}$ is the original feature vector, $\bar{d}$ is the mean of the feature vector and σ is used to denote the standard deviation of the feature vector.

In general, feature standardization helps the machine learning algorithm to improve the result. This is due to the fact that the range of raw feature values vary widely. Given that a time-series data has a broad range of values for a specific period of time, the training will be affected by this interval and generality of the algorithm might be downgraded. Moreover, the most commonly used activation functions such as sigmoid has a range between $[-1\ 1]$, which means that the output using non-normalized data will most likely be very close to the boundary of the activation functions. To avoid this phenomenon, feature standardization is considered and frequently used.

CHAPTER 5
OPTIMIZATION PROBLEM MODEL

5.1 Introduction

The UC problem is an optimization problem to schedule the number of commitment of power generating units at the least possible cost. The ED problem is another type of optimization problem to find the power level of the committed units to meet power demand at the least possible cost. The cost term is task dependent and can include different terms such as emission, price, fuel usage etc.

The ED and UC problems including different types of power generating units such as gas turbines, electrical grid, and intermittent resources can be solved using an optimization problem model by means of state equations. The state variables will represent the power output of each power generating unit and the input variables will be denoted as the unit commitment.

A recent study covering the UC and ED problems for optimizing fuel consumption of vessels [13] has shown the fuel saving of 4-6 %. In [13], SFOC is estimated using an online estimation method and the optimal power distribution is proposed using integer programming considering on/off decisions of diesel generators with different power outputs. In [15], operational data from the ferry, platform supply vessel and seismic survey vessel is used to observe fuel savings using MILP algorithm and logic-based algorithm. Furthermore, usage of ESS is experimented. In [16], MILP is combined with prediction methods and the MILP model includes scheduling the future power production and the future UC of the generators. In [17], different types of renewable energy sources are included in a MILP model as well as gas turbines. The algorithm is used for energy distribution scheduling in terms of cost-efficient operation. In [18], a MILP model is designed using dynamics of hydro and

thermal units. The non-linear behaviors are transformed using piece-wise linear approximation and included as a part of the constraints. The future condition of hydro units is estimated using stochastic dynamic programming.

The ED and UC problems were solved in [19] as a one combined problem. The dynamics of power generating units are considered for designing constraints. A piece-wise linear approximation is applied to transform MINLP to MILP and the results obtained solving MINLP and MILP are compared. The MINLP problem includes a quadratic objective function and the computational time increases exponentially with the size of the problem. With the proposed linearization method, the complexity of the problem was reduced and prevented the computational time from increasing drastically. MPC model is presented in [20] using dynamic model for the UC problem for wind farms. The UC problem is solved by combining MPC method with short-term wind power forecasting. An MPC approach is presented in [22] for solving ED of an electrical power system. Prediction of load demand and available power of intermittent resources is used together with MPC for solving ED. Robustness and feasibility of the MPC solutions for solving ED of an electrical power system are proven in [21]. Moreover, [21] recommends using MPC by demonstrating drawbacks of solving the OCDD and DED offline including the UC and ED problems and showing how the violation on the constraint regarding ramp rate can be prevented using the MPC method.

ILP has been applied in [13] to successfully solve the ED and UC combined problem. However, this approach is quite limited since the problem size shows quadratic grows as the number of generating unit increases. Since ILP approach requires the power interval to be divided into a finite number of power domains. To overcome this problem, using MILP can be considered. The model presented in [15] requires many conditional constraints, which increases the complexity of the implementation. The conditional constraints can be avoided by representing the constraints in a more compact way. The work presented in [16] shows the possibility of combining a predictive method with an optimization algorithm. [18] has proved that renewable energy sources can be modeled using MILP. [18] and [19] shows how a piecewise linear approximation can be applied to simplify the MINLP model to MILP. In general, MILP can be solved much faster compared to the MINLP models. The work in [20], [21] and [22] shows how the robustness can be increased using an MPC approach based on a MILP/MINLP model for solving the ED and UC problems. MPC can also be considered in this work to increase the robustness of the algorithm since an open-loop approach can easily violate the model constraints.

The static optimization method presented in [1] will not consider dynamic of the power system. Hence, the static optimization solution from [1] lack

robustness of adapting the fast changes in the power system. Moreover, it is not able to fully capture the saving potential due to the lack of the information about the forthcoming events. In addition, the static optimization method is very sensitive to ramp rate violation and is hard to prevent this violation. By introducing a dynamic approach by integrating a predictive method for implementing EMS, such drawbacks can be overcome.

In this work, an MINLP model is proposed and will be used for solving the UC and ED combined problem. The predictive method presented in Chapter 4 will provide the future power demand and an EMS will be designed using a MINLP model. The MINLP model will not consider the dynamics of the gas turbines since the algorithm will only be used to find the optimal power set-points of the gas turbines. Meaning that the algorithm will only run with a low frequency (0.01 Hz) to find the set-points to be reached by low-level controllers. Furthermore, the appropriate constraints will be added to the model considering soundness and feasibility of the algorithm when run in realistic industrial environments.

The model will include both binary and continuous decision variables, introducing a MINLP model. The MINLP model will be non-convex and NP-hard. However, the adequate solvers presented in Chapter 6.1 will manage to solve the problem effectively.

5.2 BSFC estimation

The BSFC curve for each gas turbine is found using an offline curve-fitting method provided by MATLAB, the further improvement of the offline BSFC estimation approach is presented in [2]. As the BSFC will vary depending on the load [13], BSFC curve will be a function of the load

$$BSFC(load),$$

which has unit $\left[\frac{Kg}{MWh}\right]$. The BSFC parameters are the key parameters of the optimization algorithm.

5.3 Fuel optimization

5.3.1 Objective function

There exists N-number of gas turbines that needs to be controlled. Given that the fuel consumption per sampling time for the ith machine f_i can be found by the product between BSFC for a given load $b_i(load_i(p_i))$ and power

p_i, the objective function can be formulated as follows

$$f(\mathbf{p}) = \sum_{i=1}^{I} b_i(load_i(p_i)) \cdot p_i = \sum_{i=1}^{I} f_i(load_i(p_i), p_i),$$

where BSFC with the unit $\left[\frac{Kg}{MWh}\right]$, has been scaled by a constant factor such that the unit becomes $\left[\frac{Kg}{MW\Delta t}\right]$, where Δt denotes the sampling interval. The total fuel consumption for a given time horizon T (the prediction horizon), can be found by a summation as follows, where each t denotes a sample time

$$\sum_{t=1}^{T} f(\mathbf{p}(t)) = \sum_{t=1}^{T}\sum_{i=1}^{I} b_i(load_i(p_i(t), t)) \cdot p_i(t) = \sum_{t=1}^{T}\sum_{i=1}^{I} f_i(load_i(p_i(t), t), p_i(t)).$$

Since the maximum power output of the gas turbines varies with time, the variable *load* will vary with time. However, in this work, for the simplicity of designing the cost function of the optimization model, the maximum power of the gas turbines will be assumed to be fixed for the given time horizon T. This assumption is valid as long as T is chosen reasonably short. The maximum power is calculated in a special way using different types of parameters such as temperature, hours of operation etc. that does not vary drastically on a short period of time. Therefore the total fuel consumption can be calculated as follows

$$\sum_{t=1}^{T} f(\mathbf{p}(t)) = \sum_{t=1}^{T}\sum_{i=1}^{I} b_i(load_i(p_i(t))) \cdot p_i(t) = \sum_{t=1}^{T}\sum_{i=1}^{I} f_i(load_i(p_i(t)), p_i(t)). \tag{5.3.1}$$

The BSFC curves are non-linear, therefore, the objective function that include BSFC curves becomes non-linear. The optimal $\mathbf{p}$ can be found by minimizing the objective function presented in (5.3.1).

$$\underset{\mathbf{p}}{\text{minimize}} \quad \sum_{t=1}^{T} f(\mathbf{p}(t))$$

5.3.2 Start-up cost

An additional aspect, which can be taken into account is the cost of turning on the gas turbines. The cost can be calculated using the fuel consumption on different power levels. It should be cheaper to increase power from an already operating gas turbine instead of turning on a new gas turbine. However, adding an additional term to the cost function is considered unnecessary since the algorithm will always find it more energy efficient to increase the power level of the already operating gas turbines, considering

BSFC. The turned off gas turbines will, therefore, stay turned off until it is absolutely necessary to turn them on to meet the power demand constraint (5.3.2).

5.3.3 Power demand balance

The amount of power produced has to match power demand,

$$\sum_{i=1}^{I} p_i(t) = p_d(t), \quad \forall t \in [1, ..., T]. \tag{5.3.2}$$

power demand changes over time and is dependent on the multiple factors such as the grid loading, weather, tide and how much power is required to operate the site. power demand also includes power change in order to keep the stability of the grid frequency. The frequency of the grid is the nominal frequency of the oscillations of AC in an electric power grid transmitted from a power station to the end-user. Depending on the location, the frequency of the grid can vary. In Europe, this frequency is 50 Hz. If the consumption exceeds the generation, the frequency of the AC will fall to a value below 50 Hz. In this case, either more power needs to be produced by the producers or load reduction can be requested to the major energy users. If the generation of power exceeds the consumption, the frequency will rise to a value above 50 Hz. The grid operator then needs to ensure that the power producers reduce the generation of power.

Power demand will be predicted using the method presented in Chapter 4, which allows the optimization algorithm to consider the future power demand. This will help the EMS algorithm to catch the dynamic of the power system and find a more reliable cost-efficient way to distribute the load throughout the system.

5.3.4 Ramp rate constraint

The ramp rate limit of the gas turbines needs to taken into account since the dynamic of the gas turbines only allows to change its power between each time sample within a defined limit

$$\mathbf{L}_{\Delta t} \leq \mathbf{p}(t) - \mathbf{p}(t-1) \leq \mathbf{H}_{\Delta t}, \quad \forall t \in [1, ..., T]. \tag{5.3.3}$$

where $\mathbf{p}(0)$ is the initial power generated from the gas turbines. The ramp rate limits $\mathbf{H}_{\Delta t}$ and $\mathbf{L}_{\Delta t}$ needs to be chosen considering the operational condition and the model of the gas turbines. In this work, the ramp rate limit is chosen to be H_s: 0.35 $\frac{MW}{s}$ and L_s: -0.35 $\frac{MW}{s}$. The lower limit is chosen to be $\|0.35\|\frac{MW}{s}$ considering the advice from the operators at the site and inspection of the data retrieved from the site. The gas turbines can

only be turned off when it is operating at the power of 3.5 MW given that the sampling frequency is 10 seconds, because of the ramp limit constraint. However, it is often desirable to turn off the gas turbines without decreasing the power output of the gas turbines all the way to 3.5 MW. This characteristic can be obtained by using Big-M relaxation on (5.3.3). The new constraint yields

$$\mathbf{L}_{\Delta t} - M_{rp,\Delta t} \cdot (\mathbf{1} - \boldsymbol{u}(t)) \leq \mathbf{p}(t) - \mathbf{p}(t-1) \leq \mathbf{H}_{\Delta t}, \quad \forall t \in [1, ..., T].$$

The relaxation term $M_{rp,\Delta t} \cdot (\mathbf{1} - \boldsymbol{u}(t))$ gives the algorithm freedom to turn off the gas turbines without decreasing the power output of the gas turbines to $\boldsymbol{L}_{\Delta t}$. $M_{rp,\Delta t}$ can be chosen based on the model of the gas turbines. In this work, $M_{rp,s}$ is set to be 0.5 $\frac{MW}{s}$. $M_{rp,10s}$ becomes 5 MW given that the sampling frequency is 10 seconds.

5.3.5 Power capacity constraint

The constraint

$$\mathbf{p}_{min}(t) \odot \mathbf{u}(t) \leq \mathbf{p}(t) \leq \mathbf{p}_{max}(t) \odot \mathbf{u}(t), \quad \forall t \in [1, ..., T]$$

is used to make sure that all the gas turbines are operating in the feasible power region. The symbol $\odot$ denotes element-wise multiplication.

Hence,

$$\begin{aligned} \mathbf{p}_{min}(t) &= \begin{bmatrix} p_{1,min}(t) & , \ldots , & p_{N,min}(t) \end{bmatrix}, \\ \mathbf{p}_{max}(t) &= \begin{bmatrix} p_{1,max}(t) & , \ldots , & p_{N,max}(t) \end{bmatrix}. \end{aligned}$$

$\boldsymbol{p}_{min}(t)$ is in most of the cases set to $\mathbf{0}$. Meanwhile, $\boldsymbol{p}_{max}(t)$ is found by the site operators and is provided to the algorithm. In this work, $\boldsymbol{p}_{max}(t)$ will be assumed to be known with complete certainty for $\forall t \in [1, ..., T]$.

5.3.6 Process required constraint

The maximum power that can be generated by each gas turbine is dependent on the weather, temperature, humidity, etc. This must be taken into account and be used as a constraint. One additional constraint that needs to be added is that at least n_{min} number of turbines needs to be active at a time and l_{min} number of turbines needs to produce power at the minimum of p_{req}. n_{min} and l_{min} must be chosen such that $n_{min} \geq l_{min}$. The minimum power constraint is not required for the rest of the turbines. This comes from a process requirement from the site. This type of constraint is in general

defined by the users. This can be modelled as follows

$$\begin{aligned}
\sum_{i=1}^{I} u_i(t) &\geq n_{min}, \\
\sum_{i=1}^{I} p_{i,min}(t) &= p_{req} \cdot \left[\sum_{i=1}^{I} u_i(t) - l_{min} \right], \\
\mathbf{0} &\leq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t), \\
\mathbf{p} &\geq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t), \\
p_{i,min}(t) &\in \{0, p_{req}\}, \\
\forall t &\in [1, ..., T].
\end{aligned} \tag{5.3.4}$$

Equation (5.3.4) ensures that only n given number of the turbines will be restricted to generate the minimum power required. The other turbines are free to operate at any power above 0 MW.

5.3.7 Start-up time constraint

A gas turbine spends about 2 hours to get ready to be operated [50]. This constraint needs to be included in the optimization problem. This can be obtained by adding a summation term on the unit commitment u of the power generating units combining with Big-M relaxations as follows,

$$\sum_{\tau=t-2h}^{t-1} u_i(\tau) \leq 2h \cdot (1 - u_i(t)) + M_{st} \cdot u_i(t-1), \quad \forall t \in [1, ..., T]. \tag{5.3.5}$$

The size of M_{st} can be chosen based on the time it takes to turn on a gas turbine, which will be 2 hours in this work. Given that the ith gas turbine has been operating for last two hours, $\sum_{\tau=t-2h}^{t-1} u_i(\tau)$ will be 2h and $u_i(t-1)$ will be one, which allows $u_i(t)$ to be either zero or one. Let's say the gas turbine has been turned off for last two hours, the summation term on the left hand size will be zero as well as $u_i(t-1)$, hence $u_i(t)$ can be set to one or stay at zero. If the summation term is in between 0 and 2h and $u_i(t-1)$ is one, which indicates that the ith gas turbine has been turned on recently, $u_i(t)$ can still be freely chosen. However, if $u_i(t-1)$ is zero, which indicates that the given gas turbine need more time to get ready to be operated, $u_i(t)$ is restricted to be zero. which secures the gas turbines to be turned off for 2 hours before it can start operating.

The constraint presented in (5.3.5) will ensure that only the operation-able gas turbines are used to produce power. However, this approach with T $<$ 2 hours will be sub-optimal since it only secures the gas turbines to be turned off for 2 hours and will not provide the optimal point for start preparing the

gas turbines. In order to make the optimization algorithm to catch the time to start preparing a gas turbine to be operated, T needs to be bigger than 2 hours. This requires the prediction method to extend its prediction horizon considerably, which will introduce a bigger prediction error.

5.3.8 The fuel consumption optimization problem model

The optimization problem can be formulated as follows with p and u as the decision variables

$$
\begin{aligned}
\underset{\mathbf{p},\mathbf{u}}{\text{minimize}} \quad & \sum_{t=1}^{T} f(\mathbf{p}(t)) \\
\text{subject to} \quad & \sum_{i=1}^{I} p_i(t) = p_d(t) \\
& \mathbf{L}_{\Delta t} - M_{rp,\Delta t} \cdot (\mathbf{1} - \boldsymbol{u}(t)) \leq \mathbf{p}(t) - \mathbf{p}(t-1) \leq \mathbf{H}_{\Delta t} \\
& \mathbf{p}(t) \leq \mathbf{p}_{max}(t) \odot \mathbf{u}(t) \\
& \sum_{i=1}^{I} u_i(t) \geq n_{min}, \\
& \sum_{i=1}^{I} p_{i,min}(t) = p_{req} \cdot \left[\sum_{i=1}^{I} u_i(t) - l_{min} \right] \\
& \mathbf{0} \leq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& \mathbf{p} \geq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& p_{i,min}(t) \in \{0, p_{req}\} \\
& \sum_{\tau=t-2h}^{t-1} u_i(\tau) \leq 2h \cdot (1 - u_i(t)) + M_{st} \cdot u_i(t-1) \\
& u_i(t) \in \{0, 1\} \\
& \forall i \in [1, ..., I], \quad \forall t \in [1, ..., T].
\end{aligned}
\tag{5.3.6}
$$

5.4 Cost optimization

5.4.1 Grid power

Import/export of power from an electrical grid can be controlled based on the price of electricity. In order to add this term to the current model, the objective function needs to be changed to include the price of producing the energy required. Fuel consumption itself cannot be used as the cost since the optimization algorithm will always choose to import, which does not require consuming fuel. The price can simply be added to the objective function by using the price of fuel. The gas price is fixed to be 350 $\frac{NOK}{MWh}$. The unit $\frac{NOK}{MWh}$ is transformed to $\frac{NOK}{kg}$ and is multiplied by the total fuel consumption. To find the price of electricity from the grid, *Nord pool* electricity price is used. Figure 5.4.1 shows the electricity price [6]. The electricity price varies on an hourly basis, hence as long as T $\leq$ 1 hour, the electricity price can be assumed to be fixed for the optimal power distribution scheduling horizon T.

