Distributed Model Predictive Control of Load Frequency for Power Networks

Christian Fogelberg

LUNDS UNIVERSITET

Department of Automatic Control

MSc Thesis
ISRN LUTFD2/TFRT--5931--SE
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Thesis Abstract

Distributed Model Predictive Control of Load Frequency for Power Networks

In recent years, there has been an increase of interest in smart grid concept, to adapt the power grid to improve the reliability, efficiency and economics of the electricity production and distribution. One of the generator side problem in this is to meet the power requirement while not wasting unnecessary power, thus keeping the cost down, which must be done while the frequency is kept in a suitable range that will not damage any equipment connected to the power grid. It would theoretically be most logical to have a centralized controller that gathers the full networks data, calculates the control signals and adjusts the generators. However in practice this is not practical, mostly due to distance. The transmission of sensor data to the controller and the transmission of control signals to the generators would have to travel far, thus taking up to much time before the generators could act.

This paper presents a distributed model predictive control based method to control the frequency of the power network. First, an augmented matrix model predictive controller is introduced and implemented on a two homogeneous subsystems network. Later the control method is changed to a state space model predictive controller and is then utilized on a four heterogeneous subsystems network. This controller implementation also includes state observers by Kalman filtering, constraints handler utilizing quadratic programming, and different connection topology setups to observe how the connectivity affects the outcome of the system.

The effectiveness of the proposed distributed control method was compared against the corresponding centralized and decentralized controller implementation results. It is also compared to other control algorithms, specifically, an iterative gradient method, and a model predictive controller generated by the MATLAB MPC Toolbox. The results show that the usage of a distributed setup improves the outcome compared to the decentralized case, whilst keeping a more convenient setup than the centralized case. It it also shown that the level of connectivity for a chosen network topology matters for the outcome of the system, the results are improved when more connections exists.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In recent years, there has been an increase of interest in smart grid concept, depicted in Fig. 1.1, to adapt the power grid to improve the reliability, efficiency and economics of the electricity production and distribution, and the many ways this can be achieved [1]. One of the generator side problem in this is to meet the power requirement while not wasting unnecessary power, thus keeping the cost down, which must be done while the frequency is kept in a suitable range that will not damage any equipment connected to the power grid. To get a result as good as possible the generators needs to be controlled with a fast and reliable control method. Since it is essential that the frequency does not deviate to far from the standard frequency of the network, since a too big difference might even cause a blackout, the control need to be as precise as possible. To get the control as accurate as possible, the more that the controller can take into account of the full network the higher the degree of accuracy will be.

In a small isolated grid, for example the one presented in [3], it is very easy to see the influence on the frequency due to renewable energy sources like wind power. The grid in that particular example has been confirmed to have an upper limit of about 15% on the amount of wind power that it can support before the frequency warps to far from normality. One of the implemented control methods in this grid is load control [2], temporarily shutting down less important power consuming equipment such as store refrigerators to alter the power load in such a way that the frequency stabilizes. The problems that wind power introduces to the grid, some which can even be due to faulty wind turbines [6], can be dealt with in many ways, especially since the systems can be very different by them selfs and how they are set up [5]. One example of available options is to do what Germany is considering doing, to help stabilize their power grid with the help of high-voltage direct current (HVDC) [4]. By introducing HVDC power corridors through the country they can direct power to where it is needed without overtaxing the existing grid, and at the same time use the power converters to stabilize the frequency of the connected AC grid.

It would theoretically be most logical to have a centralized controller that gathers the full networks data, calculates the control signals and adjusts the generators. However in practice this is not practical, mostly due to distance. The transmission of sensor data to the controller and the transmission of control signals to the generators would have to travel far, thus taking up to much time before the generators could act.

Therefore in this paper, a distributed model predictive control (MPC) approach to control each power plant output frequency as to not deviate from the predefined output is proposed. The advantage of MPC is that it generalizes directly to plants being multiple-input and multiple-output (MIMO), which can be non-square, and can take process constraints into account, which eliminates the possibility of variables exceeding their predetermined limits. Lately, there has been a lot of research about implementing MPC methods to different kinds of systems [20, 28] and a lot of books covering the basics [29, 31], proving that it is a very versatile control algorithm that often can engender a distinct result relative to the preferred outcome to the problem at hand. Although, the setup procedure of a model predictive controller might be seen as fraught with peril for the uninitiated, due to being slightly more complex than some more frequently used methods.

The main controller implementation in this paper is a distributed controller. The distributed controller gathers data from its connected neighboring subsystems so to calculate a control signal for its own generators with regards to a bit bigger part of the whole network than just its own. This should produce a better outcome than a decentralized controller that only calculates from its own data, but not quite as good as a centralized controller that takes the whole system into consideration. The advantage a distributed control implementation has over a centralized control implementation is that it is more local, thus long signal transmission times should not present itself, which thus leads to a faster control response.

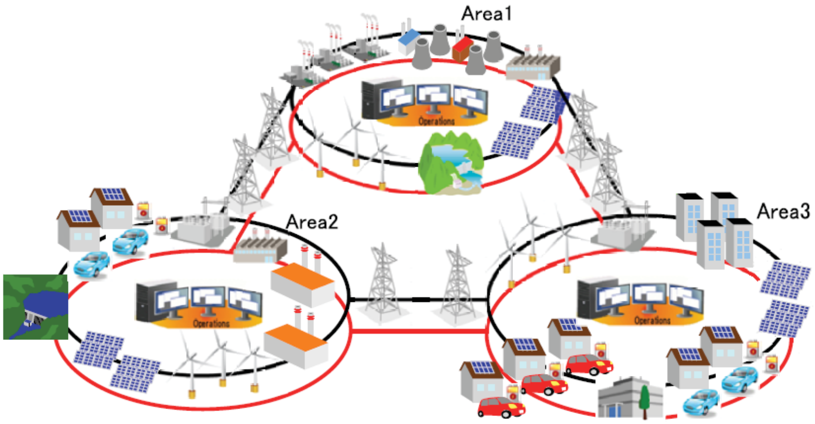


Fig. 1.1 : Smart grid

1.2 Thesis Objectives and Contributions

The objective of this paper is to investigate the benefits of using a distributed controller on a power network as an alternative to the more common centralized controller and decentralized controllers. The overall generator control cost as well as the frequency deviation output will be taken into consideration in the final evaluation.

One further matter that this paper also investigates is the implications different connection topologies has on the overall generator control cost and frequency deviation output. In this case, three different connection topologies has been considered, a linear connected topology, a square connected topology, and a fully connected topology, and their respective advantages are discussed.

The susceptibility to delay in measurement signal delivery are also studied, since the distributed controller relies on data from the neighboring subsystems as well. A simple solution to deal with long transmission delays and transmission loss of measurement data in a real system are also proposed.

1.3 Thesis Overview

Firstly, the problem formulation is presented and discussed together with the objective of the controller.

Then the first control method is presented together with a small two subsystems homogeneous network that is to be controlled. The obtained results are presented and compared with an iterative gradient control method discussed in [12, 15].

Next chapter introduces another slightly different model predictive control method and a bigger four subsystems heterogeneous network. This chapter also introduces a state observer, the constraints handler, the different connection topologies, and some discussions about data acquisition. The state observer used in this paper is a standard Kalman filter, but some alternatives are also discussed [16, 19], and the constraint handler used utilizes quadratic programming [10, 11]. This new control method is also compared to the previous method on the smaller network setup, and on the big system it is compared to another MPC method, one generated by the MATLAB MPC Toolbox [7, 9].

Finally the conclusion is presented with the discussions about the results acquired from the different inspected setups.

Chapter 2

Problem formulation

The problem that is considered is an electric power network that consist of $N(\geq 2)$ connected subsystems, as for example the linear connected network shown in Fig. 2.1.

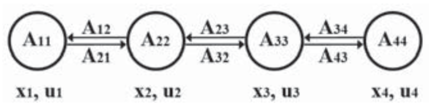


Fig. 2.1 : Linear connection

The normal state space formulation for the whole system, which is assumed to be

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + w(k) \\ y(k) &= Cx(k) + v(k) \end{aligned} \tag{2.1}$$

can be separated into smaller subsystem equations. Assuming that each subsystem can use information of neighboring subsystems, then the i-th subsystem is given by

$$x_i(k+1) = \sum_{j=1}^{N} A_{ij}x_j(k) + B_iu_i(k) + w(k) \qquad i = 1, \ldots, N \tag{2.2}$$

where k is the time, $x_i(k) \in \mathbb{R}^{n_{xi}}$ is the states of system i, $A_{ij} \in \mathbb{R}^{n_{xj} \times n_{xj}}$, if $j = i$ then $A_{ii} \in \mathbb{R}^{n_{xi} \times n_{xi}}$, $u_i(k) \in \mathbb{R}^{n_{ui}}$ is the control signals of system i and $B = \text{diag}[B_1, \cdots, B_N] \in \mathbb{R}^{n_u \times n_u}$. The process noise and the measurement noise, $w(t)$ and $v(t)$ is assumed to be uncorrelated zero-mean Gaussian white noise.

The set $\mathcal{N}_i$ includes the subsystems that subsystem i is connected to, so when the i-th subsystem is connected to the j-th subsystem it can be written as $j \in \mathcal{N}_i$. If they are not connected it will give that

$$A_{ij} = 0 \quad \text{if } j \notin \mathcal{N}_i. \tag{2.3}$$

This means that the system equation (2.2) also can be expressed as

$$x_i(k+1) = A_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i} A_{ij}x_j(k) + B_i u_i(k) + w(k). \tag{2.4}$$

The different controllers to be implemented are a centralized controller, decentralized controllers, and distributed controllers.

The centralized controller uses the information of the whole system, while the decentralized controllers only uses the information of their own subsystem. The distributed controller uses its own information and the neighboring subsystems information.

The output of the system that is to be controlled is the deviation of the frequency from the normal frequency of the whole system. To accomplish this all controllable power generator in the system are used and are actively trying to minimize the frequency deviation, since to high deviation can cause permanently damage to connected equipment and may lead to blackouts.

The system will be modeled such that the zero states and zero outputs will represent the optimal value. Thus all deviation from zero will raise some cost in the network. Changing the control signal will lead to changed state values, and changes in states can be seen as changes in energy output from the generators. A fluctuating generator output is harder to sustain and control than a stable energy output. Therefore the calculation of the system cost will depend on all or some of the states, and the control signal or the change in control signal.

Chapter 3

Augmented matrix model predictive control

3.1 Model predictive controller

First define the change in control signal as

$$\Delta u(k) = u(k) - u(k-1). \tag{3.1}$$

Inserting this into the normal state space model gives

$$\begin{aligned} x(k+1) &= Ax(k) + B(u(k-1) + \Delta u(k)) + w(k) \\ y(k) &= Cx(k) + v(k). \end{aligned} \tag{3.2}$$

Rearranging (3.1) and (3.2) we can write the system as

$$\begin{aligned} \begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} &= \underbrace{\begin{bmatrix} A & B \\ 0 & I \end{bmatrix}}_{\hat{A}} \underbrace{\begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}}_{\hat{x}(k)} + \underbrace{\begin{bmatrix} B \\ I \end{bmatrix}}_{\hat{B}} \Delta u(k) + w(k) \\ y(k) &= \underbrace{\begin{bmatrix} C & D \end{bmatrix}}_{\hat{C}} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + v(k) \end{aligned} \tag{3.3}$$

thus, simplified written as

$$\begin{aligned} \hat{x}(k+1) &= \hat{A}\hat{x}(k) + \hat{B}\Delta u(k) + w(k) \\ y(k) &= \hat{C}\hat{x}(k) + v(k). \end{aligned} \tag{3.4}$$

The eigenvalues of the augmented model will then be calculated as

$$\rho(\lambda) = \det \begin{bmatrix} \lambda I - A & B \\ 0 & (\lambda - 1)I \end{bmatrix} = (\lambda - 1)^q \det(\lambda I - A) = 0 \tag{3.5}$$

which are the eigenvalues of the original model, $\det(\lambda I - A)$, and q eigenvalues at $\lambda = 1$.

To specify how many steps ahead the controller should take into account, the prediction horizon and control horizon are defined.

Definition 1.
We denote the prediction horizon as N_p and the control horizon as N_c.
The prediction horizon defines how many steps ahead the controller should try to predict the states.
The control horizon dictates the number of steps the controller should try to complete the control objective in.

Also, the control horizon needs to be limited as to not cause problems in the calculations.

Assumption 1.
It is assumed that the control horizon N_c is chosen to be less than or equal to the prediction horizon N_p

$$N_c \leq N_p$$

since it is not possible to predict a control trajectory without having predicted the states at that time instant.

At time k we denote the future control trajectory as

$$\underrightarrow{\Delta u} = [\Delta u(k) \;\; \Delta u(k+1) \;\; \cdots \;\; \Delta u(k+N_c-1)].$$

The prediction horizon dictates how many samples ahead the future states are predicted, denoted as

$$\underrightarrow{\hat{x}} = [\hat{x}(k+1|k) \;\; \hat{x}(k+2|k) \;\; \cdots \;\; \hat{x}(k+N_p|k)]$$

where $\hat{x}(k+t|k)$ is the predicted state variables at $k+t$ given current information $\hat{x}(k)$. Based on the state space model (3.4), the future state variables are calculated sequentially using the future control parameters.

$$\begin{aligned}
\hat{x}(k+1|k) &= \hat{A}\hat{x}(k) + \hat{B}\Delta u(k) \\
\hat{x}(k+2|k) &= \hat{A}\hat{x}(k+1) + \hat{B}\Delta u(k+1) \\
&= \hat{A}^2\hat{x}(k) + \hat{A}\hat{B}\Delta u(k) + \hat{B}\Delta u(k+1) \\
&\vdots \\
\hat{x}(k+N_p|k) &= \hat{A}^{N_p}\hat{x}(k) + \hat{A}^{N_p-1}\hat{B}\Delta u(k) + \hat{A}^{N_p-2}\hat{B}\Delta u(k+1) \\
&\quad + \cdots + \hat{A}^{N_p-N_c}\hat{B}\Delta u(k+N_c-1)
\end{aligned} \tag{3.6}$$

From the above equation we can get the predicted output variables, by substitution, so all predicted variables are formulated in terms of current state variables information $\hat{x}(k)$ and the future control movement $\Delta u(k+t)$, where $t = 0, 1, \ldots, N_c - 1$.

$$\begin{aligned}
y(k+1|k) &= \hat{C}\hat{A}\hat{x}(k) + \hat{C}\hat{B}\Delta u(k) \\
y(k+2|k) &= \hat{C}\hat{A}^2\hat{x}(k) + \hat{C}\hat{A}\hat{B}\Delta u(k) + \hat{C}\hat{B}\Delta u(k+1) \\
y(k+3|k) &= \hat{C}\hat{A}^3\hat{x}(k) + \hat{C}\hat{A}^2\hat{B}\Delta u(k) + \hat{C}\hat{A}\hat{B}\Delta u(k+1) + \hat{C}\hat{B}\Delta u(k+2) \\
&\vdots \\
y(k+N_p|k) &= \hat{C}\hat{A}^{N_p}\hat{x}(k) + \hat{C}\hat{A}^{N_p-1}\hat{B}\Delta u(k) + \hat{C}\hat{A}^{N_p-2}\hat{B}\Delta u(k+1) \\
&\quad + \cdots + \hat{C}\hat{A}^{N_p-N_c}\hat{B}\Delta u(k+N_c-1)
\end{aligned} \tag{3.7}$$

Define the vectors Δu and y as

$$\begin{aligned}
\Delta u &= \begin{bmatrix}\Delta u(k)^T & \Delta u(k+1)^T & \ldots & \Delta u(k+N_c-1)^T\end{bmatrix}^T \\
y &= \begin{bmatrix}y(k+1|k)^T & y(k+2|k)^T & y(k+3|k)^T & \ldots & y(k+N_p|k)^T\end{bmatrix}^T
\end{aligned}$$

and with these, rewrite (3.7) into a compact matrix form as

$$y = F\hat{x}(k) + \Phi\Delta u \tag{3.8}$$

where

$$F = \begin{bmatrix}\hat{C}\hat{A} \\ \hat{C}\hat{A}^2 \\ \hat{C}\hat{A}^3 \\ \vdots \\ \hat{C}\hat{A}^{N_p}\end{bmatrix}, \quad \Phi = \begin{bmatrix}
\hat{C}\hat{B} & 0 & 0 & \ldots & 0 \\
\hat{C}\hat{A}\hat{B} & \hat{C}\hat{B} & 0 & \ldots & 0 \\
\hat{C}\hat{A}^2\hat{B} & \hat{C}\hat{A}\hat{B} & \hat{C}\hat{B} & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\hat{C}\hat{A}^{N_p-1}\hat{B} & \hat{C}\hat{A}^{N_p-2}\hat{B} & \hat{C}\hat{A}^{N_p-3}\hat{B} & \ldots & \hat{C}\hat{A}^{N_p-N_c}\hat{B}
\end{bmatrix}.$$

For a given set-point signal $r(k)$ at time k, within a prediction horizon the objective of the predictive control system is to bring the predicted output as close as possible to the set-point signal, where we assume that the set-point signal remains constant in the optimization window. Thus the objective is to find the best control parameters vector Δu such that an error function between the set-point and the predicted output is minimized.

With the control output weight z, and the reference r, the cost of the system can be written as

$$J = ||r - y||_2^2 + z||\Delta u||_2^2, \tag{3.9}$$

which gives the control objective

$$\min_{\Delta u} J = ||r - y||_2^2 + z||\Delta u||_2^2. \tag{3.10}$$

Combined with future prediction model (3.8) we get

$$\min_{\Delta u} J = ||r - F\hat{x} - \Phi\Delta u||_2^2 + z||\Delta u||_2^2 \tag{3.11}$$

and from the minimization

$$\frac{dJ}{d\Delta u} = 0 \Rightarrow (\Phi^T\Phi + zI)\Delta u = \Phi^T r - \Phi^T F\hat{x}, \tag{3.12}$$

it is given that

$$\Delta u = (\Phi^T\Phi + zI)^{-1}\Phi^T(r - F\hat{x}) \tag{3.13}$$

which can be separated into

$$\Delta u = Pr - K\hat{x} \tag{3.14}$$

where $P = (\Phi^T\Phi + zI)^{-1}\Phi^T$ and $K = (\Phi^T\Phi + zI)^{-1}\Phi^T F$. From (3.13) it is also seen that $(\Phi^T\Phi + zI)^{-1}\Phi^T$ and $(\Phi^T\Phi + zI)^{-1}\Phi^T F$ both depends only on the system parameters, hence are constant matrices.

Assumption 2.
It is assumed that the Hessian matrix $(\Phi^T\Phi + zI)^{-1}$ exists.

Even though Δu contains the predicted control signals for N_c steps ahead, since the calculation is made in every sample only the first $\Delta u(k)$ is used, which is called Receding Horizon Control since the horizon is always moving away. This ensures that the most recent

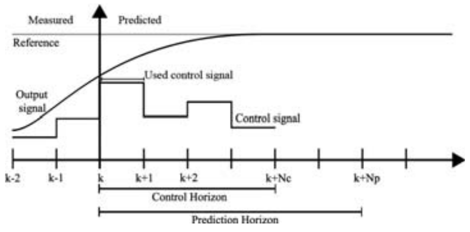


Fig. 3.1 : Receding Horizon Control

data is used, which gives a more precise control calculation and a faster response to new changes that might occur, depicted in Fig. 3.1.

Since we only take the first element of Δu, the state equation (3.4) can be written as

$$\begin{aligned}\hat{x}(k+1) &= \hat{A}\hat{x}(k) - \hat{B}e_1K\hat{x}(k) + \hat{B}e_1Pr(k) \\ &= (\hat{A} - \hat{B}e_1K)\hat{x}(k) + \hat{B}e_1Pr(k)\end{aligned} \tag{3.15}$$

where $e_1 = [I \;\; 0 \;\; 0 \;\; \ldots \;\; 0]$ eliminates all elements in K and P except for the first control sequence.

