Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Andreas Faanes

Multi-criteria Controller Design for Uncertain Isolated Power Systems with Renewable Power Generation

Master's thesis in Cybernetics and Robotics
Supervisor: Damiano Varagnolo
Co-supervisor: Spyridon Chapaloglou
June 2021

NTNU
Norwegian University of Science and Technology

Andreas Faanes

Multi-criteria Controller Design for Uncertain Isolated Power Systems with Renewable Power Generation

Master's thesis in Cybernetics and Robotics
Supervisor: Damiano Varagnolo
Co-supervisor: Spyridon Chapaloglou
June 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

To accommodate the visions of high penetration renewable energy power generation at offshore oil & gas platforms, control algorithms that effectively can deal with uncertainty and manage frequency control need to be developed. Due to the low system inertia and stochastic nature of the power generation, the uncertainty and vulnerability of such offshore isolated power systems with no connection to the main grid are significantly higher than for the main grid. In this thesis, scenario stochastic model predictive controllers with different control action parametrizations, are developed and compared to each other, the traditional deterministic model predictive controller, and the robust mixed synthesis $H\infty$ controller. The results confirm how the scenario stochastic model predictive controllers give lower constraint violation probability and how the different control parametrizations have a big influence on the behaviour. The results also highlight how the $H\infty$ controller lacks the ability to coordinate the energy resources optimally to preserve the frequency. Lastly, a challenge with the scenario model predictive controller is presented with regards to the computational time for such short time step applications as frequency control. The question of the feasibility of scenario stochastic model predictive control for frequency control remains open.

Acknowledgement

This work has been carried out at the Norwegian University of Science and Technology in Trondheim at the Department of Engineering Cybernetics. This thesis will finalize a master's degree in Cybernetics and Robotics.

I would like to thank my supervisors Damiano Varagnolo and Spyridon Chapaloglou for their supervision, guidance, discussions, and valuable time. They have welcomed me with open arms into their research team and guided me through this whole year by the weekly meetings and by answering my daily questions. A special thanks to Spyridon because of his close follow up and extraordinary dedication to helping me understand, develop and carry out the research presented in this thesis.

I would also like to thank my close family Hilde, Steinar, Ingrid Johanne, and Mathilde who always supports me and encourages me. Lastly, a special thanks to my roommates Steinar Kollerud, Vebjørn Tandberg, Marius Borge Heir, and Jakob Mestvedthagen who has made this challenging year with the pandemic a year of happiness and enjoyment.

Andreas Faanes
7th June 2021

Contents

List of Figures

List of Tables

Nomenclature

AC	Alternating Current
BESS	Battery Energy Storage System
BLP	Bi-Level Programming
CIC	Critical Intertia Constant
DC	Direct Current
DER	Distributed Energy Resources
DMPC	Deterministic Model Predictive Control
DoD	Depth of Discharge
ESS	Energy Storage System
FC	Fuel Cell
FESS	Flywheel Energy Storage System
FP	Full Parametrization
FR	Frequency Response
IBG	Inverter Based Generation
ICA	Imperialist Competitive Algorithm
IMC	Internal Model Control
KiBaM	Kinetic Battery Model
LMI	Linear Matrix Inequalities
LP	Linear Programming
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
MG	Microgrid
MIMO	Multiple Input Multiple Output
MIP	Mixed Integer Programming
MISO	Multiple Input Single Output
MPC	Model Predictive Control
NLP	Non-linear Programming
NP	No Parametrization
p.u	Per Unit
PID	Proportional Integrator and Derivative controller
PV	Photovoltaic, Solar Panels
QP	Quadratic Programming
RDRP	Regional Demand Response Program
RES	Renewable Energy Source
RMPC	Robust Model Predictive Control
RoCoF	Rate of Change of Frequency

RT	Real Time
SCMPC	Scenario Model Predictive Control
SC	Supercapacitor
SFC	Secondary Frequency Control
SF	State Feedback
SISO	Single Input Single Output
SMESS	Superconducting Magnetic Energy Storage System
SMPC	Stochastic Model Predictive Control
SOC	State of Charge
SoS	Sum of Squares
SS	State Space
tf	Transfer Function

1 Introduction

1.1 Motivation

This thesis is a continuation of the work for the project in TTK 4550 in the fall of 2020. In this project, the model predictive control (MPC) strategy was investigated for use on an isolated offshore isolated power system with high penetration of renewable energy sources (RES). It was found that the MPC control strategy outperformed a benchmark multi-loop proportional integrator and derivative (PID) controller in tracking multiple reference signals for a MIMO system. It was highlighted how the MPC control algorithm manages to utilize the dynamics of the different energy production units in order to reduce the frequency deviation from the nominal reference value. In the suggestions for future work, the development of a robust MPC control strategy was proposed in order to tackle the stochastic nature of the RES.
In addition to the mentioned uncertainty in RES power generation, there will always be uncertainty in the model of the real-world plant. This can arise from both unmodeled dynamics, which are known but neglected to reduce the model complexity, and from unknown dynamics or parametric uncertainty when model parameters cannot be measured or known at perfect precision. Therefore, a controller which is robust to uncertainty is necessary for real-world applications.