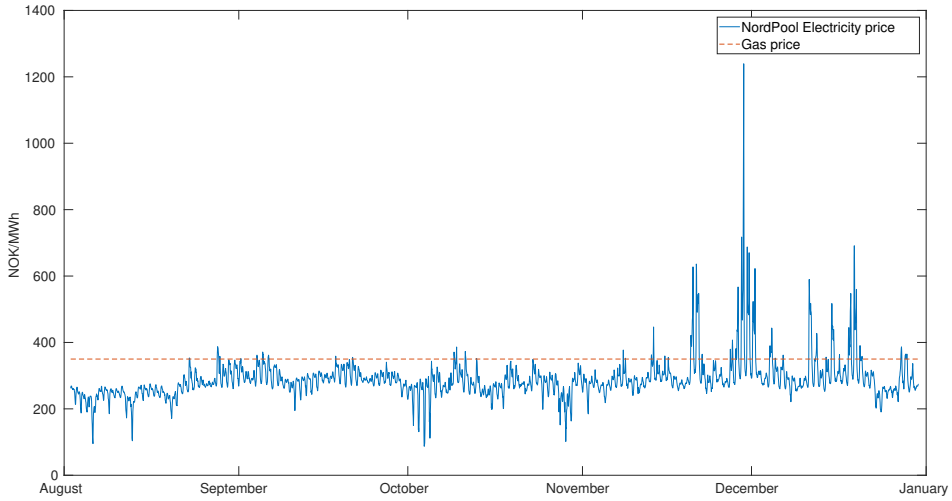


Figure 5.4.1: Grid electricity price vs. gas price

An EMS with different power sources supported by the public grid can be defined as a Hybrid Power System (HPS) [7]. HPS can be modelled as MINLP by modifying the model presented in (5.3.6).

The objective to be minimized will be the cost.

$$C(\mathbf{p}, p_g) = \sum_{t=1}^{T} f(\mathbf{p}(t)) \cdot C_{fuel} + \sum_{t=1}^{T} (p_g(t) \cdot C_{el}(t)) \tag{5.4.1}$$

p_g denotes power from the grid. The unit of $C(\mathbf{p}, p_g)$ is NOK and the unit of C_{el} is $\frac{NOK}{MW\Delta t}$.

5.4.2 Grid power change penalty

Cost of the power difference between each time step can be introduced. This power difference term can be used to achieve the smooth set-point transition of the grid power.

To obtain moderate behavior of power from the grid, a power change penalty term can be added to the objective function. The weight of this cost term is chosen by trial and error. Based on the result of the simulation, if the grid power is changed too excessively, the weight was increased and vice versa.

$$w_{\Phi} \cdot \sum_{t=1}^{T} |p_g(t) - p_g(t-1)|,$$

$p_g(0)$ is the initial power from the grid. The absolute value term introduces non-linearity to the model. However, this term can be removed by introducing a new variable that will be used as the minimum and the maximum boundary of the difference.

$$-\Phi(t) \leq (p_g(t) - p_g(t-1)) \leq \Phi(t)$$
$$\Phi(t) \geq 0, \quad \forall t \in [1, ..., T].$$

The grid power change penalty that will be added to the objective function becomes

$$w_{\Phi} \cdot \sum_{t=1}^{T} \Phi(t).$$

5.4.3 Grid power set-point penalty

Transmission of power from the power generating stations require high voltage transmission line. However, the transmission lines are not perfectly reliable. Moreover, an entire grid runs at the same frequency and the generators providing power to the grid needs to run at the same frequency and stay nearly in phase with each other and the grid. This requires a control mechanism, such that if the grid is heavily loaded, the frequency slows and the generators need to produce more power. If the grid is lightly loaded, the frequency will increase, and the generators will reduce their power output. Sometimes, operators of transmission of the grid request load reduction from the major energy users such as industrial plants for the grid frequency regulation. Meaning that the grid should not be loaded too heavily or too lightly in order to maintain the stability of the grid.

The grid power can be controlled by adding a set-point penalty to the objective function,

$$\sum_{t=1}^{T} \left(p_g(t) - p_{g,sp}\right)^2.$$

To balance the set-point term to the cost objective function, a reasonable size of weight needs to be chosen. This grid set-point penalty weight can be chosen using C_{el}, which is again multiplied with a constant scalar $w_{\Phi,g}$, such that

$$w_{\Phi,g} \cdot C_{el} \cdot \sum_{t=1}^{T} (p_g(t) - p_{g,sp})^2 \tag{5.4.2}$$

is the set-point penalty term to be added to the objective function, where,

$$w_{\Phi,g} = 0.02. \tag{5.4.3}$$

After some trials and analysis, the penalty factor was chosen to be 0.02. The bigger penalty with $p_{g,sp}$ set to zero will result in less grid power dependency. Grid dependency is an important aspect that needs to be taken into consideration when operating power system of a site. Due to the fact that it can influence the reliability of the system.

There are different ways to include the set-point penalty term in the model. For instance, a similar approach presented in Section 5.4.2 can be used by introducing a new variable that finds the difference between the set-point defined and the grid power, which is used in [1]. This term will penalize all the range of grid power that deviates from the set-point proportionally. Meanwhile, the model presented in this section will allow the small deviation of the grid power from the set-point to be penalized less than the grid power that deviates much from the set-point defined. The term (5.4.2) will be used to reduce the overall dependency of the grid power as well as to reduce the loss when the grid-failure occurs. The loss will be bigger as more power is imported instantly from the grid, which is prevented to a certain degree using a second order term. The loss can be minimized if alternative energy sources can cover the amount of energy, which was planned to be imported from the grid. Depending on the desired behavior, the model can be modified.

5.4.4 Grid power capacity constraint

The constraint

$$-p_{g,lim} \leq p_g \leq p_{g,lim}$$

secures the algorithm to choose a feasible amount of grid power that is exported/imported instantly. Based on the information from the site, $p_{g,lim}$ is set to be 115 MW.

5.4.5 Grid failure

The grid providing power to the site can fail. In another word, a power blackout that leads to a short-term or a long-term loss of the electric power to a particular area. There are many causes of power failures in an electrical

network. In terms of reliability, this is a crucial part that needs further investigation. The optimization algorithm needs to be designed such that grid failures can be controlled in order to secure robustness of the algorithm when applied in realistic industrial environments. However, in order to include the grid failures to the algorithm, it requires prediction of the grid failure points. Considering the scope of this study and lack of available data, handling the grid failure will not be covered in this work.

5.4.6 The cost optimization problem model

The optimization problem can be formulated as follows with p, p_g and u as the decision variables

$$
\begin{aligned}
\underset{\mathbf{p},p_g,\mathbf{u}}{\text{minimize}} \quad & C(\mathbf{p},p_g) + w_\Phi \cdot \sum_{t=1}^{T} \Phi(t) + w_{\Phi,g} \cdot C_{el} \cdot \sum_{t=1}^{T} (p_g(t) - p_{g,sp})^2 \\
\text{subject to} \quad & \sum_{i=1}^{I} p_i(t) + p_g(t) = p_d(t) \\
& \mathbf{L}_{\Delta t} - M_{rp,\Delta t} \cdot (\mathbf{1} - \boldsymbol{u}(t)) \leq \mathbf{p}(t) - \mathbf{p}(t-1) \leq \mathbf{H}_{\Delta t} \\
& \mathbf{p}(t) \leq \mathbf{p}_{max}(t) \odot \mathbf{u}(t) \\
& -p_{g,lim} \leq p_g \leq p_{g,lim} \\
& \sum_{i=1}^{I} u_i(t) \geq n_{min}, \\
& \sum_{i=1}^{I} p_{i,min}(t) = p_{req} \cdot \left[\sum_{i=1}^{I} u_i(t) - l_{min} \right] \\
& \mathbf{0} \leq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& \mathbf{p} \geq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& p_{i,min}(t) \in \{0, p_{req}\} \\
& \sum_{\tau=t-2h}^{t-1} u_i(\tau) \leq 2h \cdot (1 - u_i(t)) + M_{st} \cdot u_i(t-1) \\
& -\Phi(t) \leq p_g(t) - p_g(t-1) \leq \Phi(t) \\
& \Phi(t) \geq 0 \\
& u_i(t) \in \{0,1\} \\
& \forall i \in [1,...,I], \quad \forall t \in [1,...,T].
\end{aligned}
\tag{5.4.4}
$$

5.5 Cost optimization integrating ESS

5.5.1 ESS modeling

The already existing HPS from (5.4.4) can be expanded further to include ESS to analyze how ESS can improve the cost-efficiency. ESS can give a cost-effective solution in a way that the new hybrid system uses excess power from the gas turbines to charge the battery while keeping BSFC at its optimum. Moreover, the battery responds quickly to the high-frequency changes in power demand and allows the gas turbine to operate without the need of aggressive power output changes, in another word, ESS can be used for the grid frequency regulation. ESS can also help the system to exploit the grid power better in a way that ESS saves power when the grid electricity is at its low cost. If either the grid or the turbines fail, ESS can kick in immediately, which will increase the reliability of the system. In [15], a MILP model including ESS is proposed and the result shows that ESS will increase the effectiveness of the power distribution significantly.

Batteries assembled from lithium-ion cells will be coupled with the power system. The ESS is assumed to have a capacity of CC MWh and can provide power up to ±C MW. The ESS capacities and the battery-type is chosen based on the *Tehachapi Wind Energy Storage Project* [51]. Furthermore, the ESS is assumed to have an operating range between 70% and 100% of the capacity. The start capacity of the ESS is set to be 70%. The ESS is allowed to be discharged to the capacity below 70% if one of the gas turbines or the grid fails.

The ESS can be in two different states, charging and discharging modes. In discharging mode, the power output $p_{j,E}$ from jth ESS will be positive and in charging mode, $p_{j,E}$ will be negative. Moreover, an ESS in charging mode will have its corresponding u equals zero. In discharging mode, u will be one. The objective function includes the cost of charging ESS at each time step t,

$$r \cdot C_{fuel,MW\Delta t} \cdot \mathbf{p}_E^{\top}(t)(\mathbf{1} - \mathbf{u}_E(t)), \quad \forall t \in [1, ..., T], \tag{5.5.1}$$

where u_E denotes the unit commitment of an ESS and r is the recovery ratio and can be found in Table 5.1.

The cost of excess fuel that is used to charge the ESSs will be compensated by the *negative* cost of charging the ESSs. The term presented in (5.5.1) will be negative since $\mathbf{p}_E < 0$ in charging mode.

The following term will present the cost of using power from the ESSs at

each time step t

$$\mathbf{p}_E^\top(t)\mathbf{u}_E(t) \cdot C_{fuel,MW\Delta t}, \quad \forall t \in [1, ..., T].$$

The price of fuel

$$C_{fuel,MW\Delta t}\left[\frac{NOK}{MW\Delta t}\right],$$

is used to calculate the price of charging and discharging the ESSs in $[NOK]$. This will make the ESSs to be charged and discharged depending on the price of electricity. ESSs are assumed to react to the price of electricity. Moreover, operational point of the gas turbines in which, the price of power depends on the BSFC will impact the state of ESSs.

The dimension of $\mathbf{u}_E$ and $\mathbf{p}_E$ will be $\mathbb{R}^{J\times 1}$, where J is the number of the operative ESSs.

A new constraint can be introduced that limits charging/discharging rate of the ESSs,

$$-p_{E,min} \cdot \mathbf{1} \odot \mathbf{u}_E(t) + p_{E,min} \cdot \mathbf{1} \leq \mathbf{p}_E(t) \leq p_{E,max} \cdot \mathbf{1} \odot \mathbf{u}_E(t) \\ \forall t \in [1, ..., T].$$

$p_{E,min} \cdot \mathbf{1}$ limits the charging rate of the ESSs, hence $p_{E,min} \cdot \mathbf{1} < \mathbf{0}$ and $p_{E,max} \cdot \mathbf{1}$ limits the discharging rate of the ESSs, hence $p_{E,max} \cdot \mathbf{1} > \mathbf{0}$. In this work, $p_{E,min}$ and $p_{E,max}$ are set to be -8 MW and 8 MW respectively [51].

The energy stored in the ESSs can be calculated as follows,

$$\mathbf{E}(t) = \mathbf{E}(t-1) \\ - \left[\mathbf{p}_E(t-1) \odot \mathbf{u}_E(t-1) + r \cdot \mathbf{p}_E(t-1) \odot (\mathbf{1} - \mathbf{u}_E(t-1))\right] \cdot \frac{\Delta t}{3600}, \\ \forall t \in [2, ..., T+1]$$

where $\mathbf{E}(1)$ denotes the initial energy stored in the ESSs. $r \in [0, 1]$ represents the recovery ratio of the power conversion. Note that $\mathbf{E}(T+1)$ is included in the model. $\mathbf{E}(T+1)$ will be used as the initial stored energy of ESSs for the next sequential time horizon: $\forall t \in [T+1, ..., 2T+1]$.

The states of the ESSs are decided relative to the capacity of the ESSs. If the total capacity of the ESSs is about to be exceeded, the ESSs is constrained to operate in discharging mode and vice versa. This statement can be transformed into two conditional constraints as follows

$$E_j(t) - p_{E,max} \cdot \frac{\Delta t}{3600} \leq E_{min} \Longrightarrow u_{j,E}(t-1) = 0 \qquad (5.5.2) \\ \forall t \in [2, ..., T+1]$$

and

$$E_j(t) - p_{E,min} \cdot \frac{\Delta t}{3600} \geq E_{max} \Longrightarrow u_{j,E}(t-1) = 1 \qquad \forall t \in [2, ..., T+1] \tag{5.5.3}$$

where j is the ESS index. The ESS conditional constraint (5.5.2) needs to be considered only when the corresponding ESS with the index j is in discharging state, $u_{j,E}(t) = 1$. Moreover, (5.5.3) has to be considered only when the jth ESS is in charging state, $u_{j,E}(t) = 0$. Therefore, (5.5.2) and (5.5.3) can be transformed to

$$u_{j,E}(t-1) = 1 \Longrightarrow E_{min} \leq E_j(t) - p_{E,max} \cdot \frac{\Delta t}{3600} \qquad \forall t \in [2, ..., T+1] \tag{5.5.4}$$

and

$$u_{j,E}(t-1) = 0 \Longrightarrow E_j(t) - p_{E,min} \cdot \frac{\Delta t}{3600} \leq E_{max} \qquad \forall t \in [2, ..., T+1] \tag{5.5.5}$$

The constraints (5.5.4) and (5.5.5) can be again transformed using Big-M notation such that

$$\begin{aligned} E_{min} &\leq E_j(t) - p_{E,max} \cdot \frac{\Delta t}{3600} + M_E \cdot (1 - u_{j,E}(t-1)) \\ E_j(t) - p_{E,min} \cdot \frac{\Delta t}{3600} &\leq E_{max} + M_E \cdot u_{j,E}(t-1) \\ &\qquad \forall t \in [2, ..., T+1]. \end{aligned}$$

The Big-M constant M_E also prevents the ESS capacity constraints from being violated in a $failure\ state$ of the power system. If one of the gas turbines or the grid trips, the $failure\ state$ is entered and the ESSs are allowed to be discharged further to the capacity below 70 % by reducing the minimum capacity E_{min}. If the tripped gas turbine/grid has been recovered (E_{min} is changed back to 22.4 MWh) and meanwhile the energy of the ESSs are being charged back to the *normal state*, the minimum ESS energy constraint is violated, however, this violation can be relaxed by having the Big-M constant M_E.

Table 5.1: ESS parameters

Parameter	Value
Recovery ratio, r	80 - 90 %
ESS max. capacity, E_{max}	32 MWh (100 %)
ESS min. capacity, E_{min}	22.4 MWh (70 %)
ESS discharging rate, $p_{E,max}$	8 MW
ESS charging rate, $p_{E,min}$	-8 MW

Table 5.1 shows all the necessary parameters needed for lithium-ion ESS. The numbers are from the *Southern California Edison / Tehachapi Wind Energy Storage Project* presented in [51].

The approximated recovery ratio r is used to define how much power will be lost when charging the ESSs. Power loss can occur due to the power conversion, heating energy, during the electricity transfer to the Li-Ion cells, cooling, etc.

5.5.2 GT power change penalty

One additional cost term that can be added is the cost of changing set-points of the operating gas turbines. Similar to the cost term presented in Section 5.4.2,

$$w_{\Phi} \cdot \sum_{i=1}^{I} \sum_{t=1}^{T} \Phi_{i,gt}(t),$$

with the constraint

$$-\boldsymbol{\Phi}_{gt}(t) \leq (\boldsymbol{p}(t) - \boldsymbol{p}(t-1)) \leq \boldsymbol{\Phi}_{gt}(t)$$
$$\boldsymbol{\Phi}_{gt}(t) \geq \mathbf{0}, \quad \forall t \in [1, ..., T].$$

By adding this term, power from the gas turbines can be stabilized [1]. The ESSs are used to shift the total load and used for frequency regulation. The varying power demand can also cause wear and tear of the gas turbines since it requires the gas turbines to change its power output aggressively, however, this can be controlled by using the ESSs and the maintenance cost can be reduced. It might also be hard for the low-level power controller to control the power of the gas turbines for the grid stabilization. In some cases, the dynamic of the gas turbines would not let power demand to be met, however, with the ESSs adjusting the total load, this problem can be overcome.

5.5.3 The cost optimization problem model integrating ESS

The optimization problem can be formulated as follows with p, p_g, p_E, u and u_E as the decision variables

$$\begin{aligned}\underset{\mathbf{p},p_g,\mathbf{p}_E,\mathbf{u},\mathbf{u}_E}{\text{minimize}} \quad & C(\mathbf{p},p_g)+w_\Phi\cdot\sum_{t=1}^{T}\Phi(t)+w_{\Phi,g}\cdot C_{el}\cdot\sum_{t=1}^{T}\left(p_g(t)-p_{g,sp}\right)^2 \\ & +r\cdot C_{fuel,MW\Delta t}\cdot\sum_{t=1}^{T}\mathbf{p}_E^\top(t)(\mathbf{1}-\mathbf{u}_E(t)) \\ & +C_{fuel,MW\Delta t}\cdot\sum_{t=1}^{T}\mathbf{p}_E^\top(t)\mathbf{u}_E(t) \\ & +w_\Phi\cdot\sum_{i=1}^{I}\sum_{t=1}^{T}\Phi_{i,gt}(t)\end{aligned}$$

$$
\begin{aligned}
\text{subject to} \qquad & \sum_{i=1}^{I} p_i(t) + p_g(t) + \sum_{j=1}^{J} p_{j,E}(t) = p_d(t) \\
& \mathbf{L}_{\Delta t} - M_{rp,\Delta t} \cdot (\mathbf{1} - \boldsymbol{u}(t)) \leq \mathbf{p}(t) - \mathbf{p}(t-1) \leq \mathbf{H}_{\Delta t} \\
& \mathbf{p}_{min}(t) \odot \mathbf{u}(t) \leq \mathbf{p}(t) \leq \mathbf{p}_{max}(t) \odot \mathbf{u}(t) \\
& -p_{g,lim} \leq p_g \leq p_{g,lim} \\
& \sum_{i=1}^{I} u_i(t) \geq n_{min}, \\
& \sum_{i=1}^{I} p_{i,min}(t) = p_{req} \cdot \left[\sum_{i=1}^{I} u_i(t) - l_{min} \right] \\
& \mathbf{0} \leq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& \mathbf{p} \geq p_{req} \cdot \mathbf{u}(t) - \mathbf{p}_{min}(t) \\
& p_{i,min}(t) \in \{0, p_{req}\} \\
& \sum_{\tau=t-2h}^{t-1} u_i(\tau) \leq 2h \cdot (1 - u_i(t)) + M_{st} \cdot u_i(t-1) \\
& -p_{E,min} \cdot \mathbf{1} \odot \mathbf{u}_E(t) + p_{E,min} \cdot \mathbf{1} \leq \mathbf{p}_E(t) \\
& \mathbf{p}_E(t) \leq p_{E,max} \cdot \mathbf{1} \odot \mathbf{u}_E(t) \\
& -\Phi(t) \leq p_g(t) - p_g(t-1) \leq \Phi(t) \\
& \Phi(t) \geq 0 \\
& -\boldsymbol{\Phi}_{gt}(t) \leq \boldsymbol{p}(t) - \boldsymbol{p}(t-1) \leq \boldsymbol{\Phi}_{gt}(t) \\
& \boldsymbol{\Phi}_{gt}(t) \geq \mathbf{0} \\
& u(t) \in \{0, 1\} \\
& \forall i \in [1, ..., I], \quad \forall j \in [1, ..., J], \quad \forall t \in [1, ..., T]
\end{aligned} \tag{5.5.6}
$$

$$
\begin{aligned}
& E_{min} \leq E_j(t) - p_{E,max} \cdot \frac{\Delta t}{3600} + M_E \cdot (1 - u_{j,E}(t-1)) \\
& E_j(t) - p_{E,min} \cdot \frac{\Delta t}{3600} \leq E_{max} + M_E \cdot u_{j,E}(t-1) \\
& \mathbf{E}(t) = \mathbf{E}(t-1)... \\
& -[\mathbf{p}_E(t-1) \odot \mathbf{u}_E(t-1)... \\
& +r \cdot \mathbf{p}_E(t-1) \odot (\mathbf{1} - \mathbf{u}_E(t-1))] \cdot \frac{\Delta t}{3600} \\
& \forall j \in [1, ..., J], \quad \forall t \in [2, ..., T+1].
\end{aligned}
$$

5.6 Initial states

All the model formulations (5.3.6), (5.4.4) and (5.5.6) presented in this Chapter schedules an optimal output of the generating units given the initial power outputs. The ramp limit constraints and the start-up constraints makes sure that the result is always feasible given the initial states of the generating units.