The control signal equation (3.14) will thus be implemented as

$$\Delta u = e_1Pr - e_1K\hat{x}. \tag{3.16}$$

3.2 System setup

The following system equations are acquired from the system shown in Fig. 3.2. The states x are the tie-line power flow deviation, ΔP_{tie_i}, frequency deviation, Δf_i, output of the gas turbine generator, ΔP_{g_i}, governor input of the gas turbine generator, Δx_{g_i}, output of the Battery Energy Storage System, ΔP_{E_i}, output of the thermal system, ΔP_{H_i}, and the demand, U_{AR_i}.

$$x_i = \begin{bmatrix} \Delta P_{tie_i} \\ \Delta f_i \\ \Delta P_{g_i} \\ \Delta x_{g_i} \\ \Delta P_{E_i} \\ \Delta P_{H_i} \\ U_{AR_i} \end{bmatrix}, \quad A_{ii} = \begin{bmatrix} 0 & -T_{ij} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{M_i} & -\frac{D}{M_i} & \frac{1}{M_i} & 0 & \frac{1}{M_i} & -\frac{1}{M_i} & 0 \\ 0 & 0 & -\frac{1}{T_{di}} & \frac{1}{T_{di}} & 0 & 0 & 0 \\ 0 & -\frac{1}{T_{gi}R_{gi}} & 0 & -\frac{1}{T_{gi}} & 0 & 0 & \frac{a_g}{T_g} \\ 0 & 0 & 0 & 0 & -\frac{1}{T_{Ei}} & 0 & \frac{a_E}{T_E} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{Hi}} & \frac{a_H}{T_H} \\ K_i & -B_iK_i & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A_{ij} = A_{ji} = \begin{bmatrix} 0 & T_{ij} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B_{ii} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{T_{gi}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{T_{Ei}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{T_{Hi}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C_{ii} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The off-diagonal matrices of B and C are zero.

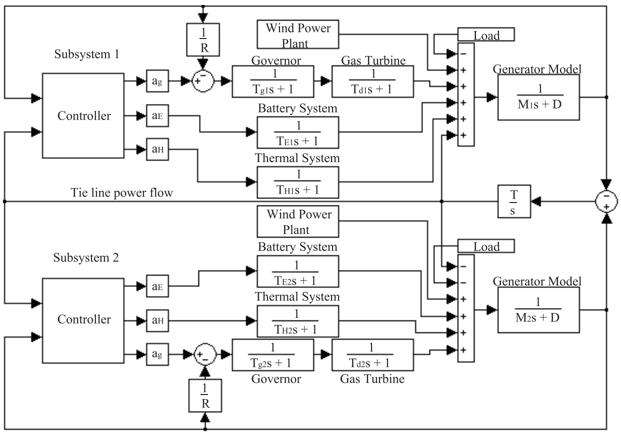


Fig. 3.2 : Model of small homogeneous power network

The setup shown in Fig. 3.2 shows two identical subsystems connected into a system. This model includes transfer functions for the different generators, and the frequency output is generated from the "Generator Model" transfer function. This system is based on the system in [14, 15]. Parameters used can be seen in Table 3.1 and below.

Table 3.1 : Power network parameters

Parameter	Symbol	Value	Unit
Inertia constant	M	0.2	puMW $\cdot$ s/Hz
Damping constant	D	0.26	puMW/Hz
Governor time constant	T_g	0.2	s
Gas turbine constant	T_d	5.0	s
BESS time constant	T_E	0.2	s
HP time constant	T_H	4.5	s
Regulation constant	R_g	2.5	Hz/puMW
Synchronizing coefficient	T_{ij}	0.50	puMW
Sampling time	T_s	0.1	s

The system capacity distribution for the gas turbine, BESS and thermal system are set to

$$a_g = 0.80, \quad a_H = 0.15, \quad a_E = 0.05,$$

and since the output of the systems are Δf, which we want to have as close to zero as possible, the control reference is set to

$$r = 0.$$

The MPC is implemented as a centralized and decentralized controller with the model specific parameters

$$N_p = 100, \quad N_c = 10, \quad z = 10^8.$$

Also included as a reference value is an iterative distributed controller, which was implemented with the controller specific parameters

$$n = 5, \quad Q = 0.5 \cdot I, \quad R = [1\ 1\ 1\ 4\ 0.02\ 0.01\ 1].$$

3.3 Results

The results show the response to a load frequency change of 0.1 Hz at the time 0.1s. The results from the different simulation setups can be seen below in Fig. 3.3 - 3.7.

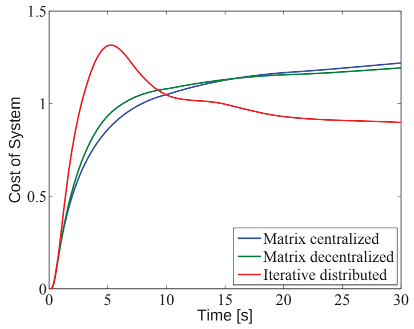


Fig. 3.3 : Zoomed system cost

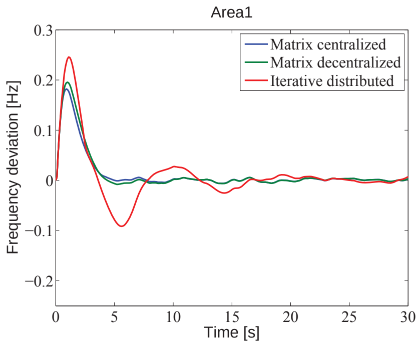


Fig. 3.4 : Zoomed frequency deviation

In Fig. 3.3 - 3.4 it can be seen that while the iterative method has a better final cost, it does have a bigger initial cost and a lot higher frequency deviation than the MPC method. It can also be seen that the centralized controller is slightly better than the decentralized one.

Fig. 3.5 - 3.6 shows these same result in a different way, showing the summarization of the first 30 seconds of the simulations.

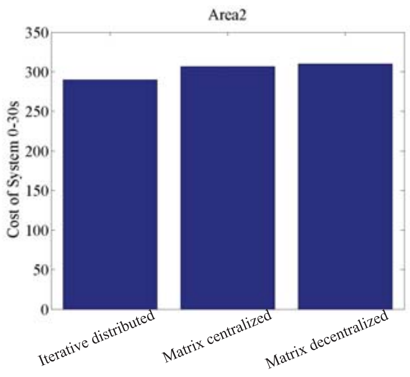


Fig. 3.5 : System cost comparison

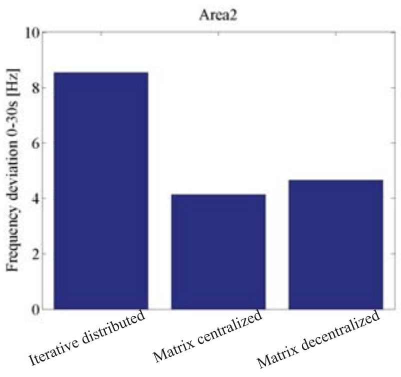


Fig. 3.6 : Frequency deviation comparison

Fig. 3.7 shows the reason why the centralized controller passes the decentralized controller in the last few seconds in Fig. 3.3. The controller is slightly out of tune, so the control signal for the battery system in area 1 starts to drift.

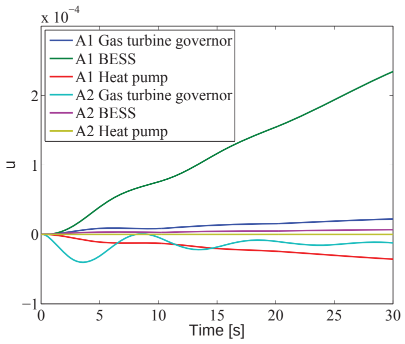


Fig. 3.7 : Matrix MPC controller output

One of the big downsides of the iterative method is that it requires extensive online calculations, thus slowing it down, whereas the MPC can make all calculations offline. The runtime for the two different methods are shown in Table 3.2.

As can be seen, the MPC only takes around one second, whereas the iterative method takes a few minutes.

Table 3.2 : Runtime for a 60 seconds simulation

MPC	Iterative
1.05s	17.65m

The reason why these results were never improved was because a homogeneous system is very unlikely to appear in a real situation. Also, the controller was changed to a different MPC method.

The reason why there was no distributed implementation was also due to the system setup. With only two subsystems the distributed controllers would simply be two centralized controllers only controlling their own subsystem.

Chapter 4

Distributed state space model predictive control

4.1 Model predictive controller

To specify how many steps ahead the controller should take into account, the prediction horizon and control horizon are defined the same as in the previous control method.

Definition 2.
We denote the prediction horizon as N_p and the control horizon as N_c.
The prediction horizon defines how many steps ahead the controller should predict the states.
The control horizon dictates the number of steps the controller should try to complete the control objective in.

Also, the control horizon to be limited such that problems won't occur in the calculations.

Assumption 3.
It is assumed that the control horizon N_c is chosen to be less than or equal to the prediction horizon N_p

$$N_c \leq N_p$$

since it is not possible to predict a control trajectory without having predicted the states at that time instant.

Based on the state space model (2.1), the future state variables are calculated sequentially using the future control parameters. By substituting the previous row into the next one, we can get a predicted state estimate at a certain time with calculations only depending on the current states $x(k)$ and the control input u.

$$\begin{aligned}
x(k+1) &= Ax(k) + Bu(k) \\
x(k+2) &= Ax(k+1) + Bu(k+1) \\
&= A^2x(k) + ABu(k) + Bu(k+1) \\
&\vdots \\
x(k+N_p) &= A^{N_p}x(k) + A^{N_p-1}Bu(k) + A^{N_p-2}Bu(k+1) \\
&+ \cdots + A^{N_p-N_c}Bu(k+N_c-1)
\end{aligned} \tag{4.1}$$

From the above equation and the original state space model (2.1) we can get the predicted output variables, by substitution, so all predicted variables are formulated in terms of current state variable information $x(k)$ and the future control movement $u(k+t)$, where $t = 0, 1, \ldots, N_c - 1$.

$$\begin{aligned}
y(k+1) &= CAx(k) + CBu(k) \\
y(k+2) &= CA^2x(k) + CABu(k) + CBu(k+1) \\
y(k+3) &= CA^3x(k) + CA^2Bu(k) + CABu(k+1) + CBu(k+2) \\
&\vdots \\
y(k+N_p) &= CA^{N_p}x(k) + CA^{N_p-1}Bu(k) + CA^{N_p-2}Bu(k+1) \\
&+ \cdots + CA^{N_p-N_c}Bu(k+N_c-1)
\end{aligned} \tag{4.2}$$

Rearranging these into matrices thus gives the system as

$$\underbrace{\begin{bmatrix} x(k+1) \\ x(k+2) \\ x(k+3) \\ \vdots \\ x(k+N_p) \end{bmatrix}}_{\underset{\rightarrow}{x}} = \underbrace{\begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^{N_p} \end{bmatrix}}_{P_x} x(k) + \underbrace{\begin{bmatrix} B & 0 & 0 & \ldots & 0 \\ AB & B & 0 & \ldots & 0 \\ A^2B & AB & B & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ A^{N_p-1}B & A^{N_p-2}B & A^{N_p-3}B & \ldots & A^{N_p-N_c}B \end{bmatrix}}_{H_x} \underbrace{\begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \\ \vdots \\ u(k+N_c-1) \end{bmatrix}}_{\underset{\rightarrow}{u}} \tag{4.3}$$

and

$$\underbrace{\begin{bmatrix} y(k+1) \\ y(k+2) \\ y(k+3) \\ \vdots \\ y(k+N_p) \end{bmatrix}}_{\underset{\rightarrow}{y}} = \underbrace{\begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}}_{P_y} x(k) + \underbrace{\begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix}}_{H_y} \underset{\rightarrow}{u} \tag{4.4}$$

giving the prediction system

$$\begin{aligned} \underset{\rightarrow}{x} &= P_x x(k) + H_x \underset{\rightarrow}{u} \\ \underset{\rightarrow}{y} &= P_y x(k) + H_y \underset{\rightarrow}{u}. \end{aligned} \tag{4.5}$$

Using the cost function

$$J = \underset{\rightarrow}{x}^T Q \underset{\rightarrow}{x} + \underset{\rightarrow}{u}^T R \underset{\rightarrow}{u} \tag{4.6}$$

where $Q \geq 0$ and $R > 0$ are the weighting matrices, the minimization in regards to $\underset{\rightarrow}{u}$ using the prediction system from (4.5) becomes

$$\min_{\underset{\rightarrow}{u}} J = (P_x x(k) + H_x \underset{\rightarrow}{u})^T Q (P_x x(k) + H_x \underset{\rightarrow}{u}) + \underset{\rightarrow}{u}^T R \underset{\rightarrow}{u}, \tag{4.7}$$

and from the minimization that the derivative should be zero, we get that

$$\frac{dJ}{d\underset{\rightarrow}{u}} = 0 \Rightarrow -(H_x^T Q H_x + R)\underset{\rightarrow}{u} = H_x^T Q P_x x(k). \tag{4.8}$$

From this it is given that the optimal control law is

$$\underset{\rightarrow}{u} = -(H_x^T Q H_x + R)^{-1} H_x^T Q P_x x(k) = -Kx(k) \tag{4.9}$$

where $K = (H_x^T Q H_x + R)^{-1} H_x^T Q P_x$. From (4.3) and (4.9) it can also be seen that K only depends on the system parameters, hence is a constant matrix that can be calculated offline.

Assumption 4.
It is assumed that the inverse Hessian matrix $(H_x^T Q H_x + R)^{-1}$ exists.

Assumption 4 is in theory always fulfilled, due to the condition that $R > 0$ makes it impossible for the Hessian matrix to be zero, since the first term can not be negative to take out the second term due to $Q \geq 0$ and H_x is squared. But in reality it is possible to get a non existent inverse, since the tools used to calculate the inverse has limitations. If the values of the Hessian matrix are too big, the inverse will be so small the calculation software might truncate the values to zero or simply give an error.

To ensure that the most recent data is used, which gives a more precise control calculation and a faster response to new changes that might occur, even though $\underset{\rightarrow}{u}$ contains the predicted control signals for N_c steps ahead, the calculation is made in every sample so only the first $u(k)$ is used, as shown in Fig. 3.1.

So from the complete set of predicted control signals

$$\underset{\rightarrow}{u} = \begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \\ \vdots \\ u(k+N_c) \end{bmatrix} \tag{4.10}$$

we only want the most relevant control signals for the next control correction

$$u(k) = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n_u} \end{bmatrix}, \tag{4.11}$$

n_u being the number of control signals to the plant.

Since we only take the first element of $\underset{\rightarrow}{u}$, we can write the control signal as

$$u(k) = e_I \underset{\rightarrow}{u} = -e_I K x(k), \tag{4.12}$$

where $e_I = [\underbrace{1\ 1\ \ldots\ 1}_{n_u}\ 0\ 0\ \ldots\ 0]$ eliminates all elements in K except for the first control sequence.

Thus the state equation can be written as

$$\begin{aligned} x(k+1) &= Ax(k) - Be_I K x(k) \\ &= (A - Be_I K)x(k). \end{aligned} \tag{4.13}$$

4.2 Constraints

Due to the presence of constraints, there is a need for an algorithm to recalculate the control action if it conflicts with the constraints.

For this purpose a Quadratic Programming solver is used to recalculate the control signal in case of conflicts with constraints, which optimizes the problem on the form

$$\min_u (f^T u + \frac{1}{2} u^T H u) \tag{4.14}$$

under the constraints such that

$$A_{QP} u \leq b, \tag{4.15}$$

where in this case u is the control signal $\underset{\rightarrow}{u}$, and H and f are from the optimal control law (4.9),

$$H = H_x^T Q H_x + R, \quad f = H_x^T Q P_x x(k), \tag{4.16}$$

where f depends on the current state values, thus is time-varying.

The constraints are formulated into A_{QP}, which is a matrix of linear constraint coefficients, and b, which is a time-varying vector. Constraints on the control signal, for example

$$-0.5 \leq \underset{\rightarrow}{u} \leq 0.5 \tag{4.17}$$

would be rearranged into

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} \underset{\rightarrow}{u} \leq \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}. \tag{4.18}$$

Constraints on the output signal, such as

$$-0.2 \leq \underset{\rightarrow}{y} \leq 0.2 \tag{4.19}$$

needs to be rewritten in terms of u, which by using (4.5),

$$\underset{\rightarrow}{y} = P_y x(k) + H_y \underset{\rightarrow}{u}, \tag{4.20}$$

becomes

$$-0.2 \leq P_y x(k) + H_y \underset{\rightarrow}{u} \leq 0.2. \tag{4.21}$$

Then by splitting (4.21) into two parts

$$\begin{aligned} -0.2 &\leq P_y x(k) + H_y \underset{\rightarrow}{u} \\ P_y x(k) + H_y \underset{\rightarrow}{u} &\leq 0.2 \end{aligned} \tag{4.22}$$

and then rearranging, gives the boundaries as

$$\begin{bmatrix} -H_y \\ H_y \end{bmatrix} \underset{\rightarrow}{u} \leq \begin{bmatrix} 0.2 + P_y x(k) \\ 0.2 - P_y x(k) \end{bmatrix}. \tag{4.23}$$

Constraints on the states would be rewritten the same way as constraints on the output signal, but using $\underset{\rightarrow}{x}$ from (4.5) instead of $\underset{\rightarrow}{y}$.

By combining the constrains on the control signal (4.18) and the constraints on the output signal (4.23), we get the complete constraints matrix (4.15) as

$$\underbrace{\begin{bmatrix} -1 \\ 1 \\ -H_y \\ H_y \end{bmatrix}}_{A_{QP}} \underset{\rightarrow}{u} \leq \underbrace{\begin{bmatrix} 0.5 \\ 0.5 \\ 0.2 + P_y x(k) \\ 0.2 - P_y x(k) \end{bmatrix}}_{b}. \tag{4.24}$$

From this we can use the Quadratic Programming solver to get the new optimized control signal u from (4.14) under the constraints from (4.24).

This will implement the constraints as hard constraints, Fig. 4.1, where as a lower constraint value and soft constraints, Fig. 4.2, might also be an reasonable alternative.

Hard constraints are an absolute block, not letting anything pass beyond its boundary. Whereas the soft constraints has a buffer zone where it can pass into as long as it does not pass the outer boundary and it recovers into the neutral zone as fast as possible, not staying in the danger zone too long.

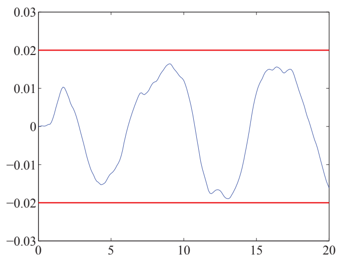


Fig. 4.1 : Hard constraints

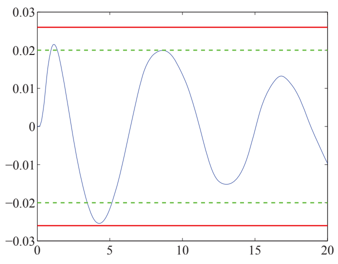


Fig. 4.2 : Soft constraints

4.3 State estimator

Since the plants only output is the frequency deviation, and the above described state space MPC uses all the states, a state observer is needed.

Currently the simulation operates with a Kalman Filter to estimate the states from the output and the tie-line values, using a model of the system linearized around the operating point.