5.7 Static modeling vs. dynamic modeling

The static model presented in [1] does not take the dynamic of the power system into the consideration and finding the optimal power set-points of the gas turbines at the present time without considering the feasibility of the power that needs to be generated in the future. In another word, the static model does not have the possibility to control the ramp rate of the gas turbines in a way that a feasible optimal solution is always secured. The ramp rate can easily be violated when using the static model since the gas turbines does not allow to be turned on immediately from the off state. It needs 2 hours to get ready to be functional again. The optimal algorithm can suggest to turn off one of the gas turbines only considering the current operational condition, however, after a while, the turned off gas turbine can have to generate power in order to match the power demand constraint. The dynamic model presented in this work will be able to control the gas turbines in an optimal way and secure feasibility of the optimal solutions without violating the ramp rate and power demand constraints if T is chosen reasonably big.

5.8 Time varying parameters

The parameters $\mathbf{p}_{max}(t)$, $C_{el}(t)$ and $p_d(t)$ are time varying. They are constant for 10 seconds, which is the sampling rate. The parameter $p_d(t)$ is predicted using the method presented in Chapter 4 for a given time horizon T. $C_{el}(t)$ is assumed to be constant for the time horizon T. $\mathbf{p}_{max}(t)$ is assumed to be known with complete certainty for the time horizon T.

5.9 The actual cost vs. the objective cost

The actual total cost of operating the site is calculated using (5.4.1). Meaning that the objective costs from the MINLP models presented in this Chapter are not used to calculate the actual total cost. The objective cost is only an artificial cost used to balance the objective function to the model. The cost of operating the ESSs is neglected since the ESS is operating at the expense of the energy from the gas turbines and the grid.

5.10 The overall system architecture

The overall system consists of gas turbines, electrical grid, and ESS. They are connected together to always provide the power required for the site. The cost-efficient distribution of the load is controlled by the EMS based on the models presented in this Chapter.

CHAPTER 6

TOOLS AND ALGORITHM

6.1 Tools

Choosing the suitable programming tools is a critical issue to accomplish tasks and obtaining valuable results. Finding the adequate tools will make it easier to complete some particular tasks. There are many different tools adequate for solving the tasks that are of the same type. Therefore, it can be hard to find a specific one that will perform the best. It requires many trials and time. Due to the limitation of time, the best approach will be trying to find the adequate tools based on the experience and feedback from communities.

6.1.1 TensorFlow

TensorFlow is an open-source software library for data flow programming across a range of tasks. It is a symbolic math library and can be used for machine learning applications. TensorFlow was developed by the Google Brain team primarily for internal Google use. However, it was released as an open-source on November 9, 2015. TensorFlow can run on multiple CPUs and GPUs using Nvidia CUDA and SYCL as the backend.

TensorFlow is an adequate tool for machine learning implementation since it is capable to perform operations on multidimensional data arrays efficiently. The multidimensional arrays are referred as tensors.

TensorFlow framework

TensorFlow library provides Python API, which simplifies the use of TensorFlow. Any TensorFlow program is executable in the same way as Python

programs.

A simple layer neural network can be designed as follows:

```
X = tf.placeholder(tf.float32, [None, inputLen])
W = tf.Variable(tf.zeros([inputLen, outputLen]))
b = tf.Variable(tf.zeros([outputLen]))
Y = tf.nn.sigmoid(tf.matmul(X, W) + b)
optimizer = tf.train.GradientDescentOptimizer(step)
Y_gt = tf.placeholder(tf.float32, [None, outputLen])
crossEntropy = -tf.reduce_sum(Y_gt * tf.log(Y))
trainStep = optimizer.minimize(crossEntropy)
```

The example script presented above makes a single layer neural network graph, which can be executed with the appropriate inputs. Placeholders are the access point of the inputs to the graph. The prepared data for training will be the input of the placeholder called X in this case. The size of the input data can be chosen freely using parameter *inputLen*. W and b are the trainable variables: weight and bias. Y is the output of the single layer neural network. The dimension of the output is controlled by the parameter *outputLen*. The sigmoid function is chosen as the output activation function for this particular case. Gradient decent optimization algorithm is chosen as the training method. The input *step* represents the learning rate. The output of the graph is then compared with the ground truth values using cross-entropy cost function. This cost function is used to determine the accuracy of the trained graph.

As a single layer neural network is designed, the only thing that needs to be done is to feed the input data to the graph and start training.

```
sess = tf.Session()
sess.run(tf.global_variables_initializer())
train_data = {X: bacth_X, Y_gt: batch_Y}
sess.run(train_step, feed_dict=train_data)
```

The first step is to make a session object. A Session object encapsulates the environment in which operation objects are executed, and Tensor objects are evaluated. The session object acts as the main function. Next step is to initialize all the parameters needed for the neural network. This can be done using $tf.global\_variables\_initializer()$ function provided by the TensorFlow library. The final step is to prepare the training data and feed into the graph together with the designed graph.

6.1.2 MILP/MINLP solvers and framework

Selection of the suitable solvers will be a crucial point considering the practical computational time and securing soundness and completeness of the results. Fortunately, there exists a large number of free and commercial solvers adequate for solving MILP and MINLP. The solvers support different frameworks such as MATLAB, C/C++, FORTRAN, JAVA, PYTHON, GAMS, AMPL etc.

MILP

Most of the MILP solvers implement a combination of the cutting-plane and the branch-and-bound algorithm in conjunction with many adjustable heuristics, allowing quite large problems to be solved in a practical computational time.

MILP solvers free for academia:

- CPLEX
- XPRESS
- GUROBI
- MOSEK
- INTLINPROG

Free MILP solvers:

- CBC
- GLPK
- LPSOLVE
- SCIP

All of the solvers listed above can be used with YALMIP and MATLAB.

MINLP

MINLP solvers implement linearization method and provide practical computational time to solve a large range of complex non-linear problems.

MINLP solvers:

- IPOPT
- SCIP
- SNOPT
- KNITRO

Framework

YALMIP is a toolbox used for optimization modeling in MATLAB. Mathematical optimization models can be converted to a high-level algebraic representation using YALMIP in MATLAB. However, YALMIP relies on external solvers for the low-level numerical solution of the optimization problems.

The MILP model from (2.5.1) can be presented in MATLAB using YALMIP as follows:

```
x_c = sdpvar(2,1);
x_I = intvar(2,1);
x = [x_c; x_I];
F = [A*x<=b, x_c >= 0];
h = c'*x;
optimize(F,-h);
```

F presents all the constraints and h is the objective to be maximized. *intvar* restricts the variables to be an integer. By default, *optimize* function will try to minimize the objective, which is the second argument of the function so the negative sign is used to convert the problem to maximization.

6.2 Algorithm structure

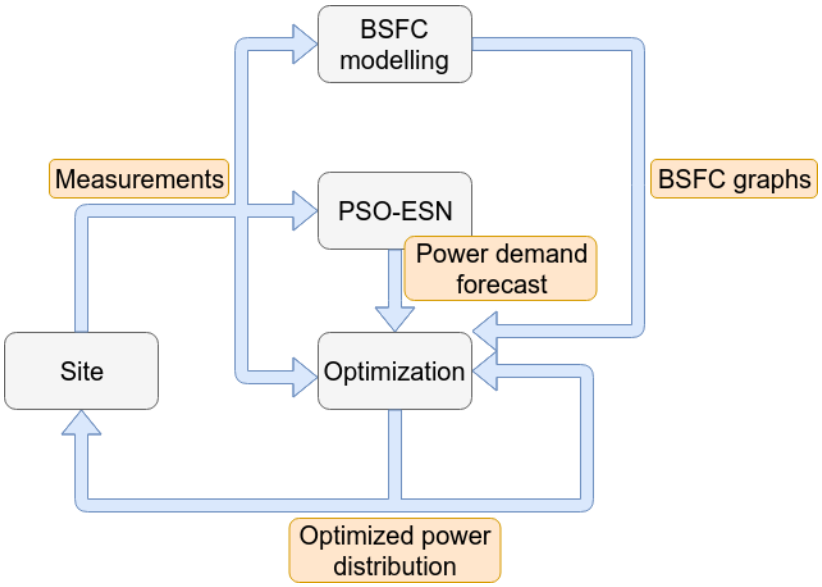


Figure 6.2.1: Structure diagram of algorithm

As shown in Figure 6.2.1, the complete algorithm is divided into three different sub-modules. BSFC modeling, PSO-ESN, and Optimization. The measurements obtained from the site is sent to all of the sub-modules. For the further work, the system modeling module, which is the work conducted in parallel with this work can provide BSFC modeling algorithm and will use the measurements to perform the online update of the BSFC parameters of the gas turbines [3]. However, in this work, only the offline curve-fitting method is used, which means that BSFC modeling will only provide the parameters of the BSFC curves found using the available data-set for the simulation period once and is not updated further. The PSO-ESN module will use the raw data from the measurements to forecast the future power demand. The power demand forecast is then forwarded to the optimization module. In this work, the forecast horizon is chosen to be 30 minutes and as soon as the new power demand measurements corresponding to 30 minutes is ready, PSO-ESN will provide new power demand forecast. In this way, the optimization module will always have the power demand forecast corresponding 30 minutes available. The optimization module will use the measurements, BSFC parameters and the power demand forecast to schedule the optimal load distribution of the generating units. The optimization result is forwarded to the site and is further used as the initial states for optimal scheduling of load distribution for the further computation. This procedure is repeated every 30 minutes (forecast horizon).

CHAPTER 7

RESULTS

7.1 Power demand forecast

The time-series data power demand collected from 7 month of operation from January of 2017 to August of 2017 is partitioned such that 90 % of the data is used to train ESN and 5 % of the data that is used to train is again used as validation data for PSO optimization to find the FV. The rest 10 % of the data is used as test data. The trained ESN models will be used for power demand forecast using the sub-sequential data collected from 3 month of operation starting from August of 2017. The forecast data will be used for the optimization, and the result will be presented in Section 7.2. As explained in Section 4.6, the direct approach has been applied for forecast. The input dimension and the output dimension is chosen after some trials and errors. The recent studies show that long-term forecasts can be quite inaccurate, which was also for this case. If the forecast horizon was increased further to cover a longer period time, the result degraded significantly. Ideally, T should be chosen as big as possible, however, due to the degraded accuracy of the forecast, which will impact soundness of the algorithm when run in realistic industrial environments, the size of T had to be limited. Therefore, 30 minutes is chosen considering the accuracy of forecast and choosing T reasonably big.

Three ESN models are trained to use the same set of 180 sequential historical data as its input to forecast the subsequent 180 data points, which corresponds to the forecast horizon of 30 minutes. Each model is used to forecast a sub-domain corresponding to 10 minutes of data. The first model is used to forecast the first 60 points, which corresponds to first 10 minutes of the forecast horizon of 30 minutes. The second model is used to forecast the subsequent 60 data points and the last model is used to forecast the last

60 data points.

The initial transient response of the ESNs is filtered out by discarding the first 30 % of reservoir states when calculating the readout weights.

For each ESN model, PSO is used for optimizing the global ESN parameters. The PSO particle positions and velocities are updated according to the formula (2.4.1) and (2.4.2). The PSO optimization algorithm is terminated neither after 50 iterations or if the convergence of FV is observed. In total, 70 PSO particles are used. The number of the iterations to be executed and the number of the PSO particles are chosen considering the required time to perform the optimization. However, in order to secure the optimality of the ESN global parameters, further investigation of choosing the number of PSO particles and the number of iterations is required, which is not covered in this work.

Table 7.1: PSO parameters

$Inertial\ \ coefficient : w$	0.8
$Cognitive\ \ component : c_1$	1.8
$Social\ \ component : c_2$	1.8

Table 7.1 shows the key PSO parameters, which is presented in (2.4.1). According to [9], the PSO parameters should be chosen such that $0 \leq w \leq 1.2$, $0 \leq c_1 \leq 2$ and $0 \leq c_2 \leq 2$. w affects convergence of the swarm to optimal and is typically between 0.8 and 1.2 and in general, lower values speed up the convergence. Therefore, 0.8 is chosen as the inertial coefficient. The cognitive component, c_1 decides how big step it takes toward its individual best candidate solution. c_1 is usually close to 2. In this work, 1.8 is chosen as its value. The social component, c_2 is used to define how fast the particles move toward the global best solution and is in general close to 2. For this work, 1.8 is chosen as its value.

The upper and the lower bounds of the parameters to be optimized are as follows:

$$\begin{aligned} Connectivity &\in [0.01, 0.5] \\ Spectral\ \ Radius &\in [0.01, 0.95] \\ Input\ \ Weight\ \ Scale &\in [0.01, 3] \\ Noise\ \ Amplitude &\in [0, 0.5] \\ Leaky &\in [0, 1] \\ Reservoir\ \ Size &\in [180, 2500]. \end{aligned}$$

The bounds are chosen in order to give the PSO algorithm sufficient room

to search for the optimal solution. Moreover, they are chosen based on the previous case studies presented in many ESN related papers. E.g., the spectral radius needs to be less than 1 in order to secure the echo state property of ESN. Considering the upper and lower limit of the sigmoid functions, which is 1 and -1 respectively, the maximum noise amplitude is chosen to be 0.5. The leaky parameter value of 1 transforms the activation function to a classic sigmoid function, hence, the bound is chosen to be between 0 and 1. The upper bound of the connectivity that refers to the sparsity of the reservoir weight matrix is chosen to be 0.5. The minimum reservoir size is chosen based on the input dimension and the maximum is chosen after some trials and errors. The reservoir size bigger than 2500 slowed down the computational performance significantly since it required large memory to be processed.

In addition to defining lower and upper bounds of the global ESN parameters, a constraint is used when searching for the optimal combination using PSO, which yields restricting the leaky parameter from becoming bigger than the spectral radius parameter. This needs to be fulfilled in order to ensure the stability of ESN [30].

The readout weight is trained using (2.3.5) with a very small regularization parameter $\beta : 1e-3$. This will prevent the readout weights from becoming unstable when $\boldsymbol{X}\boldsymbol{X}^\top$ is close to zero. The boundaries of the regularization factor for PSO optimization could not be chosen intuitively, hence, instead of tuning β directly to find the best suitable regularization factor, in this work, a white noise term when updating the reservoir states is added and tuned by the PSO algorithm. The amplitude of this white noise is optimized using PSO. According to [32], the noise size affects the size of the readout weights, hence a similar effect using a regularization term when training the readout weight can be obtained.

The PSO optimization is performed using a computer with the hardware specification:

OS	Ubuntu 16.04 64 bit
CPU	Intel(R) Core(TM) i7-5820K CPU 3.30 GHz
GPU	NVIDIA GeForce GTS 980 Ti, 6144 MB
RAM	16 GB

Table 7.2: Hardware specification

The key performance measurements of the prediction will be Mean Squared

Error (MSE) and Maximum Forecast Error (MFE).

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2$$
$$MFE = max(|y_t - \hat{y}_t|), \quad \forall t \in [1, ..., T].$$

where T is the prediction horizon. MSE is one of the most used quality measurement parameters of a time-series estimator. The quality measurement size is intuitively interpreted and gives a direct insight into the overall precision of the prediction. MFE will be a crucial measurement for this work, since if the prediction deviates too much from the real value, it will be hard to fill the gap of power demand. It requires ramping the power from the gas turbines instantly, which might be infeasible considering ramp rate of the gas turbines. To fill the power demand gap, the power from the grid can also be used and it has more flexibility considering the ramp rate. MFE is sensitive to the outliers hence it does not provide information about the overall performance of the prediction. MSE will, therefore, be used together with MFE to find the model with the best performance.

7.1.1 Power demand forecast using normalized data

The normalized power demand is used to train ESN. The test data is used to evaluate the performance of the trained models and compute the forecast accuracy. The normalization of power demand is performed according to the method presented in Section 4.7.2.