For this a state estimate with Kalman filter is implemented as

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + K_f(y(k) - C\hat{x}(k)), \tag{4.25}$$

where K_f is given by

$$K_f = (APC^T)(CPC^T + R_f)^{-1} \tag{4.26}$$

in which R_f contains the weights and P is the symmetric positive semidefinite solution of the algebraic Riccati equation

$$AP + PA^T - (PC^T)R_f^{-1}(PC^T)^T = 0. \tag{4.27}$$

The Kalman filters that is implemented is slightly of tune currently and the state estimation diverts from the real value in some cases, with the possibility that the system then goes out of control. One of the few positive aspects of this is that it is possible to see how big error the controller can deal with and still keep a stable output from the plant.

Since there now are more subsystems, consequently there are more Kalman filters, and thus more parameters to try to tune. The decentralized controller has one Kalman filter for each subsystem, the centralized controller uses these to build up a complete estimate of the system, and each of the distributed controller uses the ones they needs for the region it controls. Since all these are of different sizes with different inputs and outputs, they needs to be tuned differently, which is an inconvenient and tedious task thats needs to be done every time the layout of the system changes shape.

Alternatively a Moving Horizon Estimator could be used, using a similar technique as the Model Predictive Control.

The downside with Moving Horizon Estimation is that it requires online optimization, thus slowing down the overall performance of the controller and using up computation resources. Another reason why Moving Horizon Estimation is not as used in industry as

Kalman Filter is because it requires more user experience to set up properly. Some of the benefits are that it can incorporate state constraints and by increasing the horizon length the performance of the estimation can be improved.

4.4 Connection topology

The first subsystem connection topology that was implemented was a linear one, which can be seen in Fig. 4.3.

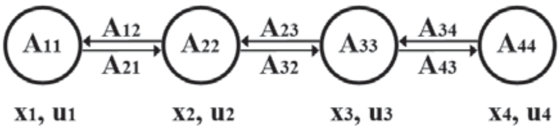
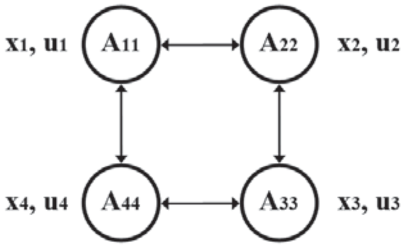


Fig. 4.3 : Linear connection

This was later extended on, and two new implemented connection topology can be seen below in Fig. 4.4 and Fig. 4.5.



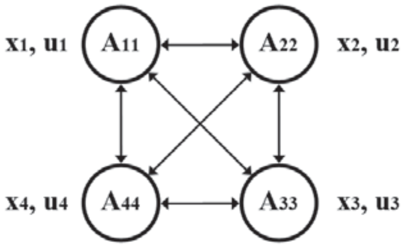


Fig. 4.4 : Square connection

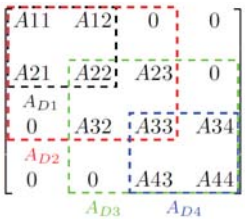
Fig. 4.5 : Full connection

The topology in Fig. 4.3 would mostly correlate with longer connections, like the connection between cities, while the topology in Fig. 4.4 and Fig. 4.5 more correlate to connections with less distance between, like major power distributor inside a city, where it is easier to install more connections.

4.5 Controller implementation

The controllers was here implemented both as separate decentralized controllers, as a centralized controller, and as distributed controllers.

The distributed control model were split into sections that encompass data that each subsystem have access to. The matrices for each distributed control section can then be taken from the complete power network system model, as shown below on the A matrix (4.28) for the linear connection topology.

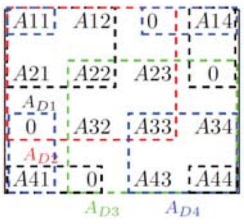
$A =$  (4.28)

Due to the changes in topology, the A matrix need to be changed to include the new connections. In the system with full connected topology, all subsystem is connected to each other, thus changing the A matrix to

$$A = \begin{bmatrix} A11 & A12 & A13 & A14 \\ A21 & A22 & A23 & A24 \\ A31 & A32 & A33 & A34 \\ A41 & A42 & A43 & A44 \end{bmatrix}. \tag{4.29}$$

This results in that each distributed controller essentially is a centralized controller that only outputs its control signal to one subsystem.

For the square connected system topology, only one connection between subsystem 1 and subsystem 4 has been introduced, thus changing the A matrix to

$A =$  . (4.30)

The distributed controllers A matrices being

$$\begin{aligned} A_{D1} &= \begin{bmatrix} A11 & A12 & A14 \\ A21 & A22 & 0 \\ A41 & 0 & A44 \end{bmatrix}, \quad A_{D2} = \begin{bmatrix} A11 & A12 & 0 \\ A21 & A22 & A23 \\ 0 & A32 & A33 \end{bmatrix}, \\ A_{D3} &= \begin{bmatrix} A22 & A23 & 0 \\ A32 & A33 & A34 \\ 0 & A43 & A44 \end{bmatrix}, \quad A_{D4} = \begin{bmatrix} A11 & 0 & A14 \\ 0 & A33 & A34 \\ A41 & A43 & A44 \end{bmatrix}. \end{aligned} \tag{4.31}$$

4.6 Data acquisition

The distributed controllers for the two subsystems implementation assumes that the tie-line power flow deviation ΔP_{tie_i} is known. From this and the plants own frequency deviation Δf_i, the connected subsystem frequency deviation Δf_j is calculated, as shown in (4.32), and then all states of the connected subsystem are estimated with a Kalman filter.

$$\Delta f_j = \frac{P_{tie_i}}{T_{ij}} - \Delta f_i \tag{4.32}$$

This method still works good when connected to one other subsystem, like $Area1$ and $Area4$ in the linear topology, but the two other, $Area2$ and $Area3$, have connections to two other subsystems. And since each subsystems contribution is not known, it can be assumed that both contributes equally, changing (4.32) into (4.33).

$$\Delta f_j = \Delta f_k = \frac{\frac{P_{tie_i}}{T_{ij}} - \Delta f_i}{2} \tag{4.33}$$

This does not give a satisfactory result, since the subsystems are of different setup and subsequently would not contribute equally.

Since the contribution from each subsystem is needed to make a Kalman estimation to then calculate a correct control action, a different method is needed.

The alternative method that the distributed controller implementation currently uses is a slightly time delayed value of the real value, as if the power plants shares its information with the other plants over for example an internet connection. So instead of calculating with a probably wrong estimate, the controller now has a correct, although slightly old, estimate of the states from the connected subsystems. As long as the time delay is not too long, or some big changes happens to the connected subsystems, the calculated control signal is accurate enough to give a good result.

In cases with long distances or slow information transfer, where the time delay might become too great, it would be reasonable to time stamp the information when sending it, so that the receiving controller can check if it is relevant. If the information is too old, it can instead use a previous calculated control signal from $\underset{\rightarrow}{u}$ that used relevant information, or ignore the connected subsystem altogether and calculate a decentralized control signal instead.

4.7 System setup

The setup shown in Fig. 4.7 - Fig. 4.9 shows the four subsystems connected into a system, each having different connection topologies, and the following system equations are acquired from it. The x, B and C matrices are the same for the different connection topology, whereas the A matrix differs. The states x are the tie-line power flow deviation ΔP_{tie_i}, frequency deviation Δf_i, output of the gas turbine generator ΔP_{g_i}, governor input of the gas turbine generator Δx_{g_i}, output of the Battery Energy Storage System ΔP_{E_i}, output of the thermal system ΔP_{H_i} and the demand U_{AR_i}.

Area 1 is set up with all generators present. Area 2 only has the battery system and thermal system. Area 3 has gas and thermal system. Area 4 has gas and battery system. Since wind power is a non-controllable generator source it is not included in the system model, but instead is modeled as an added noise source in the simulation model.

Parameters used can be seen in Table 4.1 and below.

$$x_1 = \begin{bmatrix} \Delta P_{tie_1} \\ \Delta f_1 \\ \Delta P_{g_1} \\ \Delta x_{g_1} \\ \Delta P_{E_1} \\ \Delta P_{H_1} \\ U_{AR_1} \end{bmatrix}, \quad x_2 = \begin{bmatrix} \Delta P_{tie_2} \\ \Delta f_2 \\ \Delta P_{E_2} \\ \Delta P_{H_2} \\ U_{AR_2} \end{bmatrix}, \quad x_3 = \begin{bmatrix} \Delta P_{tie_3} \\ \Delta f_3 \\ \Delta P_{g_3} \\ \Delta x_{g_3} \\ \Delta P_{H_3} \\ U_{AR_3} \end{bmatrix}, \quad x_4 = \begin{bmatrix} \Delta P_{tie_4} \\ \Delta f_4 \\ \Delta P_{g_4} \\ \Delta x_{g_4} \\ \Delta P_{E_4} \\ U_{AR_4} \end{bmatrix}$$

$$A_{11} = \begin{bmatrix} 0 & -T_{12} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{M_1} & -\frac{D}{M_1} & \frac{1}{M_1} & 0 & \frac{1}{M_1} & -\frac{1}{M_1} & 0 \\ 0 & 0 & -\frac{1}{T_{d1}} & \frac{1}{T_{d1}} & 0 & 0 & 0 \\ 0 & -\frac{1}{T_{g1}R_{g1}} & 0 & -\frac{1}{T_{g1}} & 0 & 0 & \frac{a_g}{T_g} \\ 0 & 0 & 0 & 0 & -\frac{1}{T_{E1}} & 0 & \frac{a_E}{T_E} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H1}} & \frac{a_H}{T_H} \\ K_1 & -B_1K_1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad A_{12} = \begin{bmatrix} 0 & T_{21} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B_{11} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{T_{g1}} & 0 & 0 \\ 0 & \frac{1}{T_{E1}} & 0 \\ 0 & 0 & \frac{1}{T_{H1}} \\ 0 & 0 & 0 \end{bmatrix}, \quad B_{22} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{T_{E2}} & 0 \\ 0 & \frac{1}{T_{H2}} \\ 0 & 0 \end{bmatrix}, \quad B_{33} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_{g3}} & 0 \\ 0 & \frac{1}{T_{H3}} \\ 0 & 0 \end{bmatrix}, \quad B_{44} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{T_{g4}} & 0 \\ 0 & \frac{1}{T_{E4}} \\ 0 & 0 \end{bmatrix}$$

$$C_{11} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad C_{22} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \end{bmatrix},$$

$$C_{33} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad C_{44} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The off-diagonal matrices of B and C are zero.

The system capacity distribution for the gas turbine, BESS and thermal system are set to

$$a_g = 0.80, \quad a_H = 0.15, \quad a_E = 0.05.$$

The SS-MPC controller uses the parameters

$$rw = [\rlap{$\overbrace{}^{A_{D2}}$}\rlap{$\underbrace{}_{A_{D1}}$}8\ \ 0.83\ \ 8\ \ \rlap{$\underbrace{}_{A_{D3}}$}0.83\ \ 8\ \ \overbrace{8\ \ 8\ \ 8\ \ 0.83}^{A_{D4}}], \quad N_c = 10,$$

$$N_p = 100, \quad R = rw * I, \quad Q = I.$$

While the MATLAB MPC Toolbox reference uses

$$rw = [\rlap{$\overbrace{}^{A_{D2}}$}\rlap{$\underbrace{}_{A_{D1}}$}8\ \ 0.083\ \ 0.000008\ \ \rlap{$\underbrace{}_{A_{D3}}$}0.0083\ \ 0.8\ \ \overbrace{10\ \ 0.0000001\ \ 8\ \ 0.83}^{A_{D4}}], \quad N_c = 10,$$

$$N_p = 100, \quad controlWeights = rw * I, \quad outputWeight = I.$$

The Kalman filter uses the weights

$$R = 0.008, \quad Q_1 = [5\ \ 10\ \ 5] * I, \quad Q_2 = [5\ \ 1] * I, \quad Q_3 = [5\ \ 5] * I, \quad Q_4 = [5\ \ 10] * I,$$

which are also split similarly to the weights for the subsystems depending on which generator the subsystem uses.

The implementations are set to fulfill the constraints as

$$-0.2 \leq y \leq 0.2.$$

Table 4.1 : Power network parameters

Parameter	Symbol	Value	Unit
Inertia constant	M	0.2	puMW $\cdot$ s/Hz
Damping constant	D	0.26	puMW/Hz
Governor time constant	T_g	0.2	s
Gas turbine constant	T_d	5.0	s
BESS time constant	T_E	0.2	s
HP time constant	T_H	4.5	s
Regulation constant	R_g	2.5	Hz/puMW
Synchronizing coefficient	T_{ij}	0.50	puMW
Sampling time	T_s	0.1	s

4.7.1 Two heterogeneous subsystems

Due to the high amount of tunable variables, a system with two subsystems have also been used. The smaller system makes it easier to make changes in the system and to implement other controller structures to use as a reference.

The system is shown in Fig. 4.6 and is based on the previous system shown in 3.2, the difference being that it is comprised of one subsystem with all generators and one subsystem without the gas generator.

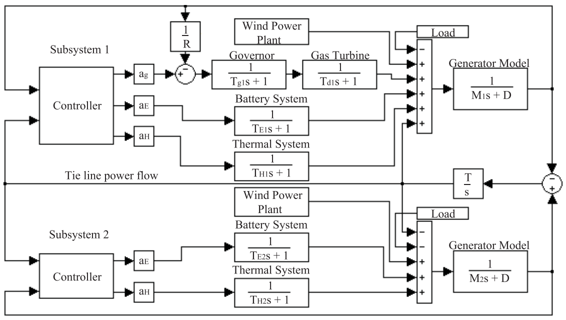


Fig. 4.6 : Model of power network

The system setup is similar to that of the system with four subsystems above. See above for the structure of x_1, x_2, B_{11}, B_{22}, C_{11}, C_{22} and the parameter table Table 4.1. The structure of A, which is a part of the one presented above, is shown below.

$$A = \left[\begin{array}{ccccccc|ccccc}
0 & -T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & T_{21} & 0 & 0 & 0 \\
\frac{1}{M_1} & -\frac{D}{M_1} & \frac{1}{M_1} & 0 & \frac{1}{M_1} & -\frac{1}{M_1} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{T_{d1}} & \frac{1}{T_{d1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{T_{g1}R_{g1}} & 0 & -\frac{1}{T_{g1}} & 0 & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{T_{E1}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H1}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 \\
K_1 & -B_1K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & -T_{21} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_2} & -\frac{D}{M_2} & \frac{1}{M_2} & -\frac{1}{M_2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E2}} & 0 & \frac{a_E}{T_E} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H2}} & \frac{a_H}{T_H} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & K_2 & -B_2K_2 & 0 & 0 & 0
\end{array}\right]$$

The system capacity distribution for the gas turbine, BESS and thermal system are set to

$$a_g = 0.80, \quad a_H = 0.15, \quad a_E = 0.05.$$

The controller is implemented similar to the system above with the parameters as

$$R = [\overbrace{8 \;\; 0.83 \;\; 8}^{A_{D1}} \overbrace{0.83 \;\; 8}^{A_{D2}}] * I, \quad Q = I, \quad N_c = 10, \quad N_p = 30.$$

The Kalman filter uses the weights

$$Q = [1 \;\; 1 \;\; 1 \;\; 5 \;\; 10 \;\; 5 \;\; 1] * I, \quad R = 1.$$

The implementations are set to fulfill the constraints as

$$-0.2 \leq y \leq 0.2.$$

Since the distributed controllers in this case are essentially two centralized controllers that only sends signals to their own subsystem, they will therefore give the same result as the centralized controller.

The result can be seen below in Fig. 4.24 - 4.25 in the result section.

4.7.2 Four heterogeneous subsystems

Below, the full A matrices and system setups are shown for the three different four subsystems topologies, the linear topology A_L, the square topology A_S, and the fully connected topology A_F.

The system models are shown in Fig. 4.7 - 4.9. These extends the previous model in Fig. 4.6 with two more subsystems. The only difference between these three models are the way the subsystems connects to each others, which in the A matrices are indicated as the difference on the off diagonal.

In the linear system in Fig. 4.7, the two subsystems in the middle has two neighbors each, thus they also has two incoming connections, hence the difference in input for only those two subsystems.

In the square connected system in Fig. 4.8, all subsystems connects to two neighbors, so the change in input that was shown for the two middle subsystems in the linear connected case will now be present at all inputs.

The connections for the fully connected system in Fig. 4.9 looks a bit different than the others, but is essentially the same as for the square connected, extended with the new connections, but the addition is made at a central place and then corrected at the input for each subsystem.

$$A_L = \left[\begin{array}{ccccccc|ccccc|cccccc|cccccc}
0 & -T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & T_{21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{M_1} & -\frac{D}{M_1} & \frac{1}{M_1} & 0 & \frac{1}{M_1} & -\frac{1}{M_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{T_{d1}} & \frac{1}{T_{d1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{T_{g1}R_{g1}} & 0 & -\frac{1}{T_{g1}} & 0 & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{T_{E1}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H1}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
K_1 & -B_1K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & T_{32} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_2} & -\frac{D}{M_2} & \frac{1}{M_2} & -\frac{1}{M_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E2}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H2}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & K_2 & -B_2K_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_{23} & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & 0 & T_{43} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_3} & -\frac{D}{M_3} & \frac{1}{M_3} & 0 & -\frac{1}{M_3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d3}} & \frac{1}{T_{d3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g3}R_{g3}} & 0 & -\frac{1}{T_{g3}} & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H3}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_3 & -B_3K_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_{34} & 0 & 0 & 0 & 0 & 0 & -T_{43} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_4} & -\frac{D}{M_4} & \frac{1}{M_4} & 0 & \frac{1}{M_4} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d4}} & \frac{1}{T_{d4}} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g4}R_{g4}} & 0 & -\frac{1}{T_{g4}} & 0 & \frac{a_g}{T_g} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E4}} & \frac{a_E}{T_E} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_4 & -B_4K_4 & 0 & 0 & 0 & 0
\end{array}\right]$$

$$A_S = \left[\begin{array}{ccccccc|ccccc|cccccc|cccccc}
0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & 0 & 0 & T_{21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_{41} & 0 & 0 & 0 & 0 \\
\frac{1}{M_1} & -\frac{D}{M_1} & \frac{1}{M_1} & 0 & \frac{1}{M_1} & -\frac{1}{M_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{T_{d1}} & \frac{1}{T_{d1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{T_{g1}R_{g1}} & 0 & -\frac{1}{T_{g1}} & 0 & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{T_{E1}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H1}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
K_1 & -B_1K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & T_{32} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_2} & -\frac{D}{M_2} & \frac{1}{M_2} & -\frac{1}{M_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E2}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H2}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & K_2 & -B_2K_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_{23} & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & 0 & T_{43} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_3} & -\frac{D}{M_3} & \frac{1}{M_3} & 0 & -\frac{1}{M_3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d3}} & \frac{1}{T_{d3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g3}R_{g3}} & 0 & -\frac{1}{T_{g3}} & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H3}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_3 & -B_3K_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & T_{34} & 0 & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_4} & -\frac{D}{M_4} & \frac{1}{M_4} & 0 & \frac{1}{M_4} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d4}} & \frac{1}{T_{d4}} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g4}R_{g4}} & 0 & -\frac{1}{T_{g4}} & 0 & \frac{a_g}{T_g} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E4}} & \frac{a_E}{T_E} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_4 & -B_4K_4 & 0 & 0 & 0 & 0
\end{array}\right]$$

$$A_F = \left[\begin{array}{ccccccc|ccccc|cccccc|cccccc}
0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & 0 & 0 & T_{21} & 0 & 0 & 0 & 0 & T_{31} & 0 & 0 & 0 & 0 & 0 & T_{41} & 0 & 0 & 0 & 0 \\
\frac{1}{M_1} & -\frac{D}{M_1} & \frac{1}{M_1} & 0 & \frac{1}{M_1} & -\frac{1}{M_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{T_{d1}} & \frac{1}{T_{d1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{T_{g1}R_{g1}} & 0 & -\frac{1}{T_{g1}} & 0 & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{1}{T_{E1}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H1}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
K_1 & -B_1K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{12} & 0 & 0 & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & T_{32} & 0 & 0 & 0 & 0 & 0 & T_{42} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_2} & -\frac{D}{M_2} & \frac{1}{M_2} & -\frac{1}{M_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E2}} & 0 & \frac{a_E}{T_E} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H2}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & K_2 & -B_2K_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{13} & 0 & 0 & 0 & 0 & 0 & 0 & T_{23} & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 & 0 & T_{43} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_3} & -\frac{D}{M_3} & \frac{1}{M_3} & 0 & -\frac{1}{M_3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d3}} & \frac{1}{T_{d3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g3}R_{g3}} & 0 & -\frac{1}{T_{g3}} & 0 & \frac{a_g}{T_g} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{H3}} & \frac{a_H}{T_H} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_3 & -B_3K_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & T_{14} & 0 & 0 & 0 & 0 & 0 & 0 & T_{24} & 0 & 0 & 0 & 0 & T_{34} & 0 & 0 & 0 & 0 & 0 & -\sum_{j\in\mathcal{N}_i} T_{ij} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{M_4} & -\frac{D}{M_4} & \frac{1}{M_4} & 0 & \frac{1}{M_4} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{d4}} & \frac{1}{T_{d4}} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{g4}R_{g4}} & 0 & -\frac{1}{T_{g4}} & 0 & \frac{a_g}{T_g} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{E4}} & \frac{a_E}{T_E} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_4 & -B_4K_4 & 0 & 0 & 0 & 0
\end{array}\right]$$

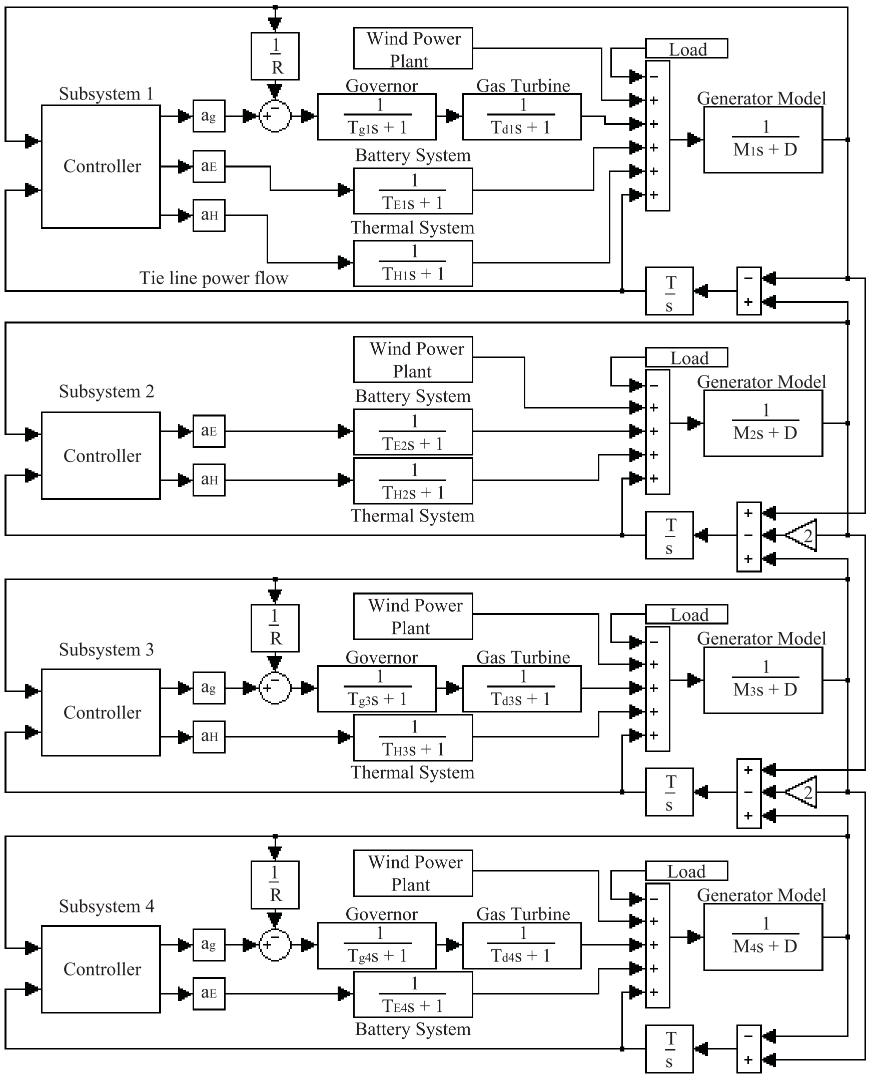


Fig. 4.7 : Model of linear connected power network

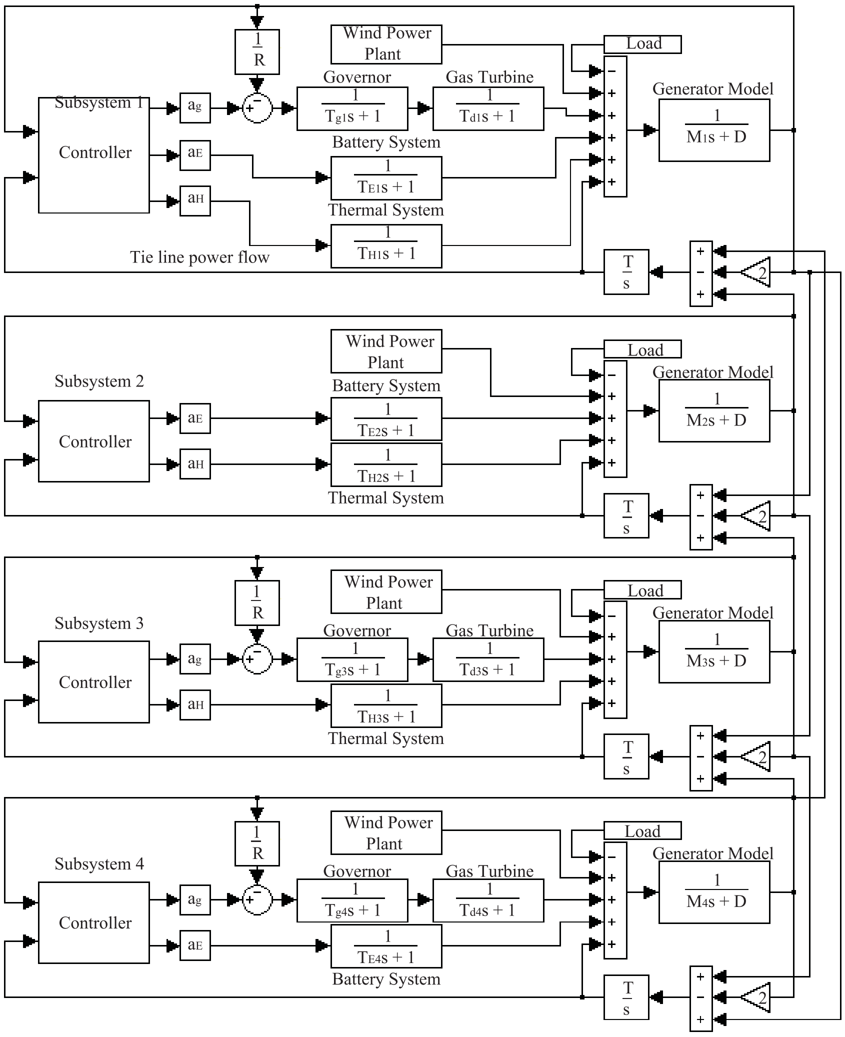


Fig. 4.8 : Model of square connected power network

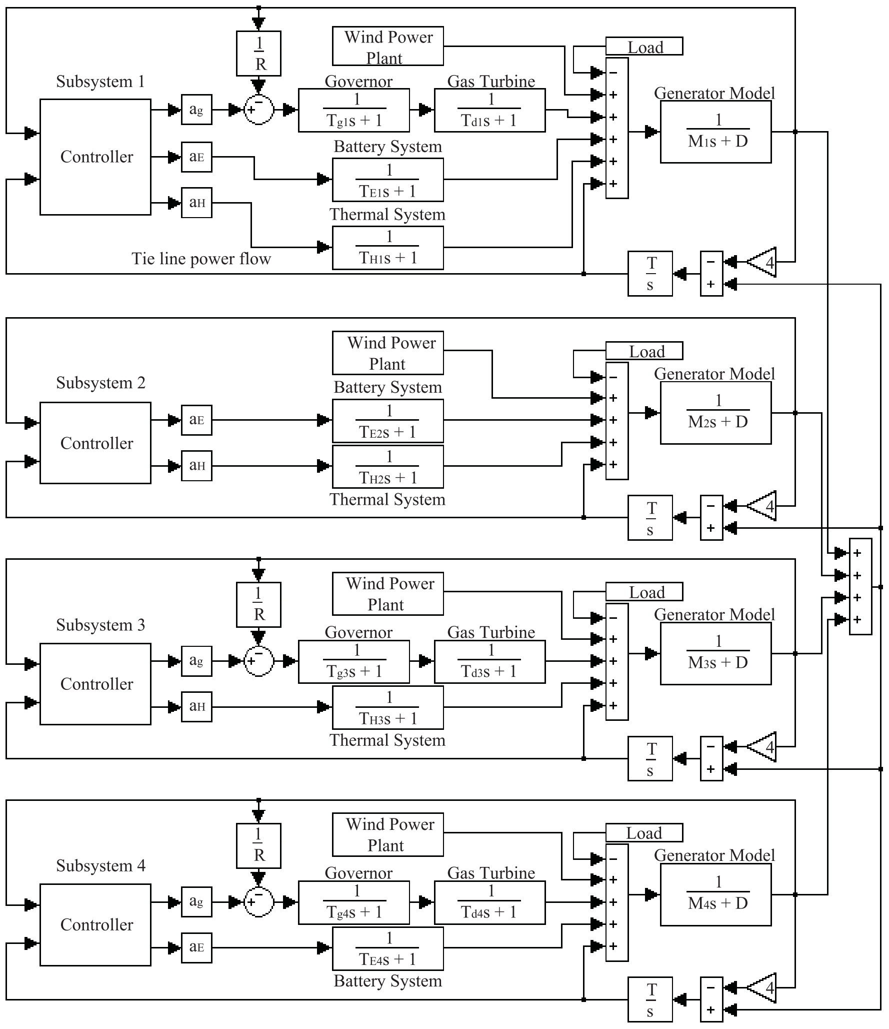


Fig. 4.9 : Model of full connected power network

4.8 Results

The results show the response to a load frequency change of 0.1 Hz at the time 0.1s. The new control method was first compared on the previous two homogeneous subsystems setup as a centralized and a decentralized controller. The comparison to the previous control method can be seen in Fig. 4.10 - 4.23. The result from the heterogeneous two subsystems setup is shown in Fig. 4.24 - 4.25. The results from the different simulation on the new four subsystems setups can be seen below in Fig. 4.26 - 4.42, and the result for the frequency deviation in more detail in Table 4.3.

4.8.1 Two homogeneous subsystems

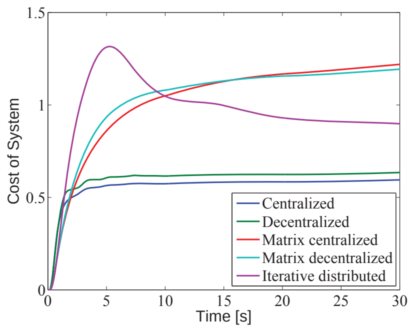


Fig. 4.10 : Zoomed system cost

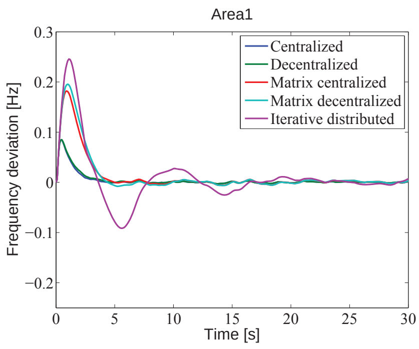


Fig. 4.11 : Zoomed frequency deviation

As can be seen in Fig. 4.10 - 4.13, the new MPC method is greatly superior to the previous one, giving both lower results in cost and in frequency deviation.

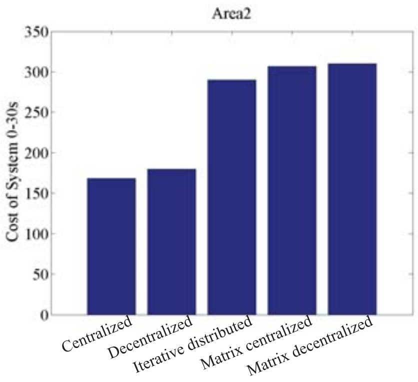


Fig. 4.12 : System cost comparison

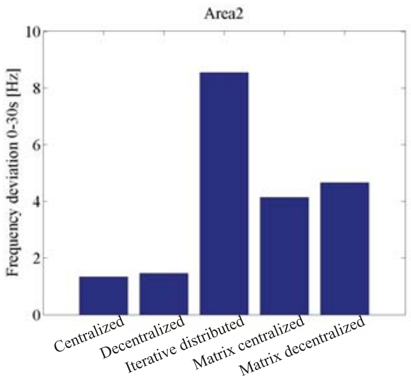


Fig. 4.13 : Frequency deviation comparison

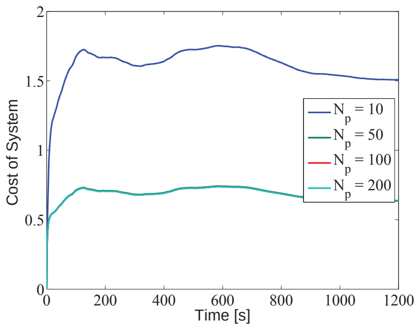


Fig. 4.14 : System cost for different N_p

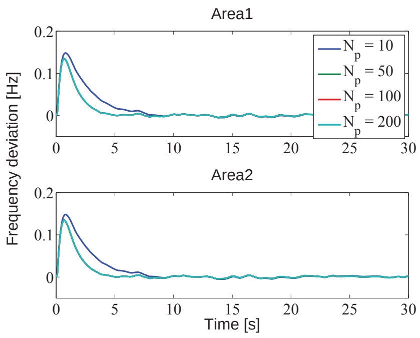


Fig. 4.15 : Frequency deviation for different N_p

Fig. 4.14 - 4.15 shows that although an increased prediction horizon N_p only has a slight impact on the frequency deviation, it has a large impact on the system cost to a certain point. In these cases the simulations with N_p at 50, 100 and 200 gives the same outcome. The trade-off being that the longer prediction horizon increases of size of the equation matrices, which in turn increases the calculation time.

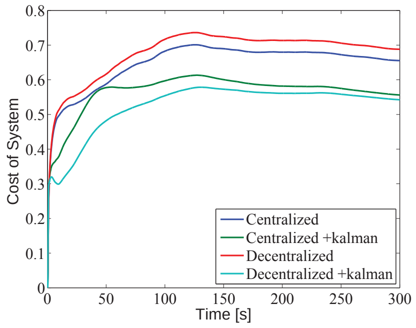


Fig. 4.16 : System cost

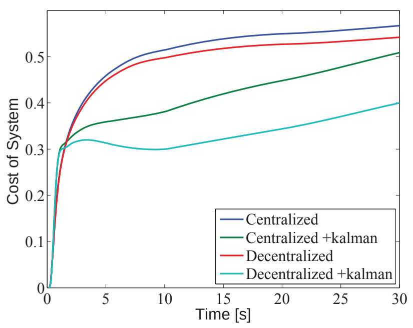


Fig. 4.17 : Zoomed system cost

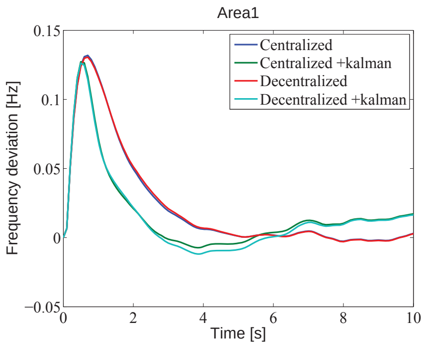


Fig. 4.18 : Zoomed frequency deviation

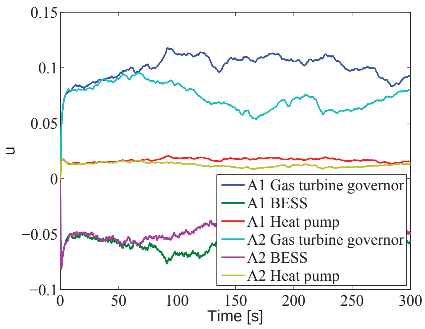


Fig. 4.19 : Controller output for decentralized

Fig. 4.16 - 4.18 shows that when the Kalman estimator is used instead of the real value, the result improves. This is due to the slight error that the estimator introduces, which in this case helps to improve the result.

Fig. 4.19 shows an example of the output from a controller in the case of a decentralized control structure.

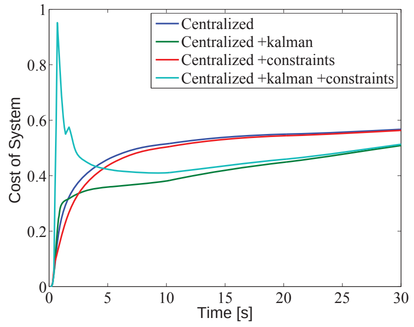


Fig. 4.20 : Cost with constraints

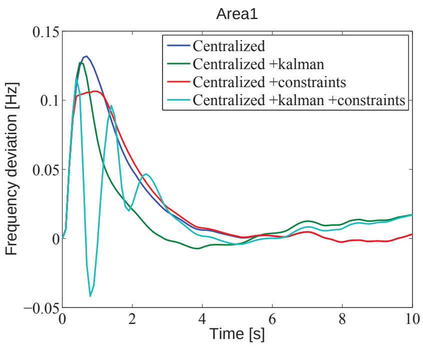


Fig. 4.21 : Frequency deviation with constraints

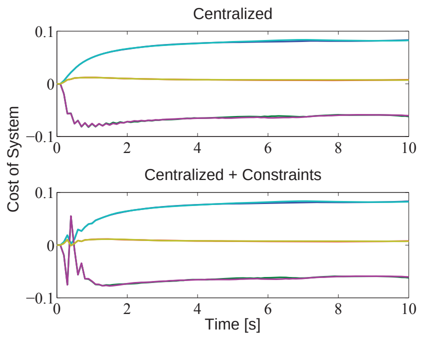


Fig. 4.22 : Controller output with constraints

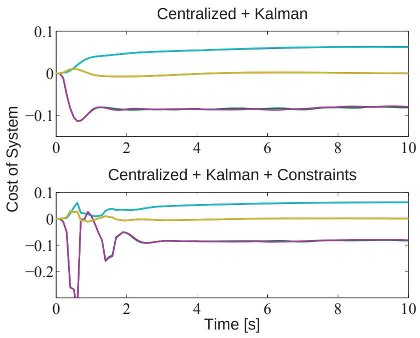


Fig. 4.23 : Controller output with constraints and Kalman filter

In Fig. 4.20 - 4.23 a constraint on the frequency deviation was introduced, set to keep the frequency deviation inside the range $-0.12 \leq y \leq 0.12$. Fig. 4.21 shows that this has been achieved, and the other three figures shows what implications this has on the cost and control signals of the system. Fig. 4.22 - 4.23 shows the cost for all of the generators, similar to Fig. 4.19.

When comparing the runtime for the state space MPC to the previous results, we can see from Table 4.2 that it takes slightly longer than the previous matrix MPC method, this is mostly due to the fact that if a constraint is met, a small online recalculation is needed.