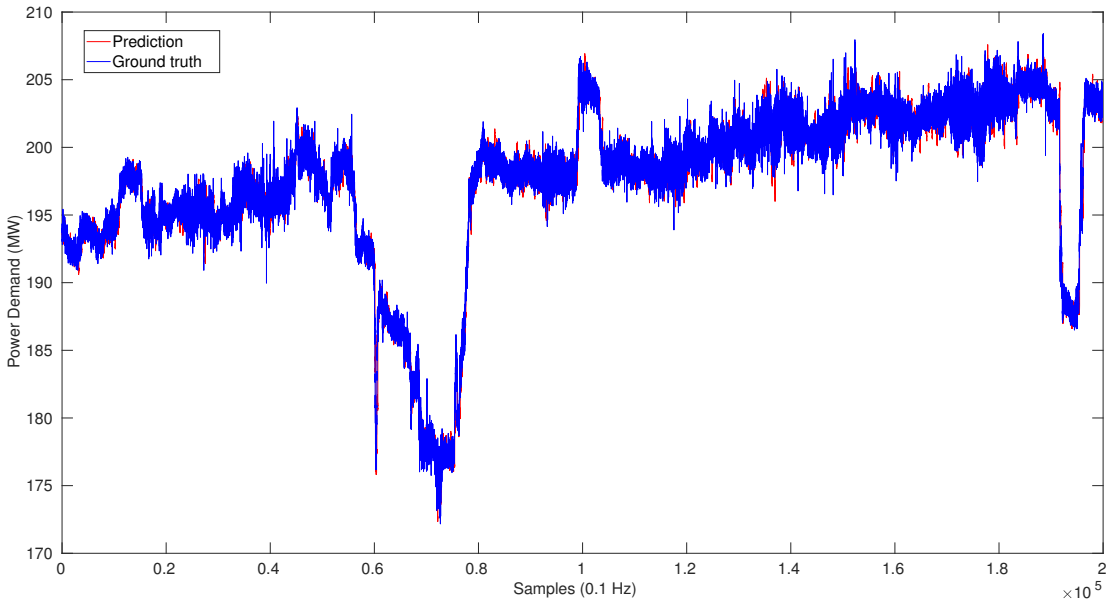


Figure 7.1.1: Power demand forecast using normalized test data

Figure 7.1.1 shows 180-step ahead forecast result using the optimized ESN global parameters presented in Table 7.3. The forecast using normalized power demand gave values between -1 and 1. They are converted back to the original range using the mean value and the standard deviation from the original data. As explained earlier, each set containing 180 data points is used to predict the subsequent 180 data points using three different ESN models. After forecasting a set, the ground truth values of this set and the current reservoir states are used to adjust the next time-step reservoir states as well as forecasting the next set according to (2.3.2) and (2.3.3). This continues until the last test data is predicted. This moving window approach will always make the sub-sequential future power demand corresponding to 30 minutes available. This forecasting strategy will be crucial obtaining the optimization result that will be presented in Section 7.2.

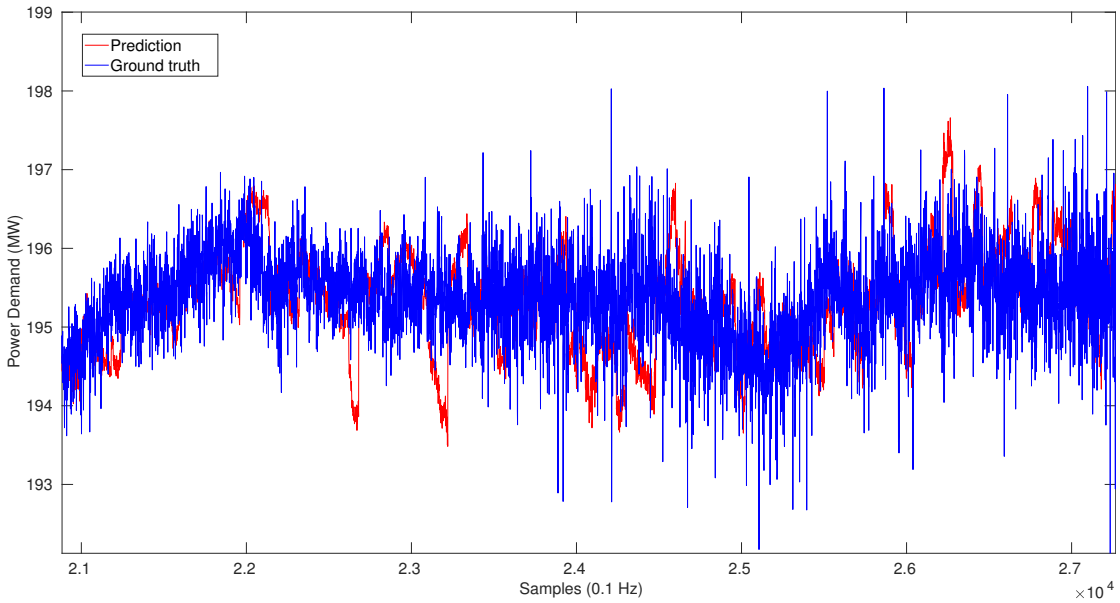


Figure 7.1.2: Power demand forecast covering sample sequence from 21000 to 27000 using the normalized test data

Observing the Figure 7.1.2, it can be seen that the ground truth data is very noisy and the amplitude of this noise is roughly 1 MW. The trained ESN model did not completely follow this noise signal while catching the slow varying trend of the original power demand. There are some jumps observed from the predicted power demand, which is illustrated using red line on Figure 7.1.2, it occurs after each prediction. As the time index of the predicted data-point reaches the end of the prediction horizon of 30 minutes, the accuracy degrades significantly. The jumps are due to the correction of the deviation every-time a new set of power demand is provided to the ESNs and connection between the forecast sets are established.

Table 7.3: Optimized global parameters for three ESN models using the normalized data

Model number	1	2	3
Connectivity	0.01	0.28	0.01
Spectral Radius	0.01	0.01	0.01
Input Weight Scale	1.66	3.0	0.01
Noise Amplitude	0.03	0.02	0
Leaky	1	1	1
Reservoir Size	534	180	180

Table 7.3 shows the optimized global ESN parameters for each model. As

explained earlier, PSO is used for the optimization of the global ESN parameters. The first model is used to forecast the first 10 minutes, the second model is used to forecast the subsequent 10 minutes and the last 10 minutes is forecast by using the last model. It took about 13 hours to finish the PSO optimization utilizing the CPU and the GPU listed in Table 7.2. The PSO optimization only needs to be performed once for each ESN model. Once the ESN global parameters er optimized, the ESN models can be trained within few seconds using those optimized parameters.

The optimized connectivity for the first and the third ESN model yields the lower bound defined. The connectivity parameter refers to the sparsity of the reservoir layer. Sparsity affects the computational effort. The required time to train a model is decreasing as the sparsity reduces. However, the PSO algorithm designed in this work does not take the computational effort into the consideration. Hence, the computational time does not affect the optimization of the sparsity of the reservoir layer. Moreover, the sparsity parameter does not affect the performance of the forecast significantly [34]. It is therefore unclear how the sparsity parameters are optimized by PSO.

Optimization result using PSO yields the smallest possible spectral radius for all of the ESN models. Low spectral radius will result fast die out of the past inputs. In another word, the past inputs will not have too much influence on the forecast. Hence, the interpretation of the choice can be that the ESN models do not need to *remember* the past inputs and do not depend so much on the historical data, which is not a part of the input.

Large input weight scale size indicates that the dynamic is highly influenced by the input data. For the second ESN model, the maximum size is chosen. Meanwhile, the minimum size is chosen for the third ESN model. This can be interpreted as a lack of correlation between the input (historical data) and the output (forecast) for the third ESN model, which is as expected. Since, the output of the third ESN model corresponds to the power demand in a distant future.

Generalization and stability of ESN are the two factors that can primarily be controlled by injecting an artificial noise into the reservoir states. The PSO optimization yields, a small size of noise for the ESN models. A possible interpretation of this observation can be that since power demand is not low-pass filtered, the data is highly affected by different types of noise. Generalization might not be an issue when using noisy data.

The PSO optimized leaking rates indicate that power demand does not contain any slow dynamic hence no need for leaky integrator neurons. Using 1 as the leaky coefficient is same as using a simple sigmoid network, which

can be observed from (2.3.2).

The reservoir size represents the complexity of ESN. Highly non-linear dynamic requires a large number of reservoir neurons. The reservoir size indicates that the highest order of complexity is required for the first ESN model. The minimum reservoir size is chosen for the second and the third model. This is unexpected since the third model is considered to perform a relatively harder task, which requires a more complex model. This can be an indication that the number of iteration of the PSO algorithm needs to be increased to ensure convergence of the FV. The algorithm can be terminated before a sufficient convergence was achieved.

Table 7.4: Mean Squared Error of the forecast using the normalized data

Model number	1	2	3
MSE	0.404	0.845	1.493

Table 7.4 shows MSE of the forecasts. Catching the dynamic of the power demand in the nearest future is easier compared to catching the dynamic of the data in the distant future. Hence, the forecast error obtained using the first ESN model is smallest compared to the second and the third model. The forecast result yields the average MSE of 1 and MFE of 9.63.

7.1.2 Power demand forecast using low-pass filtered and normalized data

The normalized power demand is filtered using 4^{th} order Butterworth filter. The frequency response of the filter is shown in Figure 4.7.2. The filtered power demand can be found in Figure 4.7.1.

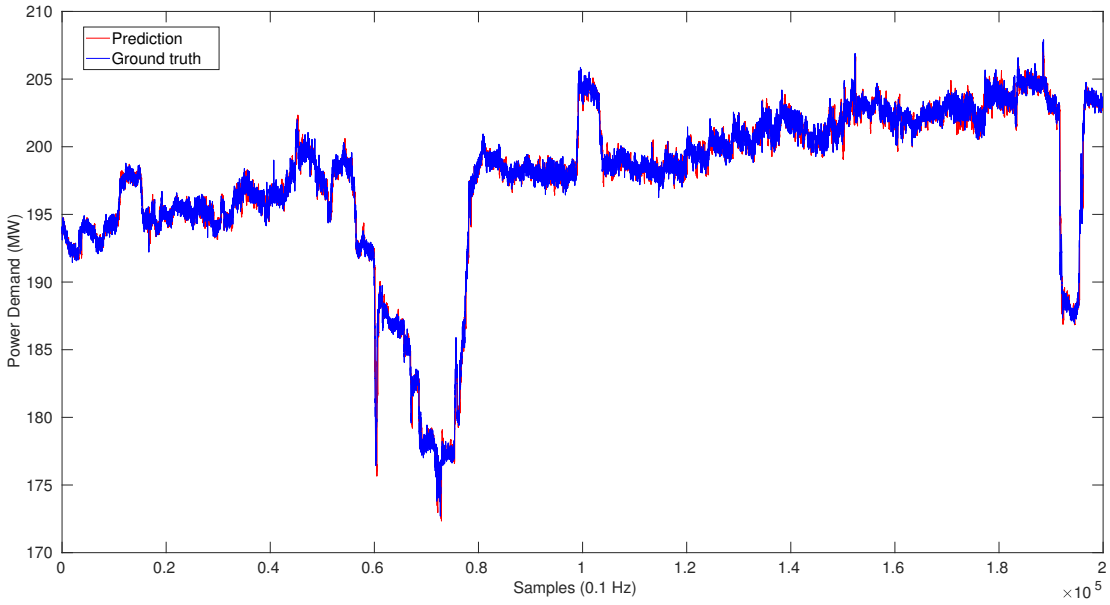


Figure 7.1.3: Power demand forecast using the low-pass filtered and normalized test data

Figure 7.1.3 shows forecast results using the optimized global ESN parameters from Table 7.5 for three ESN models designed for 180-step ahead forecast. The difference from the forecast procedure presented in Section 7.1.1 is that the data is pre-processed using a 4^{th} order Butterworth filter. Overall, the difference between the forecast power demand and the ground truth power demand is significantly small compared to the result from Section 7.1.1.

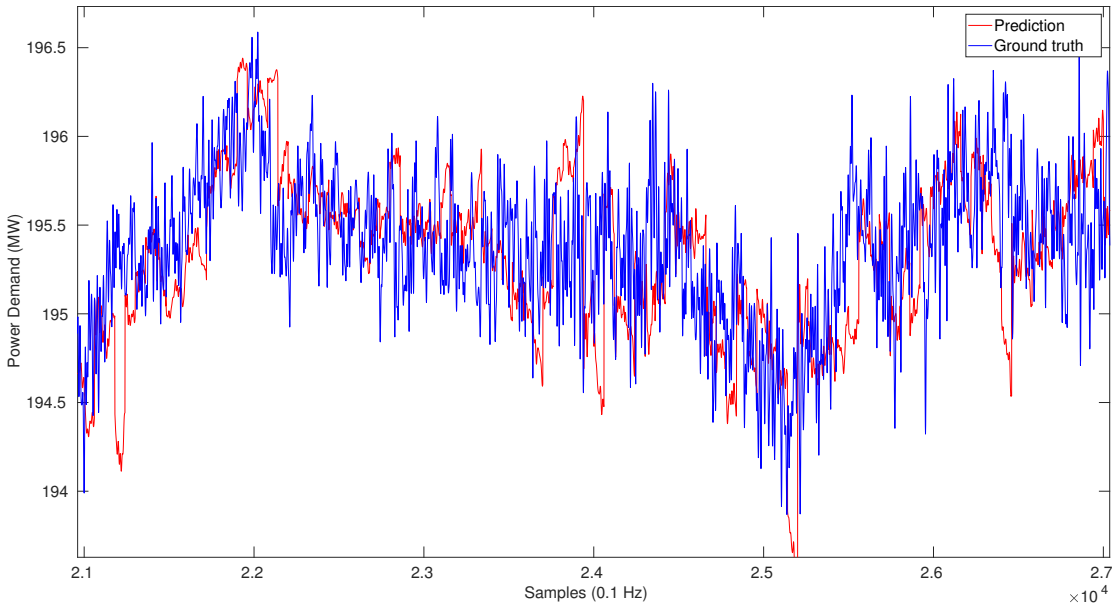


Figure 7.1.4: Power demand forecast covering sample sequence from 21000 to 27000 using the low-pass filtered and normalized test data

The accuracy of forecast can be more closely observed from Figure 7.1.4. As it is shown, the time-series estimation models are not able to follow the ground truth values completely, however, the squared error between the forecast values and the ground truth values are very small. There are some sudden jumps observed from the forecast, this is observed at the end of the forecast horizon of a set. The big jumps indicate a relatively large forecast error of the last forecast interval. Overall, the forecast do not deviate too much from the ground truth values.

Table 7.5: Optimized global parameters of the ESN models using the low-pass filtered and normalized data

Model number	1	2	3
Connectivity	0.16	0.44	0.50
Spectral Radius	0.01	0.28	0.23
Input Weight Scale	3	3	3
Noise Amplitude	0	0.5	0
Leaky	0.60	1	1
Reservoir Size	180	2500	647

Table 7.5 shows the optimized global parameters of the ESN models. It took about 13 hours to finish the PSO optimization utilizing the CPU and the GPU listed in Table 7.2.

The sparsity of the reservoir reduces with the model number. High sparsity benefits in a way that less computational effort is required for network state update. However, as mentioned, the PSO algorithm designed in this work does not take this into account, so the connectivity will be chosen randomly respect to the computational effort. It is also unclear how the connectivity can affect the forecast result. Many ESN authors claimed that sparsity will lead decomposition of reservoir dynamics into loosely coupled subsystems, which lead to large variation among the reservoir signals. However, the fully connected reservoirs have also been proved to work as well as the sparsely connected reservoirs.

Optimization result using PSO yields a smallest possible spectral radius of the first model. This indicates that the past inputs will not have too much influence on the prediction. As it can be observed, the spectral radius increases to 0.28 for the second ESN model. This indicates that forecasting a more distant future is more dependent on the past inputs.

The maximum value of the input weight scale is chosen for all of the ESN models using the PSO optimization. Considering the update stage of ESN shown in (2.3.2), the reservoir states are highly influenced by the input data. Intuitively, this can be an indication that the best "qualified" information that can be used for prediction is the most recent historical data.

The PSO optimization yields, zero noise on the first and the third ESN models, meanwhile, noise with the biggest amplitude is used for the second model. A possible explanation of this choice can be that the second model is very easily over-fitted. The stability of the model is most likely not an issue in this case since output feedback (2.3.6) is not used updating the reservoir states.

The PSO optimized leaking rates indicate that power demand does not contain any slow dynamic for the second and the first ESN models, hence no need for leaky integrator neurons. However, the first ESN model is using 0.6 as its leaking rate. This can be an indication of the existence of a slow dynamic for the first 10 minutes of forecast. Information from the past reservoir states is retained using leaky integration. In this way, capturing slow dynamics.

The PSO optimization yields smallest possible reservoir size for the first model, larger number of reservoir neurons for the second and third models, which is as expected since forecasting a more distant future is a harder task.

Table 7.6: Mean Squared Error of the forecast using the low-pass filtered and normalized data

Model number	1	2	3
MSE	0.16	0.443	0.748

Table 7.6 shows MSE of the forecasts. Catching the dynamic of power demand in the nearest future is easier compared to the dynamic in the distant future. Therefore, forecast using the first model has the smallest MSE and increases with the model number, which is as expected. The forecast result yields the average MSE of 0.489 and MFE of 7.4.

7.1.3 Increasing complexity by concatenating element-wise squared input vector to the original input vector

The input vector is expanded to include the squared version input. Such that the input vector becomes

$$\boldsymbol{u}_{squared} = [\boldsymbol{u}; \boldsymbol{u} \odot \boldsymbol{u}],$$

where $\odot$ is used to denote element-wise multiplication. The dimension of the input vector is then doubled. It leads to the changed dimension of the input weight matrix and the readout matrix along with the input vector. Moreover, (4.5.1) is using the squared version of input vector, so (4.5.1) becomes.

$$\boldsymbol{X} = [\mathbf{1}; \boldsymbol{u}_{squared}; \boldsymbol{x}],$$

Forecasting the future power demand is performed using the low-pass filtered normalized data. The filtered data is obtained using 4^{th} order Butterworth filter with the frequency response as shown in Figure 4.7.2.