Table 4.2 : Runtime for a two subsystem 60 seconds simulation

SS-MPC	M-MPC	Iterative
1.62s	1.05s	17.65m

4.8.2 Two heterogeneous subsystems

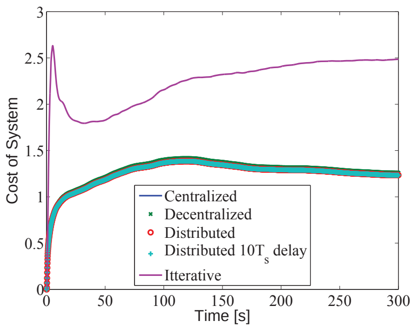


Fig. 4.24 : MPC system cost

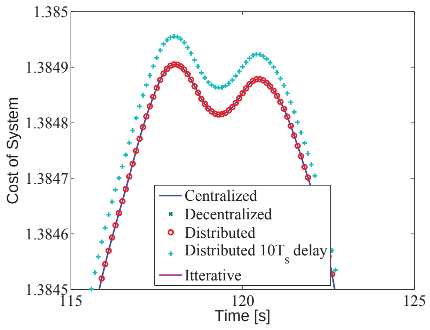


Fig. 4.25 : MPC cost zoomed

Fig. 4.24 - 4.25 shows that the controller performs very well even with a delayed signal, the delayed signal only slightly increases the cost while the frequency deviation stays almost the same as the direct signal controller.

4.8.3 Four heterogeneous subsystems

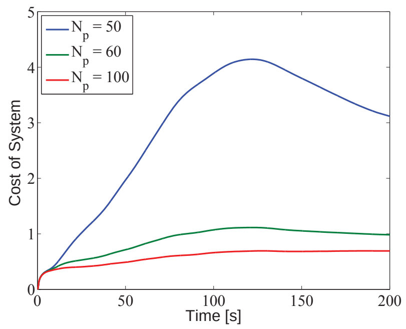


Fig. 4.26 : Centralized cost for different N_p

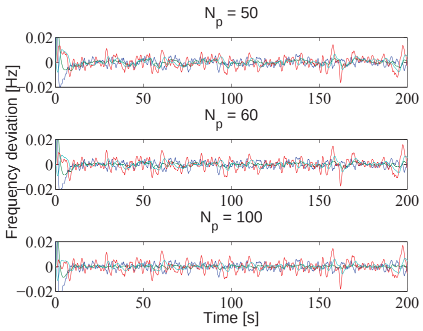


Fig. 4.27 : Centralized frequency deviation for different N_p

Fig. 4.26 - 4.27 shows, similar to the two subsystems setup, that although an increased prediction horizon N_p has almost no visible impact on the frequency deviation, it has a large impact on the system cost to a certain point. The trade-off being increased calculation time.

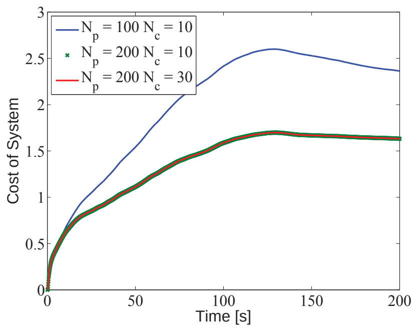


Fig. 4.28 : Distributed cost for different horizons

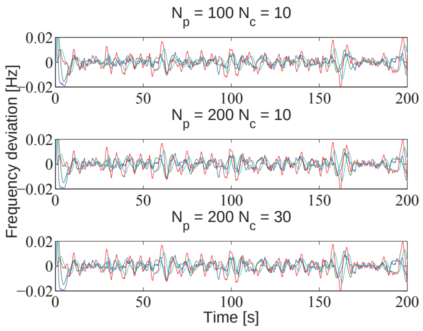


Fig. 4.29 : Distributed frequency deviation for different horizons

Fig. 4.28 - 4.29 shows that, while a change in N_p has a visible impact on the cost as shown in Fig. 4.26 - 4.27, a change in control horizon N_c gives no visible alteration to the

results. Thus a short control horizon can be chosen to improve the calculation time.

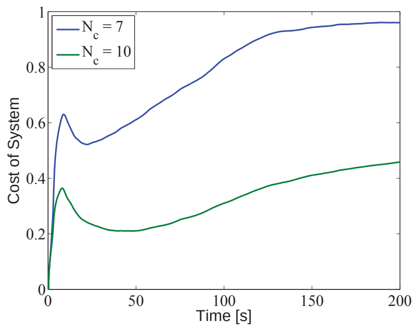


Fig. 4.30 : MPC Toolbox distributed cost

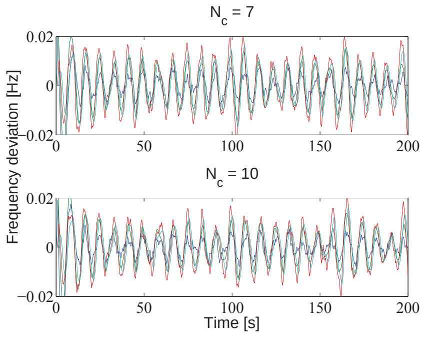


Fig. 4.31 : MPC Toolbox distributed frequency deviation

Fig. 4.30 - 4.31 shows the result of the MATLAB MPC Toolbox generated controller. While it has lower system cost, it has a more unstable frequency output. It also has a longer runtime due to the more complex algorithm used, finishing in 17.04s.

The simulations resulting in the results shown in Fig. 4.26 - 4.31 were done on a linear connected system.

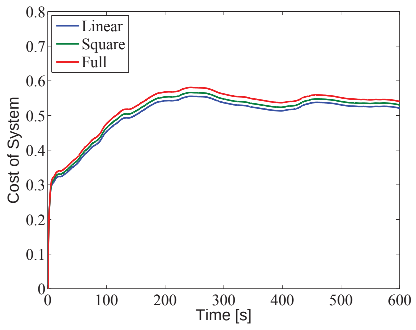


Fig. 4.32 : Centralized cost

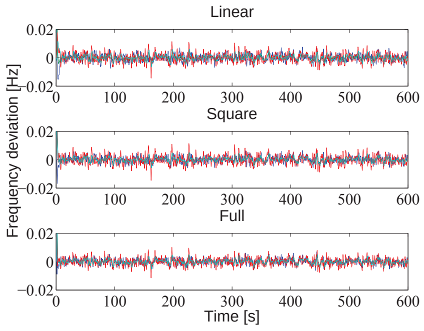


Fig. 4.33 : Centralized frequency deviation

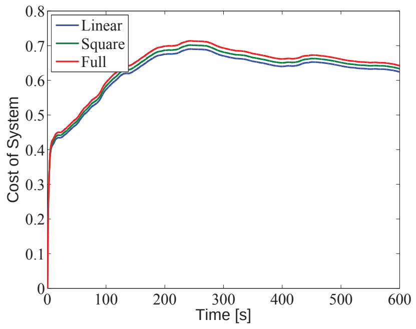


Fig. 4.34 : Decentralized cost

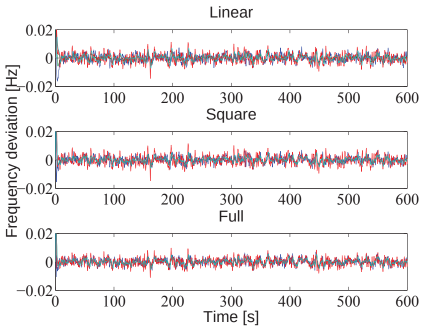


Fig. 4.35 : Decentralized frequency deviation

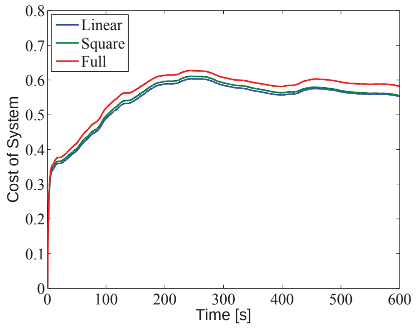


Fig. 4.36 : Distributed cost

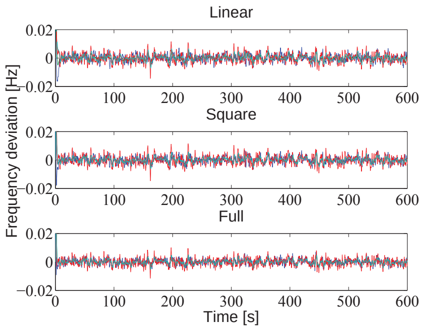


Fig. 4.37 : Distributed frequency deviation

Fig. 4.32 - 4.37 shows the effects that the topology has on the outcome for each controller type, while Fig. 4.38 - 4.43 shows the same results, but organized to show the results of the different controller on each topology.

The results showing the effect of the different topology, Fig. 4.32 - 4.37, shows that the system with fully connected topology has the highest cost, while the linear connected topology has the lowest cost.

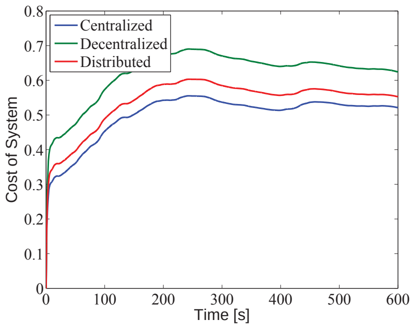


Fig. 4.38 : Linear cost

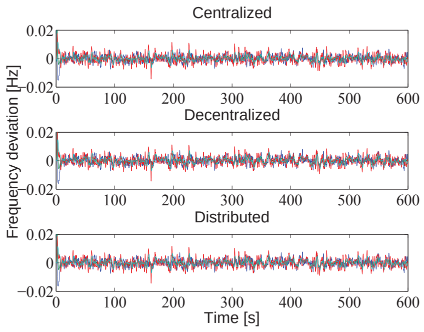


Fig. 4.39 : Linear frequency deviation

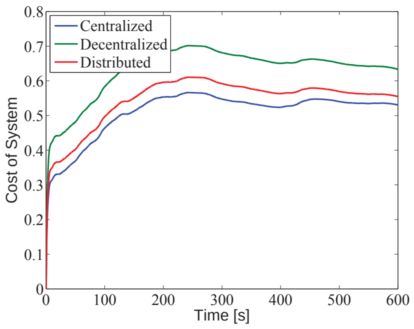


Fig. 4.40 : Square cost

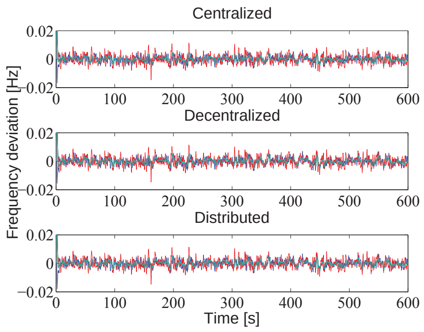


Fig. 4.41 : Square frequency deviation

The result of the different controllers on each topology, Fig. 4.38 - 4.43, shows the same outcome for each topology, the centralized controller has the lowest cost, the decentralized controller has the highest cost, and the distributed controller is in between the two others.

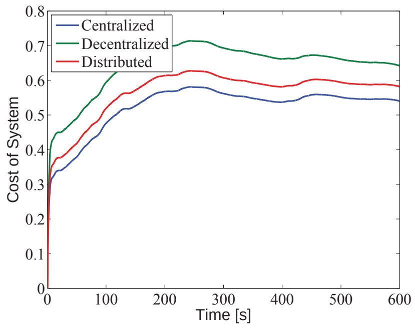


Fig. 4.42 : Full cost

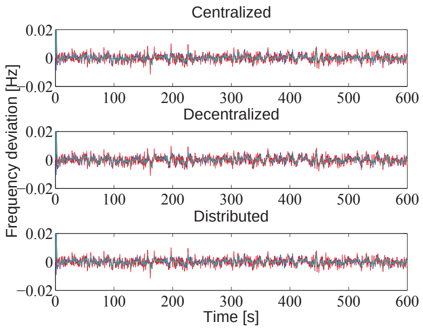


Fig. 4.43 : Full frequency deviation

Due to the fact that it is hard to see anything conclusive for the frequency deviation for the different setups since the difference is so small, except the fact that it never goes outside the predefined constraints, Table 4.3 shows the root mean square (RMS) values for the first 600 seconds.

The RMS is calculated as

$$RMS = \sqrt{\frac{1}{n}(\Delta f_1^2 + \Delta f_2^2 + \cdots + \Delta f_n^2)}.$$

From the table we can see that the effectiveness of the controller type holds true, that the centralized controller is the most effective and the decentralized is the least effective. In the regards of topology it shows that the higher the grade of connectivity there are, which leads to the controllers taking into account more of the full network, the better the outcome will be.

Table 4.3 : Frequency deviation RMS [HZ] (0-600s)

RMS	Linear	Square	Full
Centralized	$4.361 * 10^{-3}$	$3.640 * 10^{-3}$	$3.193 * 10^{-3}$
Distributed	$4.593 * 10^{-3}$	$3.901 * 10^{-3}$	$3.303 * 10^{-3}$
Decentralized	$4.917 * 10^{-3}$	$4.128 * 10^{-3}$	$3.646 * 10^{-3}$

The runtime for the four subsystems linear topology setup can be seen in Table 4.4, with the previous mentioned runtime for the MATLAB MPC Toolbox generated controller as comparison.

Table 4.4 : Runtime for a four subsystem 60 seconds simulation

SS-MPC	MPC Toolbox
2.31s	17.04s

The state space MPC does not include as many advanced subroutines and optimization calculations as the Toolbox generated one, thus the much faster runtime.

Chapter 5

Conclusion

In this paper, a distributed model predictive controller has been proposed for a network control architecture. The main feature of this concept is the possibility of implementing constraints and to include as much information as possible about the network without adding extra time delays due to long transmission distances. For solving these calculations the online complexity only consists of simple matrix multiplications, except in close encounters with a constraint, in which case the quadratic programming temporarily takes over to calculate a new control signal to avoid crossing the threshold. Also investigated were various connection topologies and their implications on the outcome of the control.

One of the big advantages of the proposed method is the calculation speed while the system is running. Although the proposed control algorithm might not be as versatile as the MATLAB MPC Toolbox generated controller, it is much faster due to its simplicity. The MPC Toolbox include features such as the ability to set constraints on almost any value, and the constraint can be configured as soft constraint with user set boundaries, and Kalman filter state estimator is also built in as standard. Similarly, the iterative gradient method also takes a lot longer time to calculate the control signal, since it will search for the most optimal value for each time instance, thus having almost all of the control calculation done online. This makes it a very adaptive method, that can take new factors into account easily without major configuration updates.

The main results of this research includes the following confirmed conclusions.

The distributed control setup is invariably more effective than the decentralized one, although it is less effective than the centralized alternative. The different control implementation have their own strong side. The decentralized controller is easy to implement and does not need to take into account any of the other parts of the system. The centralized is effective since it can take everything into account when calculating the control. But they also have downsides, the decentralized controller lacks overall efficiency since it only includes a small part of the network, and the centralized controller is impractical to implement in a real system due to the distances involved. The distributed controller can be seen as a compromise between the two other, taking some of the ease of implementation and efficiency while trying to keep the downsides as small as possible.

Another of the main results of this research is that a higher connectivity in the topology setup results in an increased performance. This is not only true for the distributed controllers that with higher connectivity takes more of the network into account in their control calculation, but also for the two other control implementations, where the centralized controller always takes everything into account and the decentralized controller only takes its own part into account and disregards everything else. Thus it stands to reason that a higher connectivity helps to stabilize the system and cancel out the unwanted frequency deviation.

It is also shown that a delayed measurement signal to the distributed controller only gives a slightly worse result than the direct signal. And in a real life implementation where measurements data would be time stamped for relevancy check when longer transfer times occurs, at its extreme it can be no worse than a decentralized controller since it would disregard irrelevant data.

The theoretical investigation of these claims are up to further research.

Appendix A

Matlab files

A.1 m-files

These files shows the system setup, controller creation and the online control algorithm.