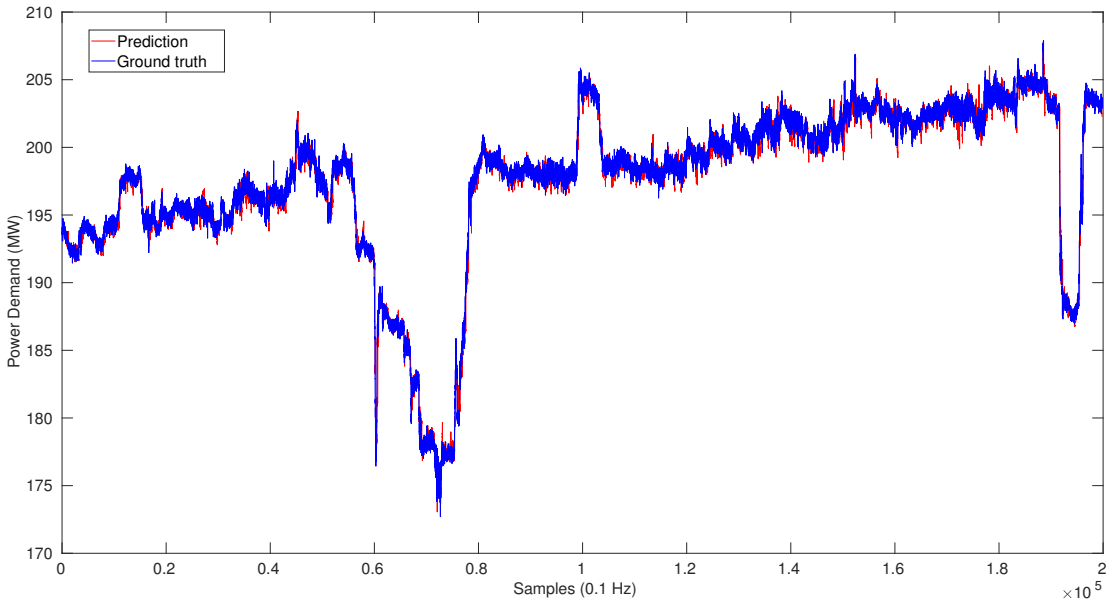


Figure 7.1.5: Power demand forecast using the squared version of input

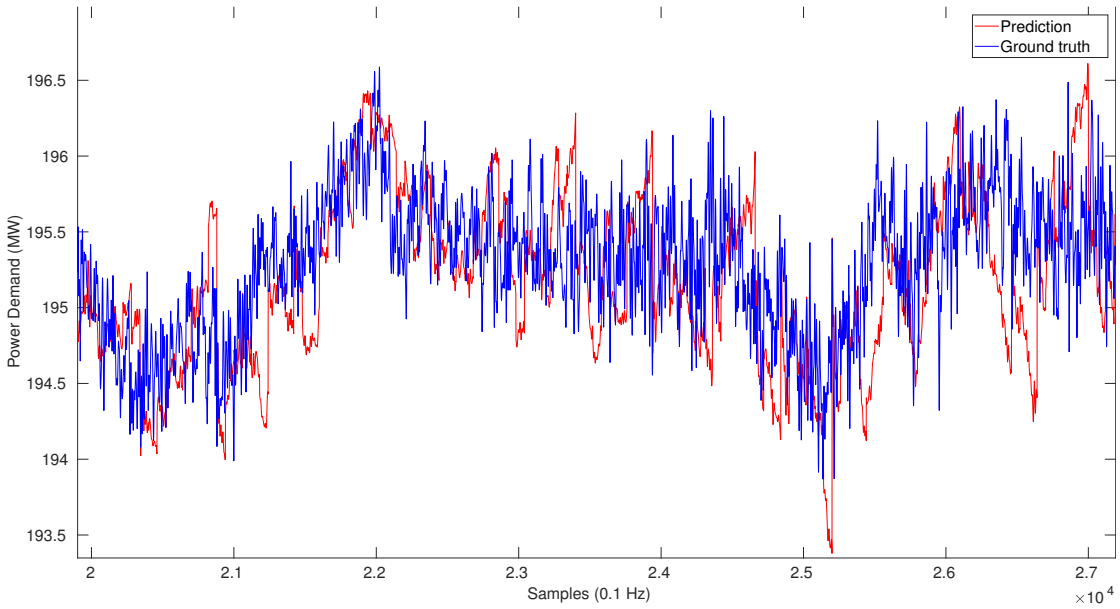


Figure 7.1.6: Power demand forecast covering sample sequence from 21000 to 27000 using the squared version of input vector

Figure 7.1.5 shows 180-step ahead forecast result using the optimized ESN global parameters presented in Table 7.7. Observing from Figure 7.1.6, there

are some jumps observed from the forecast time-series data similar to the results presented in Section 7.1.1 and Section 7.1.2. The jumps occur at the end of the forecast interval, in another word, right before preceding to the next forecast sample containing the subsequent forecasts. Except for the few jumps, in overall, the forecast is quite accurate and the forecast error is very small. The forecast did not catch the high-frequency changes of the ground truth data. Depending on the application, this might be the desired behavior. However, in this work, catching the high-frequency changes are not desired. Since the high-frequency changes are considered as noise.

Table 7.7: Optimized global parameters of the ESN models using the squared input vector

Model number	1	2	3
Connectivity	0.01	0.01	0.42
Spectral Radius	0.01	0.01	0.73
Input Weight Scale	2.57	0.01	1.22
Noise Amplitude	0	0	0
Leaky	1	1	0.84
Reservoir Size	180	180	180

Table 7.7 shows the optimized ESN global parameters. Each ESN model forecasting different non-overlapping time intervals. It took about 13 hours to finish the PSO optimization utilizing the CPU and the GPU listed in Table 7.2.

Connectivity remains at the lowest number for the first and the second model and increases to 0.42 for the last model. As explained in Section 7.1.1 and in Section 7.1.2, how the connectivity parameters are chosen cannot be explained intuitively. However, this is not a required task, since PSO will make it possible to optimize the global ESN parameters without having knowledge about how they affect the forecast accuracy.

The smallest number of the spectral radius is chosen by PSO for the first two models. Spectral radius increases to 0.72 for the last model. The forecast task of the last model is more demanding compared to the first and the second since it needs to forecast power demand in a more distant future, therefore it requires to *remember* a longer period of time.

The optimized input weight scale factor is quite randomly chosen and does not show any trend between the three models. The first model clearly emphases the input signals, which is the most recent historical power demand. In another word, prediction of the first interval depends relatively highly on the input signals.

Injecting artificial noise was not required for this case. The best guess based on the intuition is that the squared version of input vector prevents over-fitting and therefore using artificial noise was not required and did not increase the forecast performance.

Optimized leaking rate indicates non-existence of slow dynamic for the first and the second ESN models, hence no need for leaky integrator neurons. However, based on the PSO optimization, it can be guessed that there are some slow dynamics observed for the last ESN model, hence requires the leaky parameter to be less than one.

The PSO optimization yields the smallest possible reservoir size. Compared to the previous sections, the reservoir size is considerably small. The complexity is increased by expanding the input vector and some part of the non-linear dynamic is added by the squared input vector, resulting in a smaller reservoir size. This can benefit from the reduced computational expense.

Table 7.8: Mean Squared Error of the forecast using the squared input vector

Model number	1	2	3
MSE	0.178	0.458	0.917

Table 7.8 shows MSE of the forecasts. Using the forecast result, the average MSE is found to be 0.56 and MFE is found to be 9.16. Compared to the results discussed in Section 7.1.2, the performance has been decreased significantly.

7.1.4 Increasing model complexity by using the squared readout weight

The readout weight is modified by concatenating the squared version of the input data and the reservoir states when training the readout weight. Such that (4.5.1) is modified to

$$\boldsymbol{X} = [\mathbf{1}; \boldsymbol{u}; \boldsymbol{u} \odot \boldsymbol{u}; \boldsymbol{x}; \boldsymbol{x} \odot \boldsymbol{x}],$$

where $\odot$ denotes element-wise multiplication. This is a cheap way to increase complexity instead of increasing the size of reservoir states such that the input and the reservoir states can more easily map to the outputs.

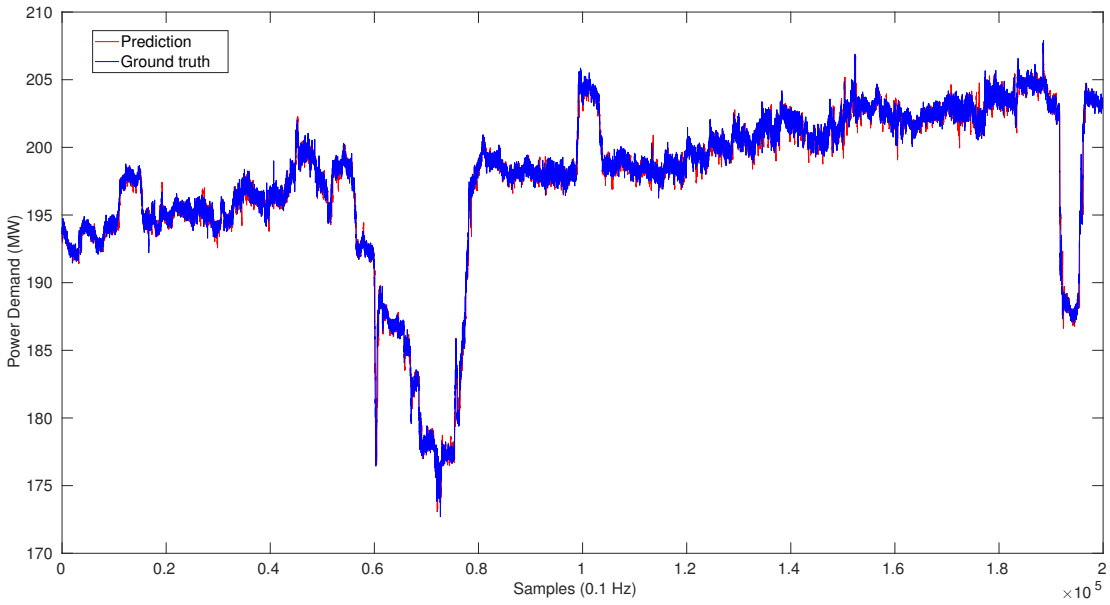


Figure 7.1.7: Power demand forecast using the squared version of trained readout weight

The normalized power demand is filtered using 4^{th} order Butterworth filter. The filtered power demand is shown in Figure 4.7.1. The forecast result is shown in Figure 7.1.7. The forecast result is obtained using the optimized ESN global parameters presented in Table 7.9.

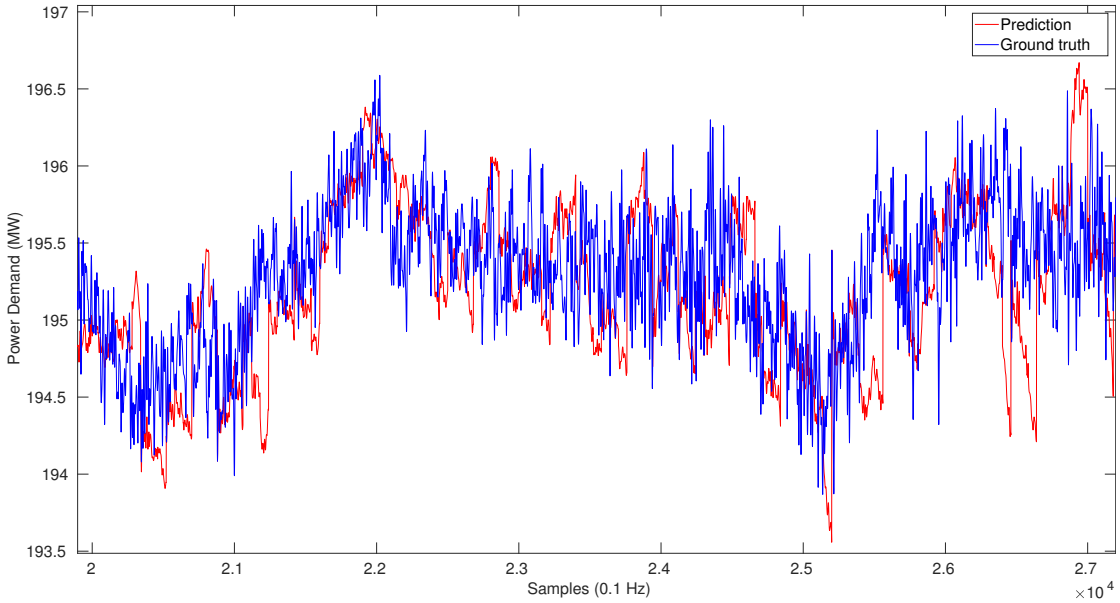


Figure 7.1.8: Power demand forecast covering sample sequence from 21000 to 27000 using the squared version of trained readout weight

Similar to the forecast results presented in Section 7.1.2 and 7.1.3. There are still some jumps observed from the forecast time-series data. Moreover, forecast does not catch the high-frequency changes completely.

Table 7.9: Optimized global parameters of the ESN models using the squared readout weight

Model number	1	2	3
Connectivity	0.5	0.01	0.01
Spectral Radius	0.01	0.01	0.01
Input Weight Scale	1.15	0.01	0.01
Noise Amplitude	0.07	0	0
Leaky	1	1	1
Reservoir Size	180	180	180

Table 7.9 shows the optimized global parameters for the ESN models using PSO. It took about 14 hours to finish the PSO optimization utilizing the CPU and the GPU listed in Table 7.2.

The smallest number is chosen as the connectivity for the second the third model. However, for the first model, the biggest number is used as the connectivity parameter. 0.01 indicates high sparsity of the matrix.

An interesting observation is that PSO chose the smallest possible spectral radius for all of the ESN models. To recall, this indicates that the past input data will not have too much of impact on the predictions.

The input weight scale indicates that the model used for forecasting the first 60 sequential data points is highly influenced by the input data, which makes sense since they are the "nearest" points considering how the data is partitioned. The first ESN model is using 180 data-points to predict the sub-sequential 60 data-points with $t \in [0, ..., 60]$. However, the second ESN model which maps the same input to the data with $t \in [61, ..., 120]$ is much less influenced by the input data. Same goes for the third ESN model forecasting power demand with $t \in [121, ..., 180]$.

The intuitive explanation of the choice PSO made is that the first model forecasting the first interval is more sensitive to over-fitting since a very small forecast error from the training data can be obtained. The size of the white noise injected into the reservoir states is, therefore, larger compared to the second and the third model.

The values of the leaky parameter indicate that power demand does not contain any slow dynamic, hence no need for leaky integrator neurons.

As the complexity of ESN is increased using a squared variant of the readout weight, the reservoir size does not need to be large. PSO chose the smallest possible reservoir size, which is as expected.

Table 7.10: Mean Squared Error of the forecast using the squared readout weight

Model number	1	2	3
MSE	0.11	0.27	0.83

Table 7.10 shows MSE of the forecasts for each ESN model. The forecast result yields the average MSE of 0.506 and MFE of 7.067.

7.2 Optimization

The MINLP models presented in Chapter 5 is implemented and the simulation is performed using:

1. the data extracted from the five different gas turbines operating on a site
2. the import/export history of the grid power
3. price history of the grid electricity

The optimal power set-points are found using the EMS algorithm every 30 minutes. Each execution of the EMS algorithm results in 18 optimal set-points, which means that the EMS algorithm gives optimal set-points every 100 seconds. The EMS algorithm is implemented in MATLAB using YALMIP as its framework. SCIP is chosen as the MINLP solver.

The approach presented in Section 7.1.2 is used to forecast the future power demand. Meaning that the raw power demand data is normalized and low-pass filtered using 4^{th} order Butterworth filter. No modification has been performed on (4.5.1) for designing the ESN models. The predicted power demand is used as a set of parameters for the optimization algorithm to schedule the optimal power output from the generating units.

Table 7.11: Mean Squared forecast Error and Maximum Forecast Error for the prediction approaches presented in Section 7.1

Sections	7.1.1	7.1.2	7.1.3	7.1.4
MSE	1	0.489	0.56	0.506
Max error amplitude	9.63	7.41	9.16	7.067

Table 7.11 presents the error size using different approaches. The approach from Section 7.1.2 has the lowest MSE and the lowest MFE is achieved by using the approach from Section 7.1.4. However, the method presented in Section 7.1.4 requires larger memory and extra calculation compared to the method from Section 7.1.2. Moreover, the error difference seems to be minor, hence the result will not be affected so much due to the model choice. MFE of the ESN model presented in Section 7.1.2 is small enough to be corrected by the operating gas turbines. Hence, it is expected that the power demand gap due to the forecast error can always be filled by changing the power output of the gas turbines instantly.

7.2.1 Uncertainties

The results are affected by some uncertainties. The price of the gas is assumed to be fixed, however, this will vary depending on the season and gas

quality. The result will also depend on the accuracy of the data achieved from the site. The data will include uncertainties due to the measurement noise in addition to some other noise factors.

The uncertainty due to the power demand forecast error is minor considering MSE. In normal circumstances, the magnitude of power demand will be around 200 MW and an MSE value of 0.489 will indicate that the power demand forecast will give approximately 10 MW of summed deviation for the time horizon T, which can be calculated using

$$\sum_{t=1}^{T} y_t - \hat{y}_t = \sqrt{MSE \cdot T}.$$

Given that the site requires approximately 200 MW for each time instance, it will give the total power demand of 36000 MW for the given time horizon T ($200\ \ MW \cdot T$). 10 MW is a very small number (36000 MW >> 10 MW) compared to the power level that is required by the site. Hence, the optimal power set-points scheduled by the EMS algorithm using the forecast power demand will be valid. Considering MFE, the power demand prediction error can in most of the cases be controlled by adjusting the total power production by ramping the gas turbines instantly.

The maximum power the gas turbines can generate varies with time. In this work, the maximum power of the gas turbines will be assumed to be known for the given time horizon T with complete certainty. This assumption is valid as long as T is chosen reasonably short. The maximum power is calculated in a special way using different types of parameters such as temperature, hours of operation etc. that does not vary drastically on a short period of time.

The electricity price is assumed to be fixed for T. The electricity price data is achieved from $Nord\ \ Pool\ \ AS$. The price varies on an hourly basis. Since the forecast horizon T is chosen to be less than 1 hour in this work (30 minutes), the electricity price can be assumed to be fixed without leading to any significant uncertainty in the result.

7.2.2 Simulation results

C_{fuel}	1.926 $\frac{NOK}{kg}$
w_{Φ}	120
$w_{\Phi,g}$	0.02
n_{min}	4
l_{min}	4
p_{req}	35 MW
$L_{\Delta t}$	-3.5 $\frac{MW}{\Delta t}$
$H_{\Delta t}$	3.5 $\frac{MW}{\Delta t}$
$p_{g,lim}$	115 MW
$p_{g,sp}$	0 MW
$M_{rp,\Delta t}$	5 $\frac{MW}{\Delta t}$
M_{st}	2 hours

Table 7.12: Constant parameters

The non-changing parameters listed in Table 7.12 are used for the implementation. The parameters such as $p_{max}(t)$, $C_{el}(t)$ and $p_d(t)$ are not listed in the table since they are time-varying.

The simulation is performed using computers with the following hardware specification:

OS	Windows 64 bit
CPU	Intel(R) Core(TM) i7-5820K CPU 3.30 GHz
GPU	NVIDIA GeForce GTS 980 Ti, 6144 MB
RAM	16 GB

Table 7.13: Hardware specification of the 1^{st} computer

OS	Windows 64 bit
CPU	Intel(R) Core(TM) i5-6300U CPU 2.40GHz
RAM	8 GB

Table 7.14: Hardware specification of the 2^{nd} computer

Power demand forecast

The approach presented in Section 7.1.2 has been applied to forecast the power demand for the simulation period starting from August of 2017.