A.1.1 Parameter setup file

setup_mpc.m

```
clc
clear
%-------------------------------------------------

%topology type (1=linear, 2=square, 3=full)
toptype = 1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

H1=5;
H2=5;
H3=5;
H4=5;
D=0.26;
```

```
Td=5;
Tg=0.2;
TH=4.5;
TE=0.2;
Rg=2.5;
RE=2.5;
RH=2.5;
f=50;
M1=2*H1/f;
M2=2*H2/f;
M3=2*H3/f;
M4=2*H4/f;
T12=0.50;
a1=0.80;
a2=0.05;
a3=0.15;
K1=1.1;
B1=1/Rg+D;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Horizons %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Nc=10;
Np=100;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% weights %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rw1=[8 0.83 8];
rw2=[0.83 8];
rw3=[8 8];
rw4=[8 0.83];
```

```
rw0=[rw1 rw2 rw3 rw4];


R=diag(rw0);

Q=eye(24);


kalmanR = .008;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Area 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%all generators
if toptype == 1
    a11=[ 0 -T12 0 0 0 0 0
      1/M1 -D/M1 1/M1 0 1/M1 -1/M1 0
      0 0 -1/Td 1/Td 0 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 0 a1*K1/Tg
      0 0 0 0 -1/TE 0 a2*K1/TE
      0 0 0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0 0 0];
elseif toptype == 2
    a11=[ 0 -2*T12 0 0 0 0 0
      1/M1 -D/M1 1/M1 0 1/M1 -1/M1 0
      0 0 -1/Td 1/Td 0 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 0 a1*K1/Tg
      0 0 0 0 -1/TE 0 a2*K1/TE
      0 0 0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0 0 0];
elseif toptype == 3
    a11=[ 0 -3*T12 0 0 0 0 0
      1/M1 -D/M1 1/M1 0 1/M1 -1/M1 0
      0 0 -1/Td 1/Td 0 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 0 a1*K1/Tg
      0 0 0 0 -1/TE 0 a2*K1/TE
      0 0 0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0 0 0];
end

b11=[0 0 0;0 0 0;0 0 0;1/Tg 0 0; 0 1/TE 0;0 0 1/TH; 0 0 0];

c11=[0 1 0 0 0 0 0];

d11=zeros(1,size(b11,2));

kalmanQ1 = [5 10 5];

[kmpc1,kest1,H1,f1,Pyy1,Hy1]=createcontrol(a11,b11,c11,d11,rw1,kalmanQ1,kalmanR);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Area 2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%without gas / gov
if toptype == 3
    a22=[ 0 -3*T12 0 0 0
      1/M2 -D/M2 1/M2 -1/M2 0
      0 0 -1/TE 0 a2*K1/TE
      0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0];
else
    a22=[ 0 -2*T12 0 0 0
      1/M2 -D/M2 1/M2 -1/M2 0
      0 0 -1/TE 0 a2*K1/TE
      0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0];
```

```
end


b22=[0 0;0 0; 1/TE 0;0 1/TH; 0 0];


c22=[0 1 0 0 0];


d22=zeros(1,size(b22,2));


kalmanQ2 = [5 1];


[kmpc2,kest2,H2,f2,Pyy2,Hy2]=createcontrol(a22,b22,c22,d22,rw2,kalmanQ2,kalmanR);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Area 3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%without battery
if toptype == 3
    a33=[ 0 -3*T12 0 0 0 0
      1/M2 -D/M2 1/M2 0 -1/M2 0
      0 0 -1/Td 1/Td 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 a1*K1/Tg
      0 0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0 0];
else
    a33=[ 0 -2*T12 0 0 0 0
      1/M2 -D/M2 1/M2 0 -1/M2 0
      0 0 -1/Td 1/Td 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 a1*K1/Tg
      0 0 0 0 -1/TH a3*K1/TH
      1 -B1 0 0 0 0];
end
```

```
b33=[0 0;0 0;0 0;1/Tg 0;0 1/TH; 0 0];

c33=[0 1 0 0 0 0];

d33=zeros(1,size(b33,2));

kalmanQ3 = [5 5];

[kmpc3,kest3,H3,f3,Pyy3,Hy3]=createcontrol(a33,b33,c33,d33,rw3,kalmanQ3,kalmanR);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Area 4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%without HP
if toptype == 1
    a44=[ 0 -T12 0 0 0 0
      1/M2 -D/M2 1/M2 0 1/M2 0
      0 0 -1/Td 1/Td 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 a1*K1/Tg
      0 0 0 0 -1/TE a2*K1/TE
      1 -B1 0 0 0 0];
elseif toptype == 2
    a44=[ 0 -2*T12 0 0 0 0
      1/M2 -D/M2 1/M2 0 1/M2 0
      0 0 -1/Td 1/Td 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 a1*K1/Tg
      0 0 0 0 -1/TE a2*K1/TE
      1 -B1 0 0 0 0];
elseif toptype == 3
    a44=[ 0 -3*T12 0 0 0 0
      1/M2 -D/M2 1/M2 0 1/M2 0
      0 0 -1/Td 1/Td 0 0
      0 -1/(Tg*Rg) 0 -1/Tg 0 a1*K1/Tg
      0 0 0 0 -1/TE a2*K1/TE
      1 -B1 0 0 0 0];
end


b44=[0 0;0 0;0 0;1/Tg 0 ; 0 1/TE ; 0 0];


c44=[0 1 0 0 0 0];


d44=zeros(1,size(b44,2));


kalmanQ4 = [5 10];


[kmpc4,kest4,H4,f4,Pyy4,Hy4]=createcontrol(a44,b44,c44,d44,rw4,kalmanQ4,kalmanR);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Cent / Cross %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
a12=zeros(size(a11,1),size(a22,2));
a21=a12';
a12(1,2)=T12;
a21(1,2)=T12;
a23=zeros(size(a22,1),size(a33,2));
a32=a23';
a23(1,2)=T12;
a32(1,2)=T12;
a34=zeros(size(a33,1),size(a44,2));
a43=a34';
a34(1,2)=T12;
a43(1,2)=T12;


a13=zeros(size(a11,1),size(a33,2));
```

```
a14=zeros(size(a11,1),size(a44,2));
a24=zeros(size(a22,1),size(a44,2));
a31=zeros(size(a33,1),size(a11,2));
a41=zeros(size(a44,1),size(a11,2));
a42=zeros(size(a44,1),size(a22,2));

if toptype == 2
    a14(1,2)=T12;
    a41(1,2)=T12;
elseif toptype == 3
    a14(1,2)=T12;
    a41(1,2)=T12;
    a13(1,2)=T12;
    a31(1,2)=T12;
    a24(1,2)=T12;
    a42(1,2)=T12;
end

a=[a11 a12 a13 a14;
   a21 a22 a23 a24;
   a31 a32 a33 a34;
   a41 a42 a43 a44];

b=zeros(24,9);
b(1:7,1:3)=b11;
b(8:12,4:5)=b22;
b(13:18,6:7)=b33;
b(19:24,8:9)=b44;

c=zeros(4,24);
c(1,1:7)=c11;
```

```
c(2,8:12)=c22;
c(3,13:18)=c33;
c(4,19:24)=c44;

d=zeros(4,9);

%linear
[kmpcdl1,kestdl1,Hdl1,fdl1,Pyydl1,Hydl1]=createcontrol([a11 a12; a21 a22],
        b(1:12,1:5),c(1:2,1:12),zeros(2,5),[rw1 rw2],[kalmanQ1 kalmanQ2],
        diag([kalmanR kalmanR]));
[kmpcdl2,kestdl2,Hdl2,fdl2,Pyydl2,Hydl2]=createcontrol([a11 a12 a13;
        a21 a22 a23;a31 a32 a33],b(1:18,1:7),c(1:3,1:18),zeros(3,7),
        [rw1 rw2 rw3],[kalmanQ1 kalmanQ2 kalmanQ3],
        diag([kalmanR kalmanR kalmanR]));
[kmpcdl3,kestdl3,Hdl3,fdl3,Pyydl3,Hydl3]=createcontrol([a22 a23 a24;
        a32 a33 a34; a42 a43 a44],b(8:24,4:9),c(2:4,8:24),zeros(3,6),
        [rw2 rw3 rw4],[kalmanQ2 kalmanQ3 kalmanQ4],
        diag([kalmanR kalmanR kalmanR]));
[kmpcdl4,kestdl4,Hdl4,fdl4,Pyydl4,Hydl4]=createcontrol([a33 a34; a43 a44],
        b(13:24,6:9),c(3:4,13:24),zeros(2,4),[rw3 rw4],[kalmanQ3 kalmanQ4],
        diag([kalmanR kalmanR]));

%square
[kmpcds1,kestds1,Hds1,fds1,Pyyds1,Hyds1]=createcontrol([a11 a12 a14;
        a21 a22 a24; a41 a42 a44],b([1:12,19:24],[1:5,8:9]),
        c([1:2,4],[1:12,19:24]),zeros(3,7),[rw1 rw2 rw4],
        [kalmanQ1 kalmanQ2 kalmanQ4],diag([kalmanR kalmanR kalmanR]));
[kmpcds2,kestds2,Hds2,fds2,Pyyds2,Hyds2]=createcontrol([a11 a12 a13;
        a21 a22 a23; a31 a32 a33],b(1:18,1:7),c(1:3,1:18),zeros(3,7),
        [rw1 rw2 rw3],[kalmanQ1 kalmanQ2 kalmanQ3],
        diag([kalmanR kalmanR kalmanR]));
```

```
[kmpcds3,kestds3,Hds3,fds3,Pyyds3,Hyds3]=createcontrol([a22 a23 a24;
        a32 a33 a34; a42 a43 a44],b(8:24,4:9),c(2:4,8:24),zeros(3,6),
        [rw2 rw3 rw4],[kalmanQ2 kalmanQ3 kalmanQ4],
        diag([kalmanR kalmanR kalmanR]));
[kmpcds4,kestds4,Hds4,fds4,Pyyds4,Hyds4]=createcontrol([a11 a13 a14;
        a31 a33 a34; a41 a43 a44],b([1:7,13:24],[1:3,6:9]),
        c([1,3:4],[1:7,13:24]),zeros(3,7),[rw1 rw3 rw4],
        [kalmanQ1 kalmanQ3 kalmanQ4],diag([kalmanR kalmanR kalmanR]));

[kmpc0,kest0,H0,f0,Pyy0,Hy0]=createcontrol(a,b,c,d,rw0,
        [kalmanQ1 kalmanQ2 kalmanQ3 kalmanQ4],
        diag([kalmanR kalmanR kalmanR kalmanR]));
[A,B,C,Dd]=c2dm(a,b,c,d,.1);
```

A.1.2 Create controller

createcontrol.m

```
function [ Kmpc,kest,H,f,Pyy,Hy] = createcontrol( a,b,c,d,rw,kalmanQ,kalmanR)

%%%% ss-mpc
Np=evalin('base','Np'); %take Np from workspace
Nc=evalin('base','Nc'); %take Nc from workspace

[A,B,C,Dd]=c2dm(a,b,c,d,.1);

ma=size(A,1);
n1=size(C,2);
nb1=size(B,2);
Pxx = A;

R=rw;
```

```
for rr=1:size(rw,2):Np*nb1-size(rw,2)
    R=[R rw];
end
R=diag(R);
Q=eye(Np*n1);

for kk=ma+1:ma:ma*Np-1
    Pxx(kk:kk+ma-1,:)=Pxx(kk-ma:kk-1,:)*A;
end
v=[B; Pxx(1:size(Pxx)-n1,:)*B];
Hx=zeros(n1*Np,nb1*Np);
Hx(:,1:nb1)=v;
iv = 1;
for i=nb1+1:nb1:nb1*Np-nb1+1
    Hx(:,i:i+nb1-1)=[zeros(n1*iv,nb1);v(1:size(v,1)-n1*iv,:)];
    iv=iv+1;
end

Kmpc= ((Hx'*Hx+R)\Hx'*Q*Pxx);
Kmpc=Kmpc(1:nb1,:);
H= Hx'*Hx+R;
f= Hx'*Q*Pxx;
Pyy= C.*Pxx;
Hy= C.*Hx;

%%%% Kalman filter
[kest,kL,kP]=kalman(ss(a,b,c,d),diag(kalmanQ),kalmanR,zeros(size(b,2),
        size(kalmanR,1)));
end
```

A.1.3 Controller

control.m

```
function control(block)
    setup(block);


function setup(block)
    block.NumInputPorts  = 5;
    block.NumOutputPorts = 1;

    block.InputPort(1).Dimensions =[24 1];
    block.InputPort(2).Dimensions =[24 1];
    block.InputPort(3).Dimensions =[24 1];
    block.InputPort(4).Dimensions =[24 1];
    block.InputPort(5).Dimensions =[1 1];

    block.OutputPort(1).Dimensions =[9 1];

    block.InputPort(1).DirectFeedthrough=false;
    block.InputPort(2).DirectFeedthrough=false;
    block.InputPort(3).DirectFeedthrough=false;
    block.InputPort(4).DirectFeedthrough=false;
    block.InputPort(5).DirectFeedthrough=false;

    block.InputPort(1).SamplingMode = 'Sample';
    block.InputPort(2).SamplingMode = 'Sample';
    block.InputPort(3).SamplingMode = 'Sample';
    block.InputPort(4).SamplingMode = 'Sample';
    block.InputPort(5).SamplingMode = 'Sample';

    block.OutputPort(1).SamplingMode = 'Sample';
    % Override input port properties
    block.InputPort(1).DatatypeID  = 0;  % double
    block.InputPort(1).Complexity  = 'Real';
    block.InputPort(2).DatatypeID  = 0;  % double
    block.InputPort(2).Complexity  = 'Real';
    block.InputPort(3).DatatypeID  = 0;  % double
    block.InputPort(3).Complexity  = 'Real';
    block.InputPort(4).DatatypeID  = 0;  % double
    block.InputPort(4).Complexity  = 'Real';
    block.InputPort(5).DatatypeID  = 0;  % double
    block.InputPort(5).Complexity  = 'Real';

    % Override output port properties
    block.OutputPort(1).DatatypeID  = 0; % double
    block.OutputPort(1).Complexity  = 'Real';

    block.NumDialogPrms = 4;
    block.SampleTimes = [-1 0];

    block.RegBlockMethod('Outputs', @Outputs);
    block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
    block.RegBlockMethod('InitializeConditions', @InitializeConditions);
    block.RegBlockMethod('Start', @Start);

function DoPostPropSetup(block)
    block.NumDworks=2;

    names = {'x','w',};
    for n=1:2
        block.Dwork(n).Complexity='Real';
        block.Dwork(n).DatatypeID=0;
        block.Dwork(n).Dimensions=4;
        block.Dwork(n).Name=names{n};
        block.Dwork(n).UsedAsDiscState=true;
    end

function InitializeConditions(block)
    block.Dwork(1).Data=[1;1;1;1];

function Start(block)
    global A B C toptype

    A=block.DialogPrm(1).Data;
    B=block.DialogPrm(2).Data;
    C=block.DialogPrm(3).Data;
    toptype=block.DialogPrm(4).Data;

function Outputs(block)
    global A B C toptype

    x1=block.InputPort(1).Data;
    x2=block.InputPort(2).Data;
    x3=block.InputPort(3).Data;
    x4=block.InputPort(4).Data;
    p=block.InputPort(5).Data;

    if p==1 %Centralized
        %take values from workspace
        kmpc0=evalin('base','kmpc0');

        u=-kmpc0*x1;
        yabs=abs(C*A*x1+C*B*u);
        if yabs>0.2
            H=evalin('base','H0'); %take H from workspace
            f=evalin('base','f0'); %take f from workspace
            Pyy=evalin('base','Pyy0'); %take Pyy from workspace
            Hy=evalin('base','Hy0'); %take Hy from workspace
            u=quadprog(H,f*x1,[-Hy Hy]',[0.2+Pyy*x1 0.2-Pyy*x1]);
        end
    end

    if p==2 %Decentralized
        %take values from workspace
        kmpc1=evalin('base','kmpc1');
        kmpc2=evalin('base','kmpc2');
        kmpc3=evalin('base','kmpc3');
        kmpc4=evalin('base','kmpc4');

        u(1:3)=-kmpc1*x1(1:7);
        u(4:5)=-kmpc2*x2(8:12);
        u(6:7)=-kmpc3*x3(13:18);
        u(8:9)=-kmpc4*x4(19:24);
        u=u';

        yabs=abs(C*A*x1+C*B*u);
        if yabs(1)>0.2
            H=evalin('base','H1'); %take H from workspace
            f=evalin('base','f1'); %take f from workspace
            Pyy=evalin('base','Pyy1'); %take Pyy from workspace
            Hy=evalin('base','Hy1'); %take Hy from workspace
            u(1:3)=quadprog(H,f*x1,[-Hy Hy]',[0.2+Pyy*x1 0.2-Pyy*x1]);
        end
        if yabs(2)>0.2
            H=evalin('base','H2'); %take H from workspace
            f=evalin('base','f2'); %take f from workspace
            Pyy=evalin('base','Pyy2'); %take Pyy from workspace
            Hy=evalin('base','Hy2'); %take Hy from workspace
            u(4:5)=quadprog(H,f*x2,[-Hy Hy]',[0.2+Pyy*x2 0.2-Pyy*x2]);
        end
        if yabs(3)>0.2
            H=evalin('base','H3'); %take H from workspace
            f=evalin('base','f3'); %take f from workspace
            Pyy=evalin('base','Pyy3'); %take Pyy from workspace
            Hy=evalin('base','Hy3'); %take Hy from workspace
            u(6:7)=quadprog(H,f*x3,[-Hy Hy]',[0.2+Pyy*x3 0.2-Pyy*x3]);
        end
        if yabs(4)>0.2
            H=evalin('base','H4'); %take H from workspace
            f=evalin('base','f4'); %take f from workspace
            Pyy=evalin('base','Pyy4'); %take Pyy from workspace
            Hy=evalin('base','Hy4'); %take Hy from workspace
            u(8:9)=quadprog(H,f*x4,[-Hy Hy]',[0.2+Pyy*x4 0.2-Pyy*x4]);
        end
    end

    if p==3 %Distributed
        if toptype == 1 %linear
            %take values from workspace
            kmpcdl1=evalin('base','kmpcdl1');
            kmpcdl2=evalin('base','kmpcdl2');
            kmpcdl3=evalin('base','kmpcdl3');
            kmpcdl4=evalin('base','kmpcdl4');
```

```
u1=-kmpcdl1*x1(1:12);
u2=-kmpcdl2*x2(1:18);
u3=-kmpcdl3*x3(8:24);
u4=-kmpcdl4*x4(13:24);

u(1:3)=u1(1:3);
u(4:5)=u2(4:5);
u(6:7)=u3(3:4);
u(8:9)=u4(3:4);
u=u';

yabs=abs(C*A*x1+C*B*u);
if yabs(1)>0.2
    H=evalin('base','H1'); %take H from workspace
    f=evalin('base','f1'); %take f from workspace
    Pyy=evalin('base','Pyy1'); %take Pyy from workspace
    Hy=evalin('base','Hy1'); %take Hy from workspace
    u(1:3)=quadprog(H,f*x1,[-Hy Hy]',[0.2+Pyy*x1 0.2-Pyy*x1]);
end
if yabs(2)>0.2
    H=evalin('base','H2'); %take H from workspace
    f=evalin('base','f2'); %take f from workspace
    Pyy=evalin('base','Pyy2'); %take Pyy from workspace
    Hy=evalin('base','Hy2'); %take Hy from workspace
    u(4:5)=quadprog(H,f*x2,[-Hy Hy]',[0.2+Pyy*x2 0.2-Pyy*x2]);
end
if yabs(3)>0.2
    H=evalin('base','H3'); %take H from workspace
    f=evalin('base','f3'); %take f from workspace
    Pyy=evalin('base','Pyy3'); %take Pyy from workspace
    Hy=evalin('base','Hy3'); %take Hy from workspace
```

```
            u(6:7)=quadprog(H,f*x3,[-Hy Hy]',[0.2+Pyy*x3 0.2-Pyy*x3]);
        end
        if yabs(4)>0.2
            H=evalin('base','H4'); %take H from workspace
            f=evalin('base','f4'); %take f from workspace
            Pyy=evalin('base','Pyy4'); %take Pyy from workspace
            Hy=evalin('base','Hy4'); %take Hy from workspace
            u(8:9)=quadprog(H,f*x4,[-Hy Hy]',[0.2+Pyy*x4 0.2-Pyy*x4]);
        end
    elseif toptype==2 %square
        %take values from workspace
        kmpcds1=evalin('base','kmpcds1');
        kmpcds2=evalin('base','kmpcds2');
        kmpcds3=evalin('base','kmpcds3');
        kmpcds4=evalin('base','kmpcds4');

        u1=-kmpcds1*x1([1:12,19:24]);
        u2=-kmpcds2*x2(1:18);
        u3=-kmpcds3*x3(8:24);
        u4=-kmpcds4*x4([1:7,13:24]);

        u(1:3)=u1(1:3);
        u(4:5)=u2(4:5);
        u(6:7)=u3(3:4);
        u(8:9)=u4(6:7);
        u=u';

        yabs=abs(C*A*x1+C*B*u);
        if yabs(1)>0.2
            H=evalin('base','H1'); %take H from workspace
            f=evalin('base','f1'); %take f from workspace
            Pyy=evalin('base','Pyy1'); %take Pyy from workspace
            Hy=evalin('base','Hy1'); %take Hy from workspace
            u(1:3)=quadprog(H,f*x1,[-Hy Hy]',[0.2+Pyy*x1 0.2-Pyy*x1]);
        end
        if yabs(2)>0.2
            H=evalin('base','H2'); %take H from workspace
            f=evalin('base','f2'); %take f from workspace
            Pyy=evalin('base','Pyy2'); %take Pyy from workspace
            Hy=evalin('base','Hy2'); %take Hy from workspace
            u(4:5)=quadprog(H,f*x2,[-Hy Hy]',[0.2+Pyy*x2 0.2-Pyy*x2]);
        end
        if yabs(3)>0.2
            H=evalin('base','H3'); %take H from workspace
            f=evalin('base','f3'); %take f from workspace
            Pyy=evalin('base','Pyy3'); %take Pyy from workspace
            Hy=evalin('base','Hy3'); %take Hy from workspace
            u(6:7)=quadprog(H,f*x3,[-Hy Hy]',[0.2+Pyy*x3 0.2-Pyy*x3]);
        end
        if yabs(4)>0.2
            H=evalin('base','H4'); %take H from workspace
            f=evalin('base','f4'); %take f from workspace
            Pyy=evalin('base','Pyy4'); %take Pyy from workspace
            Hy=evalin('base','Hy4'); %take Hy from workspace
            u(8:9)=quadprog(H,f*x4,[-Hy Hy]',[0.2+Pyy*x4 0.2-Pyy*x4]);
        end
    elseif toptype==3 %full
        %take values from workspace
        kmpc0=evalin('base','kmpc0');

        u1=-kmpc0*x1;
        u2=-kmpc0*x2;
```

```
u3=-kmpc0*x3;
u4=-kmpc0*x4;

u(1:3)=u1(1:3);
u(4:5)=u2(4:5);
u(6:7)=u3(3:4);
u(8:9)=u4(6:7);
u=u';

yabs=abs(C*A*x1+C*B*u);
if yabs(1)>0.2
    H=evalin('base','H1'); %take H from workspace
    f=evalin('base','f1'); %take f from workspace
    Pyy=evalin('base','Pyy1'); %take Pyy from workspace
    Hy=evalin('base','Hy1'); %take Hy from workspace
    u(1:3)=quadprog(H,f*x1,[-Hy Hy]',[0.2+Pyy*x1 0.2-Pyy*x1]);
end
if yabs(2)>0.2
    H=evalin('base','H2'); %take H from workspace
    f=evalin('base','f2'); %take f from workspace
    Pyy=evalin('base','Pyy2'); %take Pyy from workspace
    Hy=evalin('base','Hy2'); %take Hy from workspace
    u(4:5)=quadprog(H,f*x2,[-Hy Hy]',[0.2+Pyy*x2 0.2-Pyy*x2]);
end
if yabs(3)>0.2
    H=evalin('base','H3'); %take H from workspace
    f=evalin('base','f3'); %take f from workspace
    Pyy=evalin('base','Pyy3'); %take Pyy from workspace
    Hy=evalin('base','Hy3'); %take Hy from workspace
    u(6:7)=quadprog(H,f*x3,[-Hy Hy]',[0.2+Pyy*x3 0.2-Pyy*x3]);
end
```

```
        if yabs(4)>0.2
            H=evalin('base','H4'); %take H from workspace
            f=evalin('base','f4'); %take f from workspace
            Pyy=evalin('base','Pyy4'); %take Pyy from workspace
            Hy=evalin('base','Hy4'); %take Hy from workspace
            u(8:9)=quadprog(H,f*x4,[-Hy Hy]',[0.2+Pyy*x4 0.2-Pyy*x4]);
        end
    end
end
block.OutputPort(1).Data =u;
```

A.2 mdl-files

These figures shows the model created for the four subsystem network.