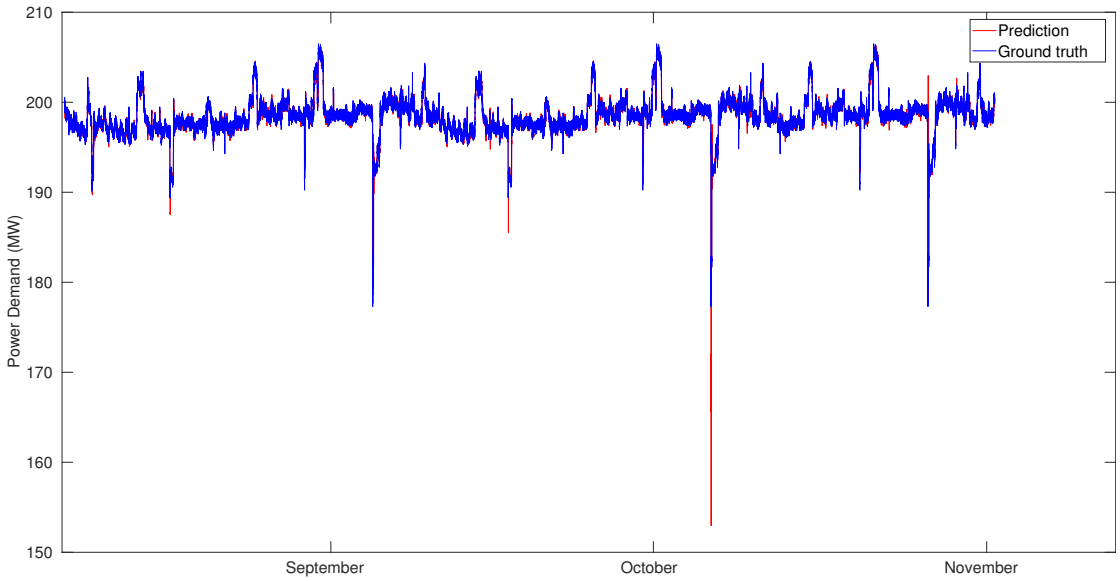


Figure 7.2.1: Power demand forecast between the period August of 2017 and November of 2017

Figure 7.2.1 shows the forecast result for the simulation period. The ESN models used for the forecast is trained again after each month of simulation using the training data as well as the test data corresponding to the passed simulation period. This benefits in terms of increasing the accuracy of forecast and is practical, since the training process only takes few seconds to finish. The average MSE from the 3 months of simulation is found to be 0.4 and MFE is found to be 21.4. MFE is higher than expected. 21.4 MW corresponds to approximately 50 % of the max power level a gas turbine can produce. Fortunately, there are five gas turbines, which can be used to fill the power demand gap. 21.4 MW can be shared between the operating gas turbines and given that all the gas turbines are operable, each gas turbine only needs to increase their power level by 4 MW.

Offline trained BSFC curves

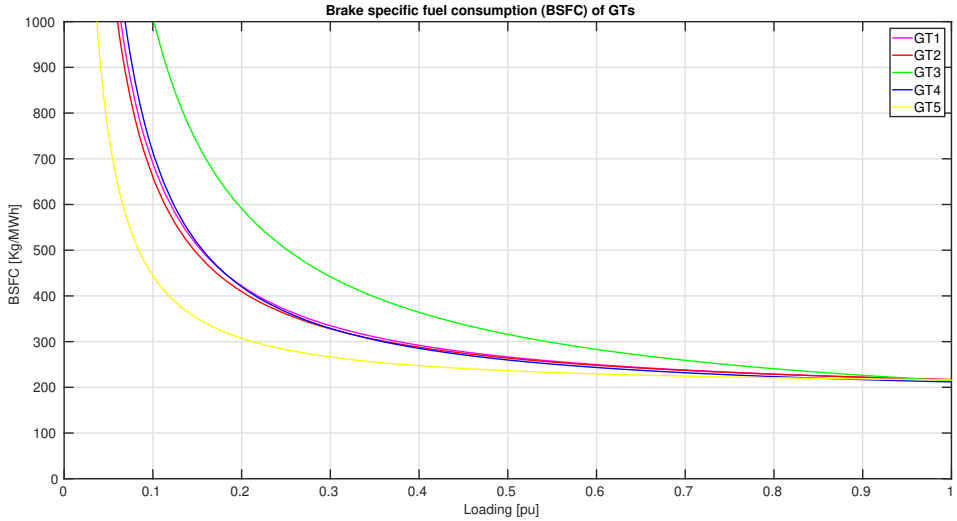


Figure 7.2.2: Brake Specific Fuel Consumption of the gas turbines normalized relative to the per unit loading

The BSFC of the gas turbines are assumed to follow the curves in Figure 7.2.2. They are found using curve-fitting on the data recorded from the gas turbines for the simulation period from August of 2017 to November of 2017. They are not updated continuously and is assumed to be fixed. The further improvement of BSFC offline modeling can be found in [2].

Cost optimization

The simulation is performed using the model (5.4.4) and the data from 3 months of operation starting from August of 2017. The simulation is performed utilizing the CPU listed Table 7.14 and it took about 5 days to finish the simulation. The optimized fuel consumption was calculated from the optimized gas turbine utilization. The result of the simulation indicates the cost saving of 2,883,237 NOK compared to the original cost, which corresponds to 1.9 % reduction.

Table 7.15: Saving results

	Fuel consumption (Kg)	Electricity cost (NOK)
Original	73,997,000	10,111,700
Optimal	72,339,000	10,421,000

Table 7.15 shows the total fuel consumption and the total electricity cost. The difference of the fuel consumption and the electricity cost between the

original data and the simulation data is found to be 2.24 % and -3.06 % respectively. By tuning the parameter explained in (5.4.3) ($w_{\Phi,g} = 0.02$), and having the grid set-point parameter equal 0 ($p_{g,sp} = 0$), the amount of the grid power that is imported could be controlled to match the original level. The result presented in [1] indicates that it could be beneficial to import more power from the grid in order to reduce the total expense and will be one of the easiest ways to save fuel and cost of operation. By raising the grid import/export set-point ($p_{g,sp}$), more power is allowed to be imported from the grid and a significant amount of operational cost can be reduced. However, it also increases the risk in case the grid fails. In another word, high grid dependency will decrease the robustness of the power system installed at the site. Therefore, the import level of the grid power needs to be controlled carefully in order to secure the soundness of the optimized operation when running in realistic industrial environments.

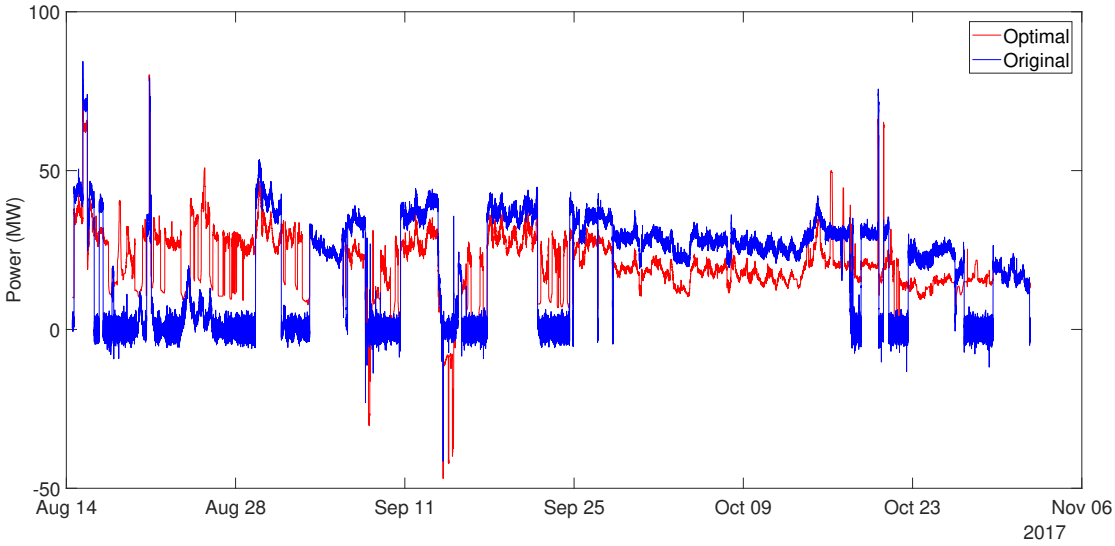


Figure 7.2.3: Optimal vs. original grid power

Figure 7.2.3 shows power from the grid. For the period where a relatively low amount of power is imported in the original case, the EMS decides to import considerably more power from the grid. However, during the period where a large amount of power is imported from the grid, the EMS decides to import relatively less power. Overall, the total amount of power that is imported is almost same as the original case.

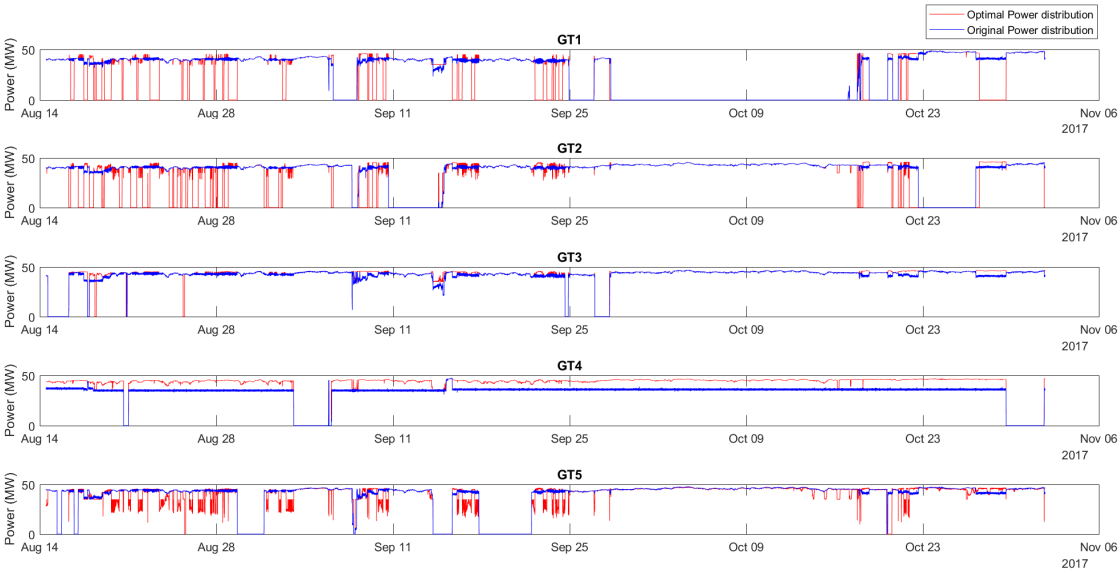


Figure 7.2.4: Optimal vs. original power distribution

Figure 7.2.4 shows the power distribution between the gas turbines. The optimal power distribution shows the set-points to be followed by the low-level controller (constant for 100 seconds), meanwhile, the original power is recorded directly from the low-level controller. It is observed that the EMS controls the gas turbines differently than in the original case considering set-point regulation. GT1 and GT2 are turned on and off more often between the period 14^{th} of August and 11^{th} of September. The power pattern of GT3 resembles the original case. GT4 provides the maximum power it can to the site for most of the time, which differs from the original case. The power level of GT5 changes more drastically compared to the original case. Considering the BSFC curves, it is as expected since GT5 has a higher efficiency when operating with a relatively low loading.

Cost optimization with 5 MW as the grid power set-point ($p_{g,sp}$)

The simulation is performed using the model (5.4.4) and the data from 3 months of operation starting from August of 2017. The simulation is performed utilizing the CPU and the GPU listed Table 7.13 and it took about 4 days to finish the simulation. The grid power set-point was increased from 0 MW to 5 MW to observe the impact of changing the grid power set-point. The result of the simulation indicates the cost saving of 3,804,169 NOK compared to the original cost, which corresponds to 2.47 % reduction.

Table 7.16: Saving results with $p_{g,sp}$: 5 MW

	Fuel consumption (Kg)	Electricity cost (NOK)
Original	73,997,000	10,111,700
Optimal	71,398,000	11,495,000

Table 7.17 shows the total fuel consumption and the total electricity cost using 5 MW as the grid power set-point ($p_{g,sp} = 5$). The difference of the fuel consumption and the electricity cost between the original data and the simulation data is found to be 3.5 % and -13 % respectively. By increasing the grid power set-point parameter from 0 MW to 5 MW, more power was imported from the grid. As expected, by allowing more power to be imported from the grid, the total expense of operating the site could be reduced further. By spending 1,074,000 extra NOK on the grid power and burning less fuel, the total expense could be reduced by 920,939 NOK. However, as mentioned in Section 7.2.2, more grid power increases the risk in case the grid fails. It is therefore desired to see how ESSs can be embedded to the optimization model, which can be used to reduce the risk of operating the site.

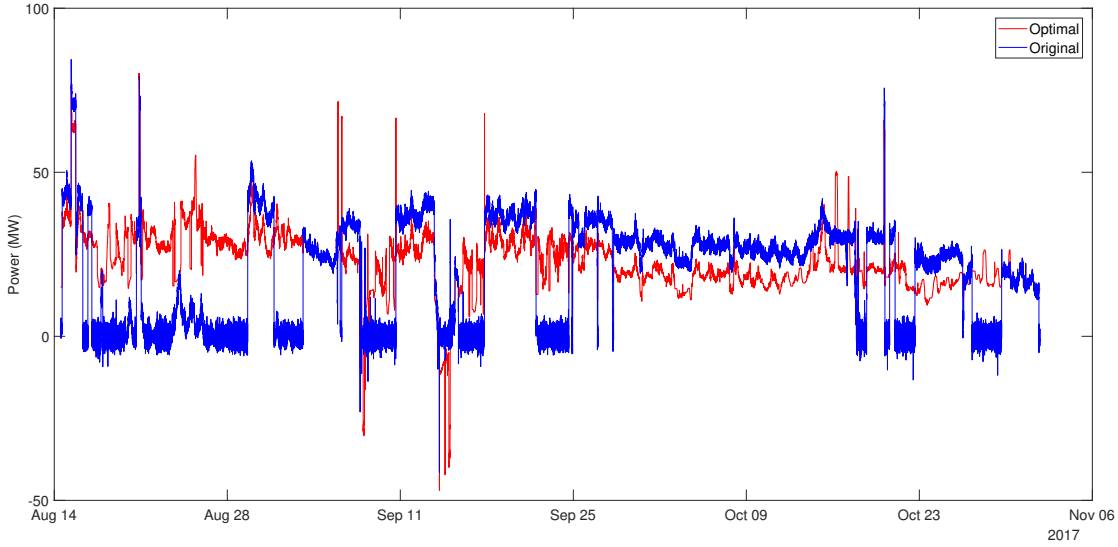


Figure 7.2.5: Optimal vs. original grid power with 5 MW as the grid power set-point

Figure 7.2.5 shows power from the grid. Similar to the behavior of the grid observed from Section 7.2.2, for the period where a relatively low amount of power is imported in the original case, the EMS decides to import considerably more power from the grid. However, during the period where a

large amount of power is imported from the grid, the EMS decides to import relatively less power.

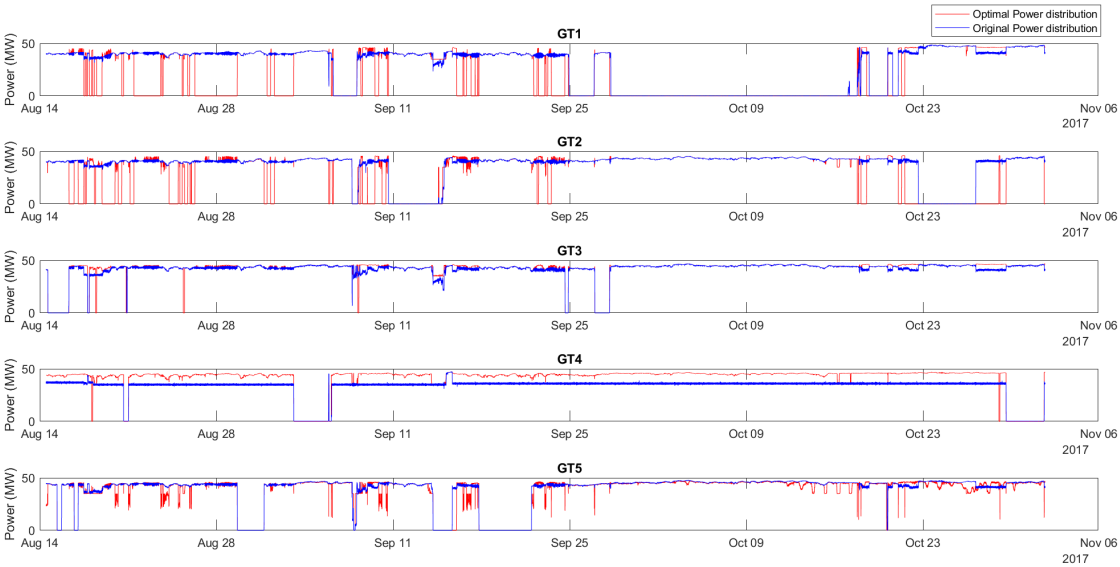


Figure 7.2.6: Optimal vs. original power distribution with 5 MW as the grid power set-point

Figure 7.2.6 shows the power distribution between the gas turbines. GT1 and GT2 are turned on and off more often between the period 14^{th} of August and 25^{th} of September. Moreover, GT1 and GT2 tend to provide the maximum power it can to the site, which is optimal considering the efficiency curves. The power pattern of GT3 resembles the original case. GT4 provides the maximum power it can to the site for most of the time, which differs from the original case. The power level of GT5 changes more drastically compared to the original case. Observing Figure 7.2.2, GT5 has relatively high efficiency operating at the minimum load compared to the other GTs, therefore, GT5 is used to match power demand by reducing the power output to the non-optimal points considering the efficiency curves.

Cost optimization integrating ESS with 5 MW as the grid power set-point ($p_{g,sp}$)

The parameter w_Φ presented in Section 5.4.2 is increased to

$$w_\Phi = 540,$$

in order to make use of ESSs to regulate the small and high-frequency changes of power demand. w_Φ is tuned using trial and error method. The amplitude of this parameter was increased if the power output of the gas turbines and the grid tends to shift aggressively. Moreover, the recovery ratio of the ESS is chosen to be 90 %.

Simulation is performed using the model presented in (5.5.6). The ESS parameters from Table 5.1 is used for the simulation. The start capacity of the ESS is chosen to be 32 MWh, which is the max capacity of the ESS. The simulation is performed using the data from 3 months of operation starting from August of 2017. The simulation is performed utilizing the CPU listed Table 7.14 and it took about 7 days to finish the simulation. The optimized fuel consumption was calculated from the optimized gas turbine utilization. The result of the simulation indicates the cost saving of 3,783,058 NOK compared to the original cost, which corresponds to 2.48 % reduction.