The top level depicted in Fig. A.1 includes: the sub level for the subsystems with plants, Fig. A.2, sub level for the controller, Fig. A.4, sub level for the cost function, Fig. A.5, sub level for the topology, Fig. A.6, sub level for the states, Fig. A.7, sub level for the controller input, Fig. A.8, and the controls for changing control method, topology and controller input signal. Also included in the top level are the scopes for the most relevant output signals, the frequency and the cost.

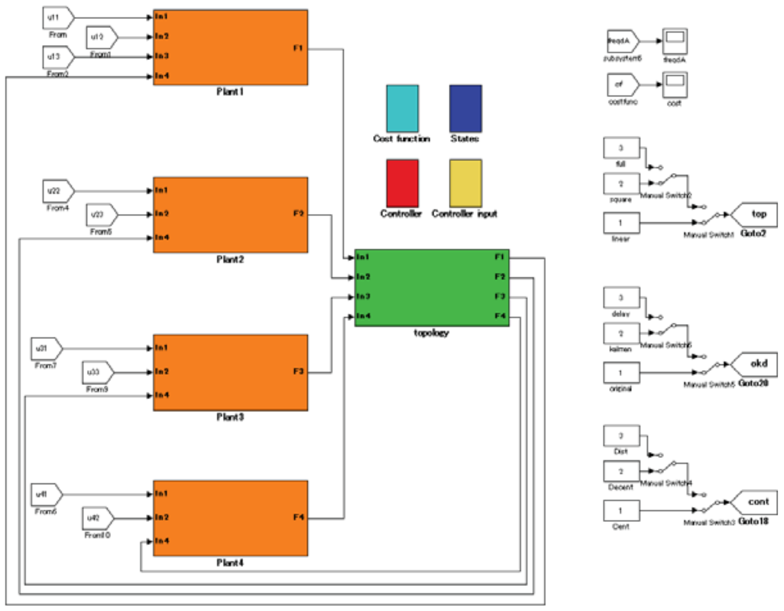


Fig. A.1 : Network

Fig. A.2 shows the contents of the first subsystem, containing the transfer functions for the gas generator and governor, thermal system and battery storage system. It also includes a sub level for the wind turbine, Fig A.3.

Fig. A.3 shows the contents of the wind turbine sub level, which basic function is to generate a fluctuating output to simulate the unstable frequency output of a wind turbine.

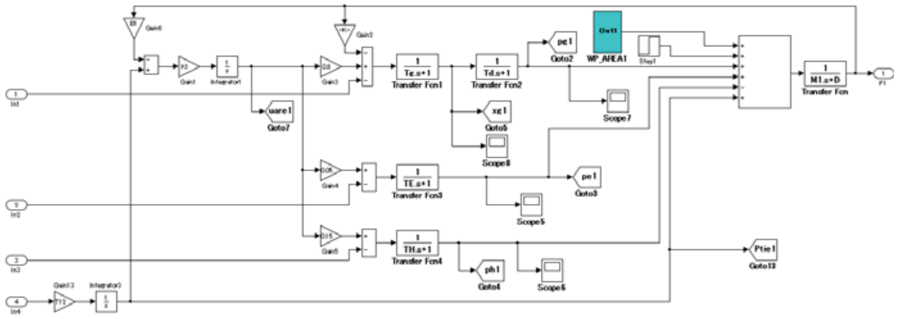


Fig. A.2 : Plant1

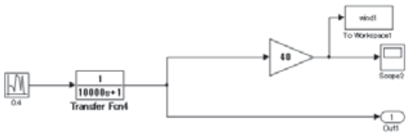


Fig. A.3 : WP_AREA1

The controller sub level depicted in Fig. A.4 gets the measurement signals and current controller type and sends them to the controller file, which calculates the new control signals, and then sends the received new control signals to the appropriate generator and to the cost calculator, Fig A.5.

The cost calculator in Fig. A.5 takes the states and the control signals and calculates the cost according to the defined cost function.

The topology sub level in Fig. A.6 takes care of the connections between the subsystems, and changes according to what topology is chosen at the top level switch.

The states sub level depicted in Fig. A.7 collects all the different states into a state vector. It also includes scopes for each group of states for inspection purposes.

The sub level depicted in Fig. A.8 takes care of what signal the controller gets as input, depending on the switch choice at the top level. Either it gets the original true values, Kalman estimated values, or delayed true values from the delay sub level, Fig. A.9.

The delay sub level in Fig. A.9 takes the original true values and delay them for ten time samples.

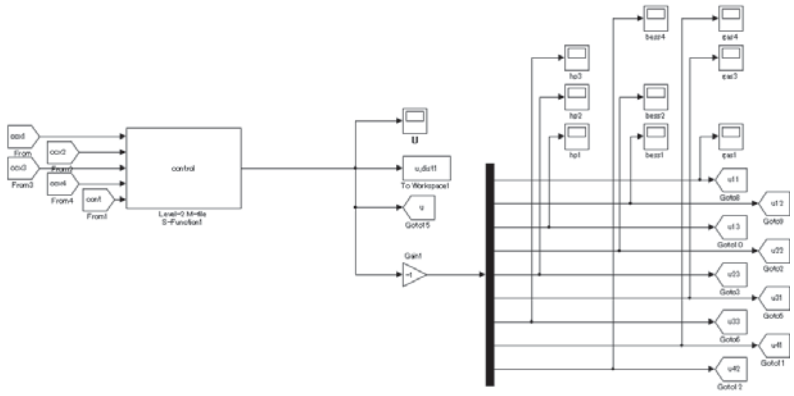


Fig. A.4 : Controller

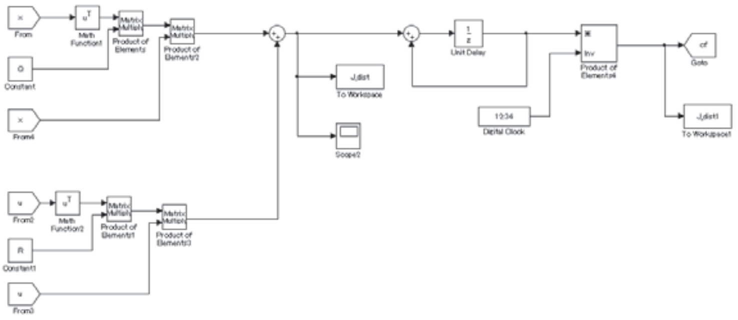


Fig. A.5 : Cost function

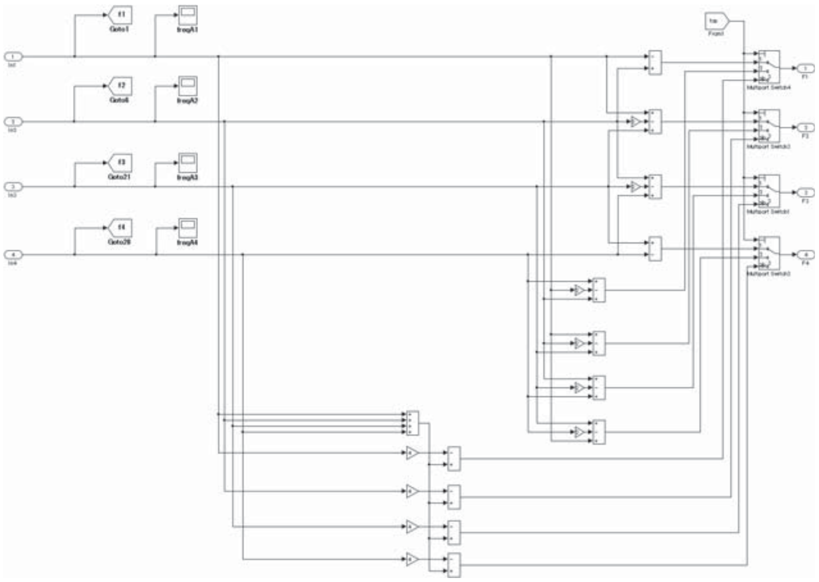


Fig. A.6 : Topology

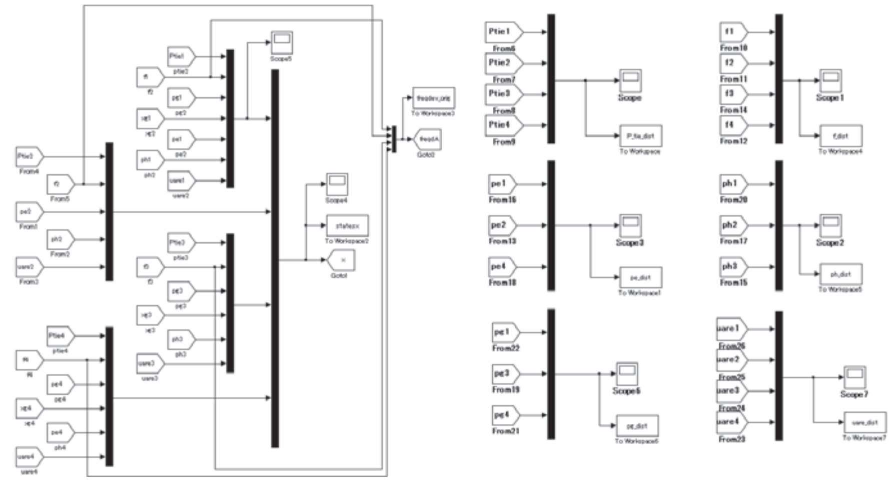


Fig. A.7 : States

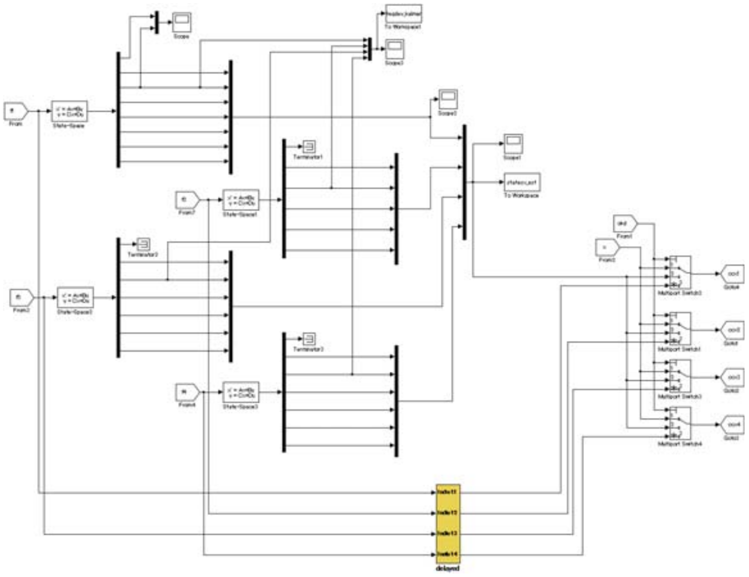


Fig. A.8 : Controller input

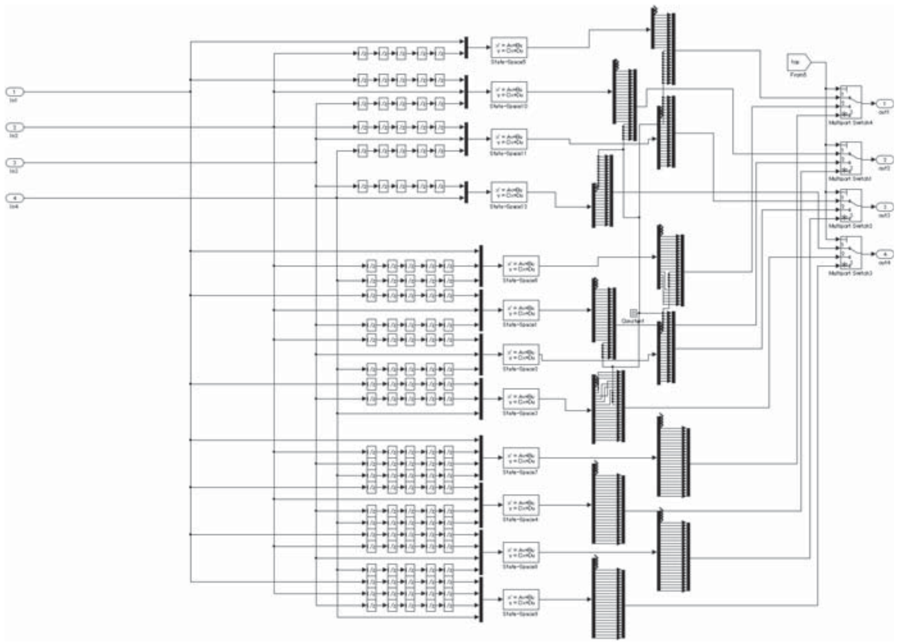


Fig. A.9 : Delay

Appendix B

Experimental setup

A power system using synchronous generators could be used for experimental verification. This chapter describes the system architecture and specification of the experiment devices, discussing the environments, and the result in the case of regular PI control.

B.1 System architecture

Fig. B.1 - B.2 shows the appearance of the current system setup.

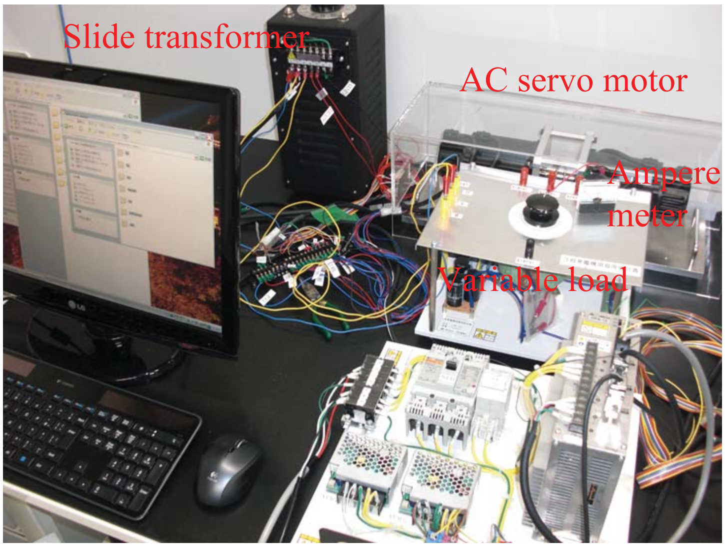


Fig. B.1 : Appearance of the system

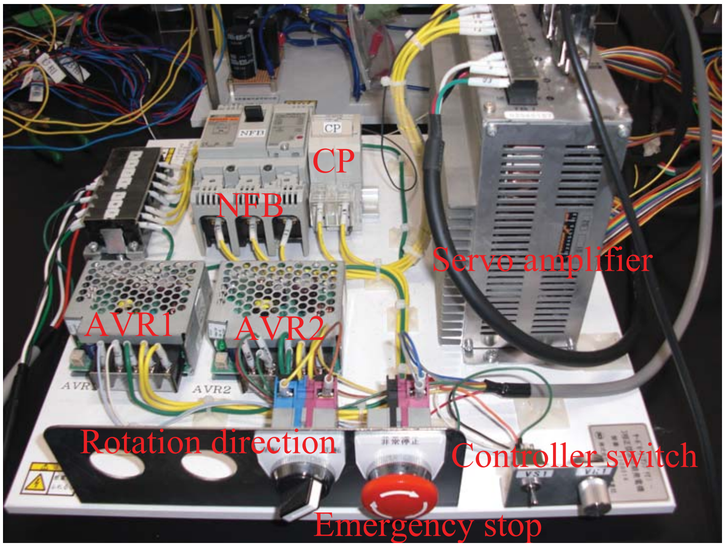


Fig. B.2 : Appearance of the control board

Fig. B.3 shows the system architecture, how the generator is connected to the variable load resistance. The rotation speed of the generator is measure from the encoder and three-phase AC voltage through dSPACE, that will send the input voltage to the servo amplifier to control the rotation speed.

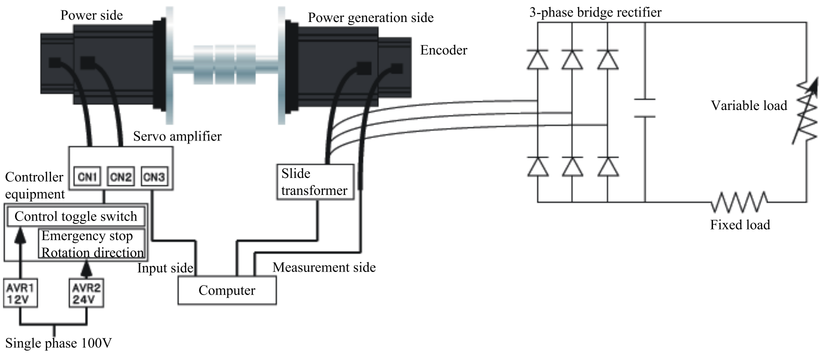


Fig. B.3 : Schematic view of the system

The detail of each system component is stated below.

1. Switching power supply AVR1

   Model number : PBA10F-12-N

   Specification : Peak power output 10.8W, DC output 12V/0.9A

2. Switching power supply AVR2

   Model number : PBA10F-24-N

   Specification : Peak power output 12W, DC output 24V/0.5A

3. Servo amplifier

   Model number : GPA-12

   Specification : Analog command, Rated current 2.4A/rms, Peak current 8.5A/rms

   The servo amplifier act as a controller for the servo motor, where the connections are as follows.

$$\text{TB1}\begin{cases}\text{RSTE}\rightarrow\text{NFB}\\ \text{rt}\rightarrow\text{NFB}\\ \text{UVWE}\rightarrow\text{Motor}\end{cases}\qquad\begin{array}{l}\text{CN1}\rightarrow\text{Controller}\\ \text{CN2}\rightarrow\text{Encoder}\\ \text{CN3}\rightarrow\text{PC}\end{array}$$

4. AC servo motor

   Model number : LNEII040C

   Specification : Rated output 400W, Rated rotation speed 3000rpm, Rated current 2.4A

5. Transformer

   Model number : RTC-5

   Specification : Output current 5A, Output capacity 1.7KVA

6. Load

   Rectifier diode : 600V, 60A

   Smoothing capacitor : 200V, 820μF

   Fixed load resistor : 10Ω, 50W

   Variable load resistor : 0-200Ω, 300W

Parameters about the generator is stated below.