Table 7.17: Saving results integrating ESS with $p_{g,sp}$: 5 MW

	Fuel consumption (Kg)	Electricity cost (NOK)
Original	73,997,000	10,111,700
Optimal	71,137,000	11,837,000

The operational cost is reduced further slightly compared to the result presented in Section 7.2.2, which is obtained by using a model that does not include ESS. The difference yields 0.01 %, which corresponds to 37,830 NOK. It can also be observed that the model with ESS allowed more grid power to be imported meanwhile reducing the fuel consumption during the simulation period.

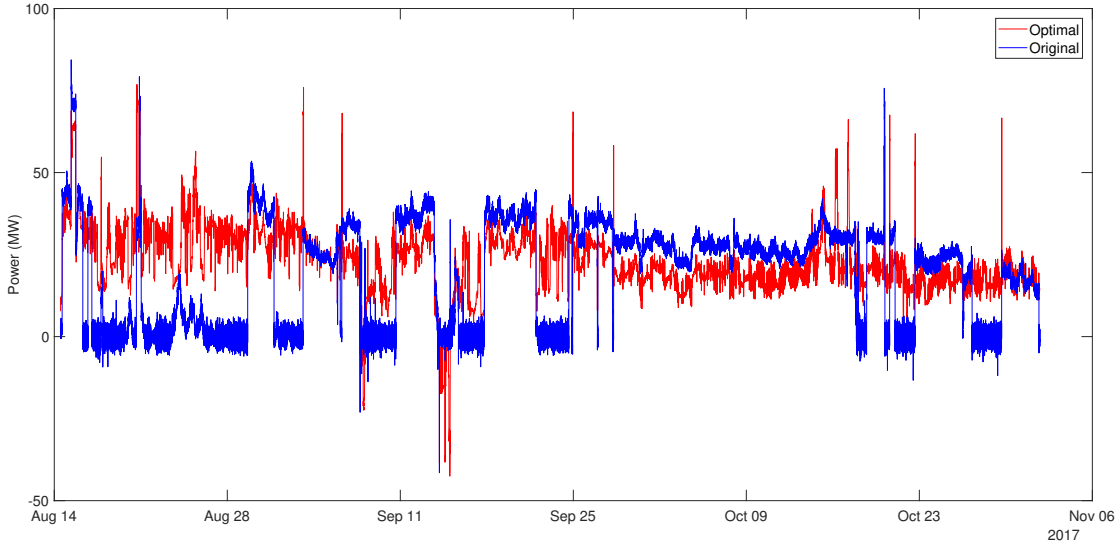


Figure 7.2.7: Optimal vs. original grid power when ESS is used

Figure 7.2.7 shows imported/exported power from the grid.

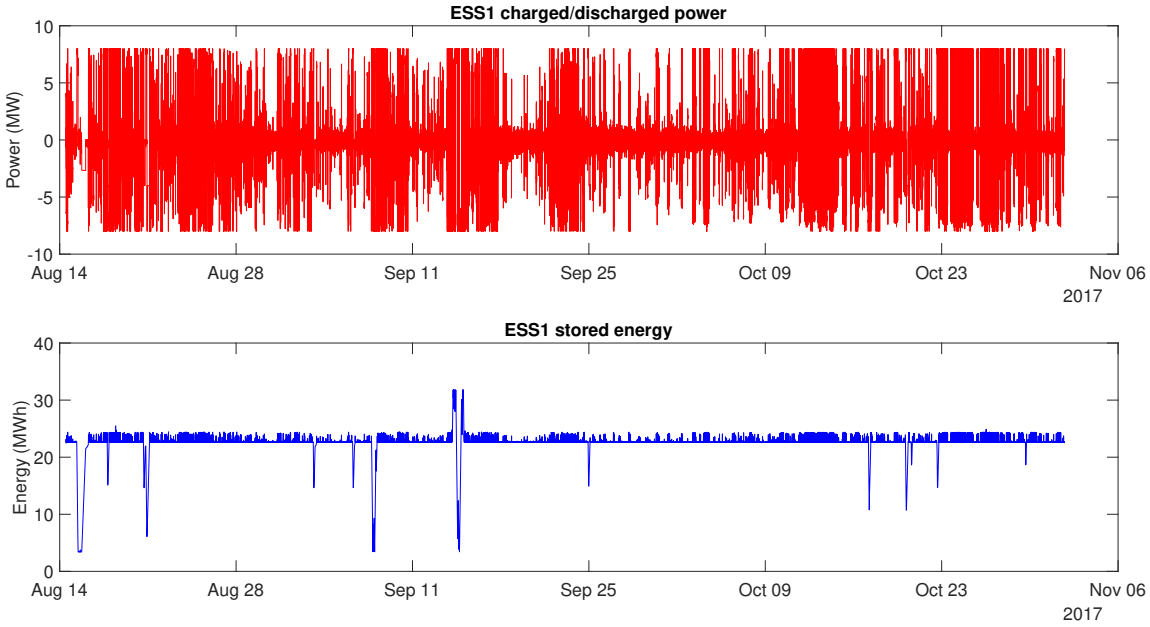
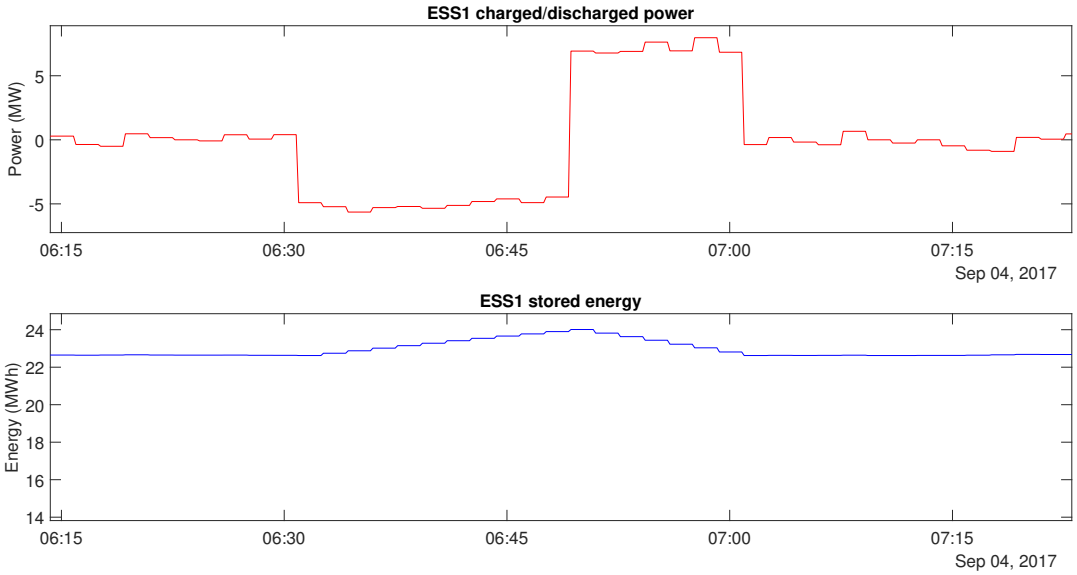


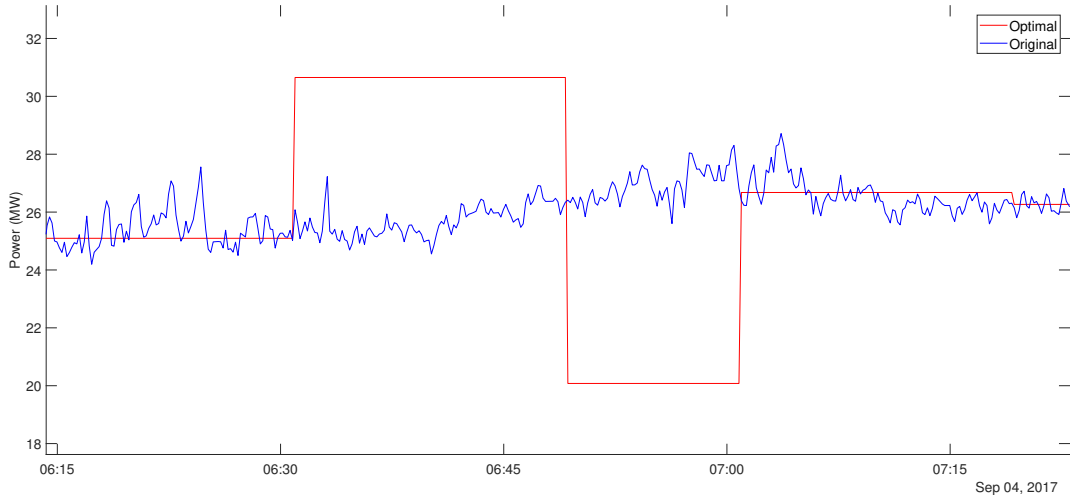
Figure 7.2.8: Charge/discharge rate and stored energy of ESS

Figure 7.2.8 shows how ESS is controlled by the EMS algorithm. ESS is

changing its state frequently. The stored energy of ESS stays at its minimum (22.4 MWh) most of the periods. If fewer than four gas turbines are operable, ESS is allowed to be discharged to an energy level below the minimum energy level defined, which is observed several times during the simulation period.



(a) ESS power and ESS energy



(b) Grid power

Figure 7.2.9: Correlation between power from the ESS and power from the grid

Figure 7.2.9 shows that the ESS is used to match the high-frequency change of power demand, which comes from stabilization of the grid frequency. The ESS can both be charged or discharged, meaning that power from the ESS can be positive and negative. This property of the ESS will let the grid to operate more smoothly. In another word, the ESS can be used to shift the total load and used for frequency regulation so that power from the grid can be stabilized as shown in Figure 7.2.9 b). The import/export level of power from the grid does not oscillate and is stabilized. Power from the ESS and the grid is correlated in a way that ESS is charged by importing more power from the grid and when ESS is being discharged, less power is imported from the grid.

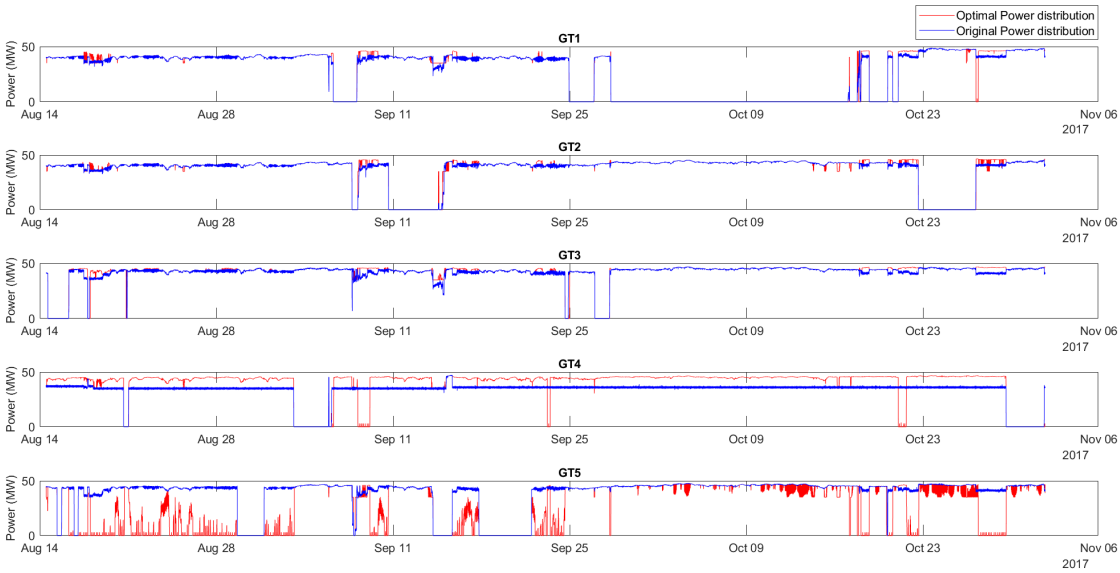
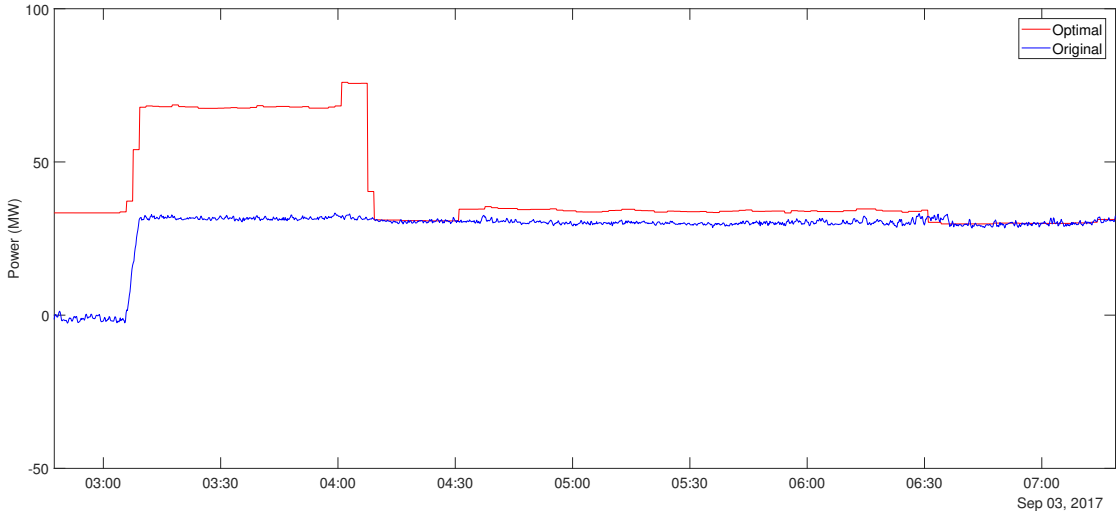
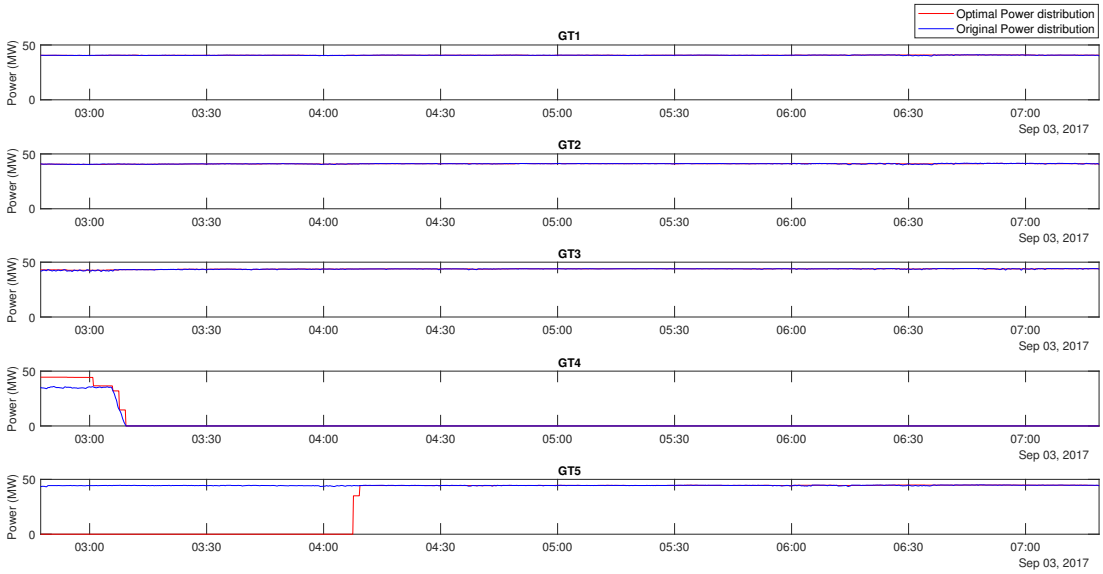


Figure 7.2.10: Optimal vs. original power distribution when ESS is used

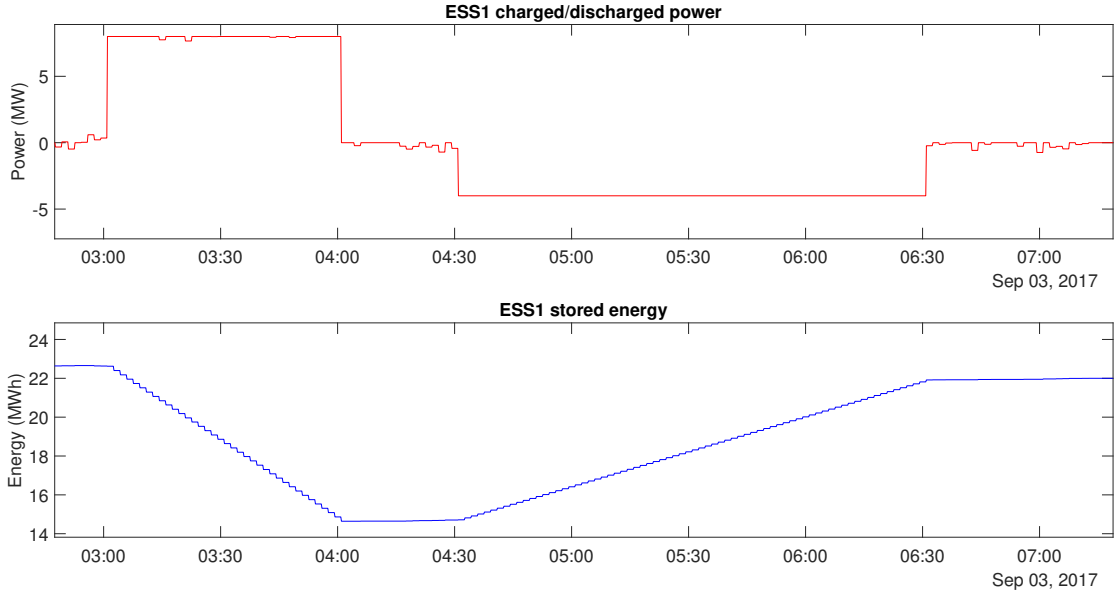
Figure 7.2.10 shows the power distribution between the gas turbines. GT1, GT2, and GT3 are controlled very similarly to the original case. GT4 is operating with its maximum capacity most of the time, which is different from the original case. GT5 does not resemble the original case at all and is controlled in a special way. The EMS is trying to keep GT5 turned off and makes GT5 produce the minimum power to match power demand. This pattern of operation is observed between the period 14^{th} of August and 25^{th} of September. This is as expected considering BSFC graphs presented in Figure 7.2.2. GT5 requires minimum fuel considering the efficiency when generating relatively low power.



(a) Grid power



(b) GT power



(c) ESS power and ESS energy

Figure 7.2.11: ESS behaviour when less than four GT turbines are operable

Based on the data from the site, each trip of the grid/turbine will result in the loss of production of approximately 41 MNOK. Installation of an ESS can minimize this loss by providing the necessary power during the recovery time. It is unclear how much the loss can be reduced by using an ESS. However, it can be assumed that ESS can reduce the number of the trip by one per year, which will give the saving of 41 MNOK per year.