Table B.1 : Generator parameters

Parameter	Value	Unit	Value	Unit
Rated capacity	400	W		
Rated torque	1.27	N·m	13	kgf · cm
Peak torque	3.8	N·m	39	kgf · cm
Rated rotation speed	3000	rpm	50	/s
Peak rotation speed	3500	rpm	175/3	/s
Rated current	2.4	A		
Peak current	7.2	A		
Rated voltage	168	V		
Torque constant	0.58	N·m/A	5.96	kgf · cm/A
Rotor inertia	1.16×10^{-4}	kg·m^2	1.18	gf · cms^2
Power rate	14	kW/s		
Mechanical time constant	7.7	ms		
Electrical time constant	1.6	ms		
General weight	3.0	kg·f	29.4	N
Motor pole number	8	pole		
Reverse voltage constant	20.4	V/krpm	1.224	V·s
Armature inductance	4.0	mH		
Armature resistance	2530	mΩ		
Phase correction angle	0	deg		
Encoder density	2000	pulse		
Encoder pole number	8	pole		

B.2 Control by dSPACE

The frequency is calculated from the revolution speed of the encoder by the following equations.

Table B.2 : Definition of symbols

Symbol	Meaning	Value	Unit
p	Encoder pole number	8	
n_e	Encoder resolution	2000	[pulse/Rev]
n	Motor rotation speed		[rpm]
f	Frequency		[Hz]
V	Input voltage		[V]
k_v	Velocity input coefficient	3000	[rpm]

From the encoder resolution, the angle per 1[pulse] is

$$\frac{360}{n_e} = 0.18[\text{deg/pulse}] \tag{B.1}$$

$$\frac{2\pi}{n_e} = \frac{\pi}{1000}[\text{rad/pulse}]. \tag{B.2}$$

From the angle rate of the encoder, the motor rotation speed n[rpm] is

$$n = \frac{d\theta}{dt} \times \frac{1}{2\pi} \times 60. \tag{B.3}$$

The frequency f[Hz] is

$$f = \frac{np}{120}. \tag{B.4}$$

The velocity input coefficient k_v is set as a motor rotation speed when a velocity input of 10[V] is added. The motor rotation speed n[rpm] for the input voltage is

$$n = V \times \frac{k_v}{10}. \tag{B.5}$$

From equations (B.4) and (B.5), the relation between the frequency and the input voltage is as follows.

$$V = f \times \frac{120}{8} \times \frac{10}{3000} \tag{B.6}$$

B.3 PI control

The Simulink model for the control is shown in Fig. B.4.

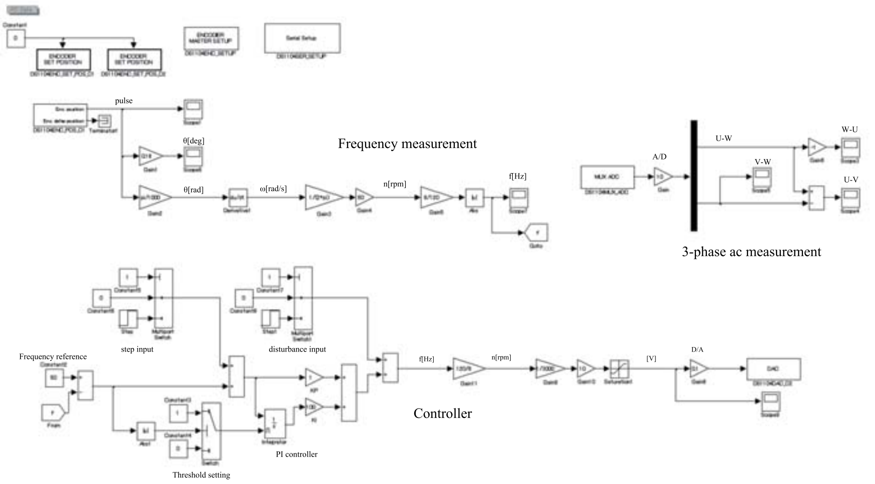
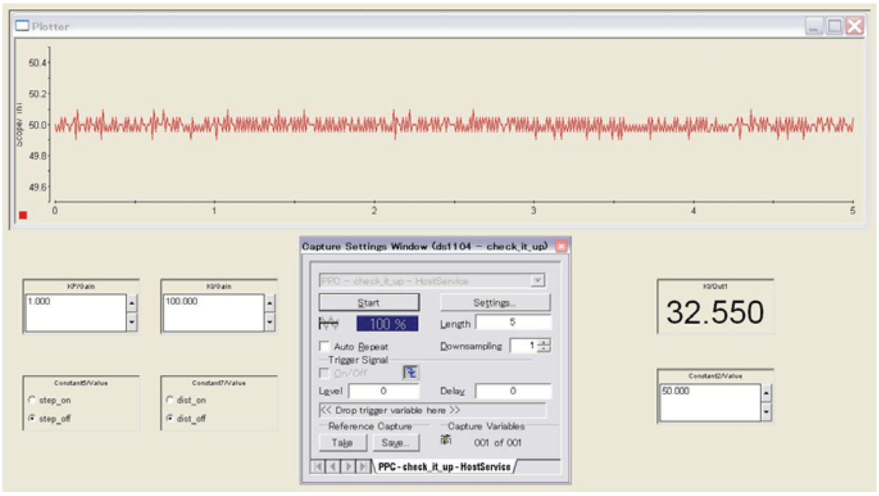


Fig. B.4 : Simulation block

In Fig. B.5 the display of Control Desk in the experiment is shown. The upper part shows the frequency deviation, and in the lower part changes to the proportional gain KP and integral gain KI, step input and disturbance input can be done.


Plotter
Scope/ in1
50.4
50.2
50.0
49.8
49.6
0
1
2
3
4
5
Capture Settings Window (ds1104 – check_it_up)
PPC – check_it_up – HostService
Start
Settings...
100 %
Trigger Signal
Drop trigger variable here
Reference Capture
Take
Save...
Capture Variables
PPC - check_it_up - HostService

KP/Gain: 1.000

KI/Gain: 100.000

KI/Out1: 32.550

Constant5/Value: step_on [✓ step_off]

Constant7/Value: dist_on [✓ dist_off]

Constant2/Value: 50.000

Length: 5

Downsampling: 1

Auto Repeat: ☐

On/Off: ☐

Level: 0

Delay: 0

001 of 001

Fig. B.5 : Control Desk

Table B.3 shows the experimental parameters. The load fluctuation is the manual change of the value on the variable load resistor.

Table B.3 : Experimental parameters

Parameters	Value	Unit
Sampling time	0.01	s
Standard frequency	50	Hz
Threshold value	100	-
Upper and lower limit of saturation	± 10	V
Velocity loop compensation gain	1	-
Proportional gain KP	1	-
Integration gain KI	100	-
Size of step input	3	Hz
Size of disturbance input	10	Hz

The result is as follows. Fig. B.6 shows the frequency deviation when the variable load resistor is changed slowly, and Fig. B.7 shows the frequency deviation when the variable load resistor is changed rapidly.

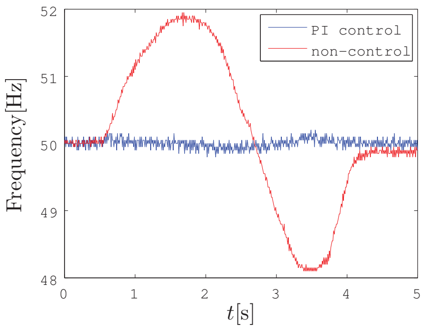


Fig. B.6 : Frequency deviation (Slow load fluctuation)

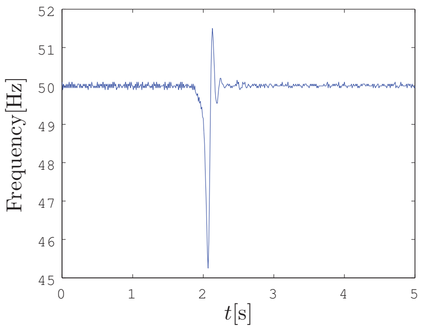


Fig. B.7 : Frequency deviation (Rapid load fluctuation)

In Fig. B.6, the frequency deviation is suppressed by PI control. On the other hand in Fig. B.7, the frequency deviation is bigger than ± 0.2[Hz] when a rapid load fluctuation is added.

Fig. B.8 shows the voltage change when no control input is added.

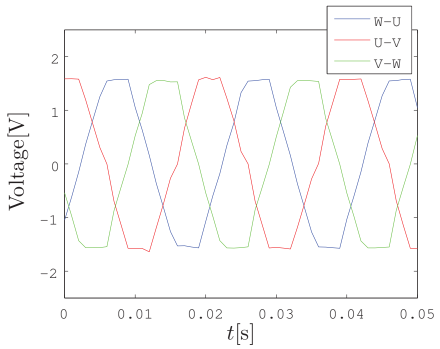


Fig. B.8 : Voltage

Fig. B.9 - B.10 shows the step response and disturbance response without any control activated.

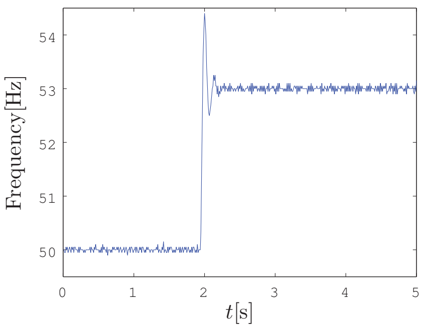


Fig. B.9 : Step response

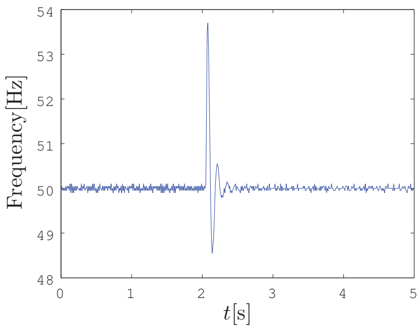


Fig. B.10 : Disturbance response

In these figures, the step input is added and the disturbance input changes the frequency, both exceeding ±0.2[Hz] safety limit.

In the future, the next step is to extend the system with another motor-generator setup and a board to represent the network, as shown in Fig. B.11. This will move it another step toward representing a real system, and from this more complex control methods can be implemented.

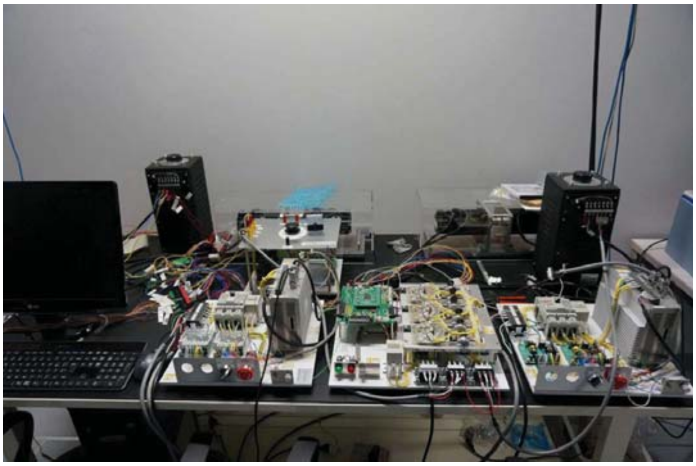


Fig. B.11 : Appearance of the bigger system

Acknowledgment

As a double-degree exchange student I would like to thank Keio University and Lund University for giving me this great opportunity that I have been given.

I would also like to thank my professor, Toru Namerikawa, for accepting me into his laboratory, and also all the members of the laboratory, for helping me both with my research and the daily life, especially Ohkawa, Ishikawa, Tokumoto, Tachibana, Haraikawa, Mukai, Fujita, Miyano, Hosoda, Suehiro, Maeda, Okubo, Kato, Ohtani, Kosugi and Ishimura.

Thanks also goes out to my examiner at Lund University, Anders Rantzer.

And I would also like to thank the various control theory teachers I have had during these years in university.

I would also like to thank my family.

Publications

[1] Christian Fogelberg and Toru Namerikawa. Distributed Model Predictive Control of Load Frequency of Power Network. *SICE Annual Conference 2013.*

Bibliography

[1] J. J. Jamian, M. W. Mustafa, H. Mokhlis, M. A. Baharudin. Smart Grid Communication Concept for Frequency Control in Distributed System. *Proceedings of The 5th International Power Engineering anp Optimization Conference*, pp. 238-242, 2011.

[2] Changhong Zhao, Ufuk Topcu, Stephen H. Low. Frequency-Based Load Control in Power Systems. *Proceedings of 2012 American Control Conference*, pp. 4423-4430, 2012.

[3] Jean Kumagai. The Smartest, Greenest Grid. *IEEE Spectrum*, pp. 38-43, May 2013.

[4] Peter Fairley. Germany Takes the Lead in HVDC. *IEEE Spectrum*, pp. 32-37, May 2013.

[5] Gianluca Antonelli. Interconnected Dynamic Systems. *IEEE Control Systems Magazine*, pp. 76-88, February, 2013.

[6] Peter F. Odgaard and Kathryn E. Johnson. Wind turbine fault detection and fault tolerant control - a second challenge. kk-electronic, 2013 [viewed 13 June 2013]. Available from: http://www.kk-electronic.com/Files/Billeder/kk-electronic 2011/Turbine Control/FDI/FDbenchmark.pdf

[7] Alberto Bemporad, Manfred Morari, N. Lawrence Ricker. *Model Predictive Control Toolbox: Getting Started Guide*. The MathWorks, Inc., 2012.

[8] Alberto Bemporad, Manfred Morari, N. Lawrence Ricker. *Model Predictive Control Toolbox: User's Guide*. The MathWorks, Inc., 2012.

[9] Manfred Morari, N. Lawrence Ricker. *Model Predictive Control Toolbox*. The MathWorks, Inc., 1998.

[10] C. Schmid, L.T. Biegler. Quadratic programming methods for reduced hessian SQP. *Computers & Chemical Engineering*, pp. 817-832, Vol. 18, Number 9, 1994.

[11] Javad Lavei, Anders Rantzer, Stephen Low. Power flow optimization using positive quadratic programming. *Proceeding of 18th IFAC World Congress*, pp. 10481-10486, 2011.

[12] Karl Mårtensson and Anders Rantzer. Gradient methods for iterative distributed control synthesis. *Proceeding of 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pp. 549-554, 2009.

[13] Håkan Hjalmarsson. Efficient Tuning of Linear Multivariable Controllers Using Iterative Feedback Tuning. *International Journal of Adaptive Control and Signal Processing*, pp. 553-572, 1999.

[14] Taichiro Kato and Toru Namerikawa. Distributed Control for Load Frequency of Power Networks based on Iterative Gradient Methods. *Proceeding of SICE Annual Conference*, pp. 1384-1389, 2011.

[15] Toru Namerikawa and Taichiro Kato. Distributed Load Frequency Control of Electrical Power Networks via Iterative Gradient Methods. *Proceeding of 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 7723-7726, 2011.

[16] James B. Rawlings and Fernando V. Lima. *State Estimation of Linear and Nonlinear Dynamic Systems.* RWTH Aachen, 2008 [viewed 5 May 2013]. Available from: http://jbrwww.che.wisc.edu/presentations/stability.pdf

[17] Eric L. Haseltine and James B. Rawlings. *Using Moving Horizon Estimation to Overcome Extended Kalman Filtering Failure.* University of Wisconsin, 2003 [viewed 12 May 2013]. Available from: http://jbrwww.che.wisc.edu/presentations/haseltine.pdf

[18] Tobias Raff and Frank Allgöver. An Observer that Converges in Finite Time Due to Measurement-based State Updates. *Proceedings of the 17th IFAC World Congress*, pp. 2693-2695, 2008.

[19] J. M. Maestre, P. Giselsson and A. Rantzer. Distributed Receding Horizon Kalman Filter. *Proceedings of 49th IEEE Conference on Decision and Control*, pp. 5068-5073, 2010.

[20] Matthieu Fruchard, Guillaume Allibert, Estelle Courtial. Choice of the control horizon in an NMPC strategy for the full-state control on nonholonomic systems. *Proceedings of 2012 American Control Conference*, pp. 4149-4154, 2012.

[21] Xianzhong Chen, Mohsen Heidarinejad, Jinfeng Liu and Panagiotis D. Christofides. Composite Fast-Slow MPC DEsign for Nonlinear Singularly Perturbed Systems: Stability Analysis. *Proceedings of 2012 American Control Conference*, pp. 4136-4141, 2012.

[22] Chi-Ying Lin and Yen-Chung Liu. Precision Tracking Control and Constraint Handling of Mechatronic Servo System Using Model Predictive Control. *IEEE/ASME Transactions on Mechatronics*, pp. 593-605, vol. 17, no. 4, August 2012.

[23] M. S. Rana, H. R. Pota and I. R. Petersen. Improvement of the Tracking Accuracy of an AFM Using MPC. *Proceedings of 8th IEEE Conference on Industrial Electronics and Applications*, pp. 1681-1686, 2013.

[24] Jun Yan, Robert R. Bitmead. Model Predictive Control and State Estimation: A Network Example. *Proceedings of 15th Triennial World Congress*, 2002.

[25] Giulio Betti, Marcello Farina and Riccardo Scattolini. An MPC algorithm for offset-free tracking of constant reference signals. *Proceedings of 51st IEEE Conference on Decision and Control*, pp. 5182-5187, 2012.

[26] Stefano Riverso, Marcello Farina and Giancarlo Ferrari-Trecate. Plug-and-Play Decentralized Model Predictive Control. *Proceedings of 51st IEEE Conference on Decision and Control*, pp. 4193-4198, 2012.

[27] Maria Prandini, Simone Garatti, and John Lygeros. A Randomized Approach to Stochastic Model Predictive Control. *Proceedings of 51st IEEE Conference on Decision and Control*, pp. 7315-7320, 2012.

[28] Pontus Giselsson and Anders Rantzer. Distributed Model Predictive Control with Sub-optimality and Stability Guarantees. *Proceeding of 49th IEEE Conference on Decision and Control*, pp. 7272-7277, 2010.

[29] Eduardo F. Camacho and Carlos Bordons. *Model Predictive Control*. Springer, 1999.

[30] Liuping Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. Springer, 2009.

[31] J.A. Rossiter. *Model-Based Predictive Control*. CRC PRESS, 2005.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
December 2013

Document Number
ISRN LUTFD2/TFRT--5931--SE

Author(s)
Chirstian Fogelberg

Supervisor
Toru Namerikawa, School of Integrated Design Engineering Graduate School of Science and Technology Keio University
Anders Rantzer, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Distributed Model Predictive Control of Load Frequency for Power Networks

Abstract
In recent years, there has been an increase of interest in smart grid concept, to adapt the power grid to improve the reliability, efficiency and economics of the electricity production and distribution. One of the generator side problem in this is to meet the power requirement while not wasting unnecessary power, thus keeping the cost down, which must be done while the frequency is kept in a suitable range that will not damage any equipment connected to the power grid. It would theoretically be most logical to have a centralized controller that gathers the full networks data, calculates the control signals and adjusts the generators.
However in practice this is not practical, mostly due to distance. The transmission of sensor data to the controller and the transmission of control signals to the generators would have to travel far, thus taking up to much time before the generators could act.

This paper presents a distributed model predictive control based method to control the frequency of the power network. First, an augmented matrix model predictive controller is introduced and implemented on a two homogeneous subsystems network. Later the control method is changed to a state space model predictive controller and is then utilized on a four heterogeneous subsystems network. This controller implementation also includes state observers by Kalman filtering, constraints handler utilizing quadratic programming, and different connection topology setups to observe how the connectivity affects the outcome of the system.

The effectiveness of the proposed distributed control method was compared against the corresponding centralized and decentralized controller implementation results. It is also compared to other control algorithms, specifically, an iterative gradient method, and a model predictive controller generated by the MATLAB MPC Toolbox. The results show that the usage of a distributed setup improves the outcome compared to the decentralized case, whilst keeping a more convenient setup than the centralized case. It is also shown that the level of connectivity for a chosen network topology matters for the outcome of the system, the results are improved when more connections exists.
system, the results are improved when more connections exists.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-96

Recipient's notes

Security classification

http://www.control.lth.se/publications/