Figure 7.2.11 illustrates how ESS is discharged to an energy level below the pre-defined minimum energy of ESS, which is 22.4 MWh. ESS is allowed to be discharged further when fewer than four gas turbines are operable. From the simulation data, it can be observed that ESS compensates one of the non-operable GT between the time 03:00 and 04:00 in 14^{th} of September. For this period, ESS is providing the maximum power it can to the site. ESS is charged back to the *normal* state with charging rate of 4 MW as soon as four gas turbines become operable, which can be observed between the time 04:30 and 06:30.

CHAPTER 8

CONCLUSIONS

This work presented a MINLP model for a hybrid power system embedding n-number of gas turbines, grid as well as ESS. The model includes a performance parameter BSFC, which is estimated based on the data from a site using curve-fitting method provided by MATLAB. The works presented in [2] and [3], which are conducted in parallel to this work describes how BSFC curves can be modeled by pre-processing the data and how BSFC curves can be updated online. Moreover, a predictive method was integrated into the system to estimate the future power demand in order to schedule the load distribution for an interval of 30 minutes between the existing generating units. The designed EMS using the MINLP model helped to achieve the optimal loading.

The simulation is performed using the data from a site during three months of operation starting from August 2017. The outcome of the simulation is compared with the original data. The simulation shows that the load distribution using the EMS based on MINLP will benefit in terms of the cost-efficient operation as well as the fuel saving.

The result yields that allowing more grid power to be imported increases the total cost saving. However, it will also increase grid dependency. Therefore, it is desired to keep the import level of the grid power at the same level as the original case. The commitment of the ESS will benefit in terms of the cost-efficient operation and the fuel saving. More grid power can be allowed to be imported by using ESS as an alternative source of energy in case the grid trips. ESS will contribute to the increased robustness of the system. In addition, the grid frequency stabilization can be achieved by shifting load using ESS. Conclusively, use of the ESSs will give more flexibility in the power production and increase the reliability of the power system.

Despite the cost-saving obtained in this work, there are some uncertainties that need to be taken into account. Firstly, BSFC estimation error can affect the performance of the EMS. BSFC will be significantly different depending on the type of the gas used as fuel as well as the quality of the gas. Moreover, the parameters of BSFC curves tends to change over time. Further research that includes a robust online update of the BSFC curves to the optimization algorithm will increase the soundness of the simulation result. The price of gas is assumed to be fixed, which is not realistic. The price of gas will vary depending on many factors and the cost-saving achieved in this work will be slightly different from the actually achievable cost-saving. The algorithm is exposed to the forecast error, which will affect the robustness of the algorithm. However, the forecast error is minor and can be corrected by the site operators. Due to this lack of the robustness of the algorithm, human interference is essentially required. The electricity price is assumed to be fixed for a time horizon of 30 minutes. Since the *Nordpool* electricity price change on an hourly basis, this assumption is valid. Moreover, the maximum power output from the gas turbines is assumed to be known with complete certainty. This will not be a realistic case. In a realistic case, the future maximum power output is unknown and needs to be predicted or assumed to be fixed. This assumption might affect the soundness of the optimization outcome and make the result achieved in this work slightly differ from the actually achievable cost-saving. However, the optimization result will not be affected too much by this since the maximum power output is not a fast-changing parameter. Furthermore, the influence of the parameters presented through this work needs to be investigated further. Most of the parameters are found using trial and error method. Thus, in order to obtain the full potential saving and the correctness of the algorithm presented in this work, systematic tuning of the parameters needs to be conducted.

The soundness of the algorithm is considered when designing the optimization problem model. For instance, operational constraint specific to the site where the data is recorded is added to the model. Moreover, the minimum time it takes to start a gas turbine is considered. By integrating a predictive method for scheduling the load distribution, the algorithm was able to consider the start-up time to some degree. However, this could not be embedded into the model completely since it requires start-up dynamic of the gas turbines. Furthermore, the reasonable energy storage capacity of the ESS is also considered and integrated into the model.

Conclusively, the result yields that the optimal load sharing can help reduce the cost of the operation substantially up to 2.48 %.

CHAPTER 9

FUTURE WORK

9.1 Robustness of the algorithm

The optimization algorithm presented in this work does not provide any feedback control of the power from the gas turbines. Meaning that if the power demand forecast error is considerably large, the power system requires human interference for correction of the generated power to match the actual power demand. An MPC approach implementation similar to the algorithm presented in [21] can be considered to overcome the uncertainties due to the forecast error. Increased robustness is expected to be achievable by introducing a feedback control using MPC. Moreover, MPC will contribute preventing ramp constraint violation and always provide the feasible solutions. MPC only implements the first solution that uses the real measurements, hence, the feasibility of the solution is secured.

9.2 Grid failure safety

The model presented in this work does not consider the grid failure safety. Grid failure that leads to a power blackout resulting a short-term or a long-term loss of the electric power to a particular area is not considered to exist. However, in a realistic industrial environment, the grid can fail quite frequently, up to two times on yearly basis. In terms of reliability, this is a crucial part that needs further investigation. One way of including safety of the grid in a certain degree is implementing an algorithm to predict the grid failure points. The optimization algorithm will use this information to control the gas turbines to minimize the possible loss due to the grid failure. A machine learning algorithm can be considered to detect the future grid failure points.

9.3 Renewable energy resources

The model (5.5.6) can be extended further to include renewable energy resources such as wind turbine and solar panels. This work requires knowledge about the renewable energy equipment dynamics that need to be presented as a set of equations. Moreover, local weather needs to be taken into account such as wind speed and heat from the Sun. Wind speed and heat can be measured using different types of sensors. For power distribution scheduling purpose, local weather needs to be predicted and the uncertainty of the prediction needs to be taken into account. For this, weather forecast from reliable sources can be used. A useful weather forecast source can be YR. Similar work of including renewable energy sources as a part of MILP model is conducted in [17] and [18].

9.4 Start-up dynamic of gas turbines

The model presented in this thesis is not able to find the optimal point to turn on the gas turbines. The model can suggest when to turn on the not-operating gas turbines, however, it does not consider the time it takes before the gas turbines can actually start to produce power. In another word, the algorithm developed in this work does not consider the turning-on dynamic of the gas turbines. Ideally, in order to find the optimal point to start preparing a gas turbine to be operated, the optimization algorithm needs a dynamic model for the off-state gas turbines. A considerable amount of work will be required to come up with an off-state dynamic model for the gas turbines. Another approach can be increasing the forecast horizon T significantly large. The optimal turning on point of the gas turbines, which needs to be provided to the site operators at-least 2 hours prior to the current time can be used as a signal to start preparing off-state gas turbines. The work presented here will use 30 minutes as the forecast horizon, which is not enough to fully catch the optimal point of turning on the gas turbines. However, having T bigger than 2 hours will still be sub-optimal without considering the off-state dynamic of the gas turbines.

9.5 Systematic tuning of the important model parameters

The influence of the model parameters presented through this work needs to be investigated further. Most of the parameters are found using trial and error method. Thus, in order to obtain the full potential saving and the correctness of the algorithm presented in this work, systematic tuning of the parameters needs to be conducted.

LIST OF SYMBOLS

f_i	Fuel consumption of ith gas turbine	$\frac{Kg}{\Delta t}$
b_i	BSFC measurement	$\frac{Kg}{MWh}$
p_i	Power from ith gas turbine	MW
$load_i$	Load on ith gas turbine	pu
p_d	Power demand from the site	MW
$L_{\Delta t}$	Lower ramp rate limit of the gas turbines	$\frac{MW}{\Delta t}$
$H_{\Delta t}$	Upper ramp rate limit of the gas turbines	$\frac{MW}{\Delta t}$
$p_{i,min}$	Minimum power from ith gas turbine	MW
$p_{i,max}$	Maximum power from ith gas turbine	MW
p_{req}	Minimum power requirement of the gas turbines	MW
p_g	Import/export power from the grid	MW
$p_{g,lim}$	Instantaneous power limit from the grid	MW
$p_{g,sp}$	Grid set-point power	MW
$p_{j,E}$	Power flow of jth ESS	MW
$p_{E,min}$	Maximum charging rate of jth ESS	MW
$p_{E,max}$	Maximum discharging rate of jth ESS	MW
E_j	Energy stored in jth ESS	MWh
E_{min}	Minimum capacity of jth ESS	MWh
E_{max}	Maximum capacity of jth ESS	MWh
C_{fuel}	Gas price	$\frac{NOK}{kg}$
$C_{fuel,MW\Delta t}$	Gas price	$\frac{NOK}{MW\Delta t}$
C_{el}	Grid electricity price	$\frac{NOK}{MW\Delta t}$
C	The total operational cost	NOK
$M_{rp,\Delta t}$	Ramp rate *Big-M*	$\frac{MW}{\Delta t}$
M_{st}	Gas turbine start-up *Big-M*	t
M_E	ESS capacity *Big-M*	MWh

$\boldsymbol{W}_{in}$	ESN input weights
$\boldsymbol{W}_{reservoir}$	ESN reservoir weights
$\boldsymbol{W}_{out}$	ESN output weights
$\boldsymbol{W}_{back}$	ESN feed back weights
$\boldsymbol{b}$	ESN bias vector
$\boldsymbol{W}_{noise}$	Scaled white noise added to ESN reservoir states
f_{state}	ESN reservoir activation function
β	Tikhonov regulation parameter
u_i	On/off decision variable of ith gas turbine
$u_{j,E}$	Discharge/charge decision variable of jth ESS
n_{min}	Minimum number of the gas turbines to be active
l_{min}	Minimum number of the gas turbines with minimum power p_{req}
w_{Φ}	Weight of the power change penalty
$w_{\Phi,g}$	Weight of the grid set-point penalty
Φ	Absolute value of the grid power difference between each EMS time interval
Φ_{gt}	Absolute value of the gas turbine power difference between each EMS time interval
r	Recovery ratio of ESS
T	Prediction Horizon

LIST OF FIGURES

LIST OF TABLES

REFERENCES

[1] Jung B.K. "An MINLP approach for optimal load sharing in generating machine". Norwegian University of Science and Technology (NTNU), Project thesis (2017).

[2] Espeland. "Offline Modelling of Fuel Efficiency Curves in Gas Turbines". Norwegian University of Science and Technology (NTNU), Project thesis (2017).

[3] Espeland. "Online Modelling of Fuel Efficiency Curves in Generating Machines". Norwegian University of Science and Technology (NTNU), Master's thesis (2018).

[4] Norwegianpetroleum.no. (2017). Emissions to air Norwegianpetroleum.no. [online] Available at: *http* : *//www.norskpetroleum.no/en/environment* − *and* − *technology/emissions* − *to* − *air/* [Accessed 18 Nov. 2017].

[5] Rajamäki, Rami. "Load Sharing Communication between Different Engine/Generator Controllers.". Vaasa University of Applied Sciences (2015).

[6] Nordpoolgroup.com. (2017). Market data, Nord Pool. [online] Available at: *https* : *//www.nordpoolgroup.com/Market* − *data*1/#/*nordic/chart* [Accessed 20 Nov. 2017].

[7] Chen, Cheng-Liang, Chieh-Ting Lai, and Jui-Yuan Lee. "Transshipment model-based MILP (mixed-integer linear programming) formulation for targeting and design of hybrid power systems." Energy 65 (2014): 550-559.

[8] Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. "Unsupervised learning." The elements of statistical learning. Springer, New York, NY, 2009. 485-585.

[9] J. Blondin, Particle swarm optimization: a tutorial. [Online]. Available from: $http://cs.armstrong.edu/saad/csci8100/pso_tutorial.pdf$. [Accessed 4 Feb. 2018].

[10] Zitzler, Eckart, Marco Laumanns, and Stefan Bleuler. "A tutorial on evolutionary multiobjective optimization." Metaheuristics for multiobjective optimisation. Springer, Berlin, Heidelberg, 2004. 3-37.

[11] Van Den Bergh, Frans, and Andries Petrus Engelbrecht. "A study of particle swarm optimization particle trajectories." Information sciences 176.8 (2006): 937-971.

[12] Van Den Bergh Frans. An Analysis of Particle Swarm Optimizers. PhD thesis, University of Pretoria. (2001)

[13] Lundh, Michael, et al. "Estimation and Optimization of Vessel Fuel Consumption." IFAC-PapersOnLine 49.23 (2016): 394-399.

[14] Castillo, Enrique, et al. Building and solving mathematical programming models in engineering and science. Vol. 62. John Wiley & Sons, 2011. APA. 172-179

[15] Skjong, Espen, et al. "Approaches to Economic Energy Management in Diesel–Electric Marine Vessels." IEEE Transactions on Transportation Electrification 3.1 (2017): 22-35.

[16] Frisk, Mikael. "On-ship Power Management and Voyage Planning Interaction.". UPPSALA University, Master thesis (2015).

[17] Ren, Hongbo, and Weijun Gao. "A MILP model for integrated plan and evaluation of distributed energy systems." Applied Energy 87.3 (2010): 1001-1014.

[18] Atla, Chandra Shekhar Reddy. "Optimal Short Term Hydro Thermal Coordination for Large Scale Power System Using MILP." Bangalore, India.

[19] Tuffaha, Mutaz, and Jan Tommy Gravdahl. "Dynamic formulation of the unit commitment and economic dispatch problems." Industrial Technology (ICIT), 2015 IEEE International Conference on. IEEE, 2015.

[20] Gui, Yonghao, et al. "Intra-day unit commitment for wind farm using model predictive control method." Power and Energy Society General Meeting (PES), 2013 IEEE. IEEE, 2013.

[21] Xia, Xiaohua, Jiangfeng Zhang, and Ahmed Elaiw. "An application of model predictive control to the dynamic economic dispatch of power generation." Control Engineering Practice 19.6 (2011): 638-648.

[22] Xie, Le, and Marija D. Ilic. "Model predictive economic/environmental dispatch of power systems with intermittent resources." Power & Energy Society General Meeting, 2009. PES'09. IEEE. IEEE, 2009.

[23] Cristopher Olah. 2015. Understanding LSTM Networks. [ONLINE] Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 26 September 2017].

[24] Schmidhuber, Jürgen, and Sepp Hochreiter. "Long short-term memory." Neural Comput 9.8 (1997): 1735-1780.

[25] Jaeger, Herbert. Short term memory in echo state networks. Vol. 5. GMD-Forschungszentrum Informationstechnik, 2001.

[26] Verstraeten, David, et al. "Memory versus non-linearity in reservoirs." Neural Networks (IJCNN), The 2010 International Joint Conference on. IEEE, 2010.

[27] Wyffels, Francis, Benjamin Schrauwen, and Dirk Stroobandt. "Stable output feedback in reservoir computing using ridge regression." International conference on artificial neural networks. Springer, Berlin, Heidelberg, 2008.

[28] Jaeger, Herbert. Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach. Vol. 5. Bonn: GMD-Forschungszentrum Informationstechnik, 2002.

[29] Jaeger, Herbert, and Harald Haas. "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication." science 304.5667 (2004): 78-80.

[30] Jaeger, Herbert, et al. "Optimization and applications of echo state networks with leaky-integrator neurons." Neural networks 20.3 (2007): 335-352.

[31] Jaeger, Herbert. "Adaptive nonlinear system identification with echo state networks." Advances in neural information processing systems. 2003.

[32] Jaeger, Herbert. "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note." Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148.34 (2001): 13.

[33] Reinhart, René Felix, and Jochen J. Steil. "Reservoir regularization stabilizes learning of Echo State Networks with output feedback." ESANN. 2011.

[34] Lukoševičius, Mantas. "A practical guide to applying echo state networks." Neural networks: Tricks of the trade. Springer, Berlin, Heidelberg, 2012. 659-686.

[35] Li, Decai, Min Han, and Jun Wang. "Chaotic time series prediction based on a novel robust echo state network." IEEE Transactions on Neural Networks and Learning Systems 23.5 (2012): 787-799.

[36] Ishu, K., et al. "Identification of motion with echo state network." OCEANS'04. MTTS/IEEE TECHNO-OCEAN'04. Vol. 3. IEEE, 2004.

[37] Maass, Wolfgang, Prashant Joshi, and Eduardo D. Sontag. "Principles of real-time computing with feedback applied to cortical microcircuit models." Advances in neural information processing systems. 2006.

[38] Skowronski, Mark D., and John G. Harris. "Minimum mean squared error time series classification using an echo state network prediction model." Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on. IEEE, 2006.

[39] Rabiner, Lawrence R. "A tutorial on hidden Markov models and selected applications in speech recognition." Proceedings of the IEEE 77.2 (1989): 257-286.

[40] Anton, Howard; Rorres,, Chris. Elementary Linear Algebra (9th ed.). John Wiley and Sons, Inc. ISBN 978-0-471-66959-3 (2005).

[41] Golub, Gene H., Per Christian Hansen, and Dianne P. O'Leary. "Tikhonov regularization and total least squares." SIAM Journal on Matrix Analysis and Applications 21.1 (1999): 185-194.

[42] Basterrech, Sebastián, Enrique Alba, and Václav Snášel. "An experimental analysis of the echo state network initialization using the particle swarm optimization." Nature and Biologically Inspired Computing (NaBIC), 2014 Sixth World Congress on. IEEE, 2014.

[43] Wei, Li, and Li Haitian. "Electrical Load Forecasting Using Echo State Network and Optimizing by PSO Algorithm." Intelligent Computation Technology and Automation (ICICTA), 2017 10th International Conference on. IEEE, 2017.

[44] Lendasse, Amaury, and Elia Liitiainen. "Variable scaling for time series prediction: Application to the ESTSP'07 and the NN3 forecasting competitions." Neural Networks, 2007. IJCNN 2007. International Joint Conference on. IEEE, 2007.

[45] Liao, T. Warren. "Clustering of time series data—a survey." Pattern recognition 38.11 (2005): 1857-1874.

[46] Winston, Wayne L., Munirpallam Venkataramanan, and Jeffrey B. Goldberg. Introduction to mathematical programming. Vol. 1. Duxbury; Pacific Grove, CA: Thomson/Brooks/Cole, 2003.

[47] Nocedal, J. and Wright, S. (2006). Numerical optimization. New York: Springer, pp.355-389.

[48] Farhang-Boroujeny, Behrouz. Adaptive filters: theory and applications. John Wiley & Sons, 2013.

[49] E. F. Camacho and C. Bordons, Model predictive control, 2nd ed. Berlin: Springer Verlag, 2004.

[50] Saravanamuttoo, Herbert Ian Howard, Gordon Frederick Crichton Rogers, and Henry Cohen. Gas turbine theory. Pearson Education, 2001.

[51] ELECTRICITY DELIVERY & ENERGY RELIABILITY, (2014). Tehachapi Wind Energy Storage Project. [online] Available at: $https://energy.gov/oe/downloads/fact-sheet-tehachapi-wind-energy-storage-project-may-2014$ [Accessed 7 Nov. 2017].