There are two main topologies for handling uncertainty with the MPC controller. The robust MPC (RMPC) and the stochastic MPC (SMPC). In short, the robust approach generally gives conservative but reliable results and has for this reason been used widely. The stochastic approach is less conservative and is considered to have the potential for increased performance compared to the robust approach.

With this justification, the goal of this thesis is to investigate the performance and feasibility of SMPC for secondary frequency control. The SMPC is implemented with different control action parameterizations and tested against other control strategies, including deterministic MPC (DMPC) and mixed synthesis $H\infty$, for use on an isolated power system with RES. The three main questions which this thesis will address are

- How does scenario SMPC perform in terms of constraint violation probabilities and average performance compared to DMPC and mixed synthesis $H\infty$ control?
- Which control action parameterization is the most appropriate for our application of secondary frequency control for the isolated power system with high penetration RES?
- Is scenario SMPC feasible for secondary frequency control with a time step of 1.5 seconds?

In reality, it is unrealistic to claim that one controller is the optimal choice for all applications. The choice must depend on the nature and constraints of the problem. In this thesis, the choice of controller will be evaluated with pros and cons for different possible problem specifications.

1.2 Structure of the thesis

This master thesis is structured in the following way

- Section 2 presents the theoretical background needed to understand the implementation of the controllers and the modeling of the plant. The theory will also help interpret the results and conclusions later in section 4 and 5.
- Section 3 presents the model of the plant consisting of the gas turbine generator, wind power generator, battery energy storage system, load profile, and isolated power system dynamics. The implementation of the DMPC, $H\infty$, and SMPC controllers are also presented. Lastly, it is explained which simulations was carried out and why.
- Section 4 presents the results in the form of simulation graphs for nominal behaviour, 10% parametric uncertainty and 50% parametric uncertainty. Statistical analysis in the form of cumulative probability density function plots and performance metrics are also presented. Key observations are made and topics for further discussion are highlighted.
- Section 5 discusses the topics highlighted in section 4 and makes some concluding remarks of the performance and feasibility of the different controllers for our purpose. Lastly, suggestions for future extension of this study is introduced.

1.3 Statement of contribution

This thesis has contributed to the field of MPC for use on isolated power systems for frequency control with a short time step. Many studies have investigated the use of MPC or similar algorithms for isolated power systems with regards to optimal scheduling of RES and traditional

fossil generation for a time frame of 24 hours. Such studies include [2], [35], and [34]. This thesis contributes with a comparison of different control action parameterizations for the scenario SMPC and an investigation of the feasibility of scenario SMPC with a short time step of 1.5 seconds. Another important contribution of this thesis is the comparison of the mixed synthesis $H\infty$ controller to scenario and deterministic MPC which reveals the weakness of $H\infty$ as a MIMO controller with multiple target objectives.

2 Theoretical background

Please note that section 2.2 and 2.3 are common sections for this thesis, and the project leading to this thesis [44]. The equations, figures, and theoretical literature are mainly common however, the angle of attack is slightly different because this thesis is focused on handling uncertainty while [44] was focused on the nominal performance of MPC.

2.1 Frequency control

In an electric grid, the demanded energy from the loads connected to the grid must be balanced instantaneously by some form of energy supply. When there is an imbalance of supply and demand, the energy difference is subtracted or injected from or to the grid via the frequency of the grid. Because the infrastructure and loads connected to the grid are designed for a specific frequency, it is of vital importance to retain the frequency within certain limits. For European grids, the frequency range is typically requested to stay within $\pm 1\%$ of the nominal frequency of 50 Hz[45]. To achieve this, frequency control is necessary. Frequency control is typically divided into three categories, primary, secondary, and tertiary frequency control.
Primary frequency control is the so-called droop control which is a proportional feedback controller. This is an automatic controller which stops the frequency drop or increase in the case of an imbalance. The time frame of the primary control is "instantaneous".
The secondary control is the restoration of the frequency in the case of a frequency decrease or increase, by increasing or decreasing the power generation from the generators. The time frame of the secondary control is seconds to minutes, which is the delay or time constant of the generators adjusting to a new set point.
The tertiary control is the last step and it restores the secondary frequency control reserve after it has reached its new set point and new nominal power generation. This is typically done manually by operators. The time frame of tertiary control is typically 10-20 minutes or the time it takes to start or shut down generators.
In this thesis, we will focus on the second layer of this pyramid, secondary frequency control, by scheduling the generating units.

2.2 Challenges in isolated power systems when integrating renewable energy sources

2.2.1 Stochastic energy delivery

One of the most prominent challenges of isolated power system frequency control with high RES penetration is the stochastic nature of the energy supply[10]. Even though the average power output can be estimated by measuring over a longer period of time, frequency control is dependent on the instant power delivery. The instant power delivery can often be regarded as Gaussian noise. This applies especially for wind power where condition changes rapidly for natural reasons such as wind bursts and also because the fluid dynamics around the turbine itself is hard to model because of the turbulent conditions caused by the turbine itself and its blades[2]. Offshore wind farms are extra complex because of the turbulence created by the waves[48].

2.2.2 Lower system inertia

Lets start by explaining why system inertia matters in frequency control. The angular momentum swing equation for a synchronous generator can be expressed as

$$M \cdot \frac{d^2\delta}{dt^2} = -D \cdot \frac{d\delta}{dt} + P_m - P_l \tag{2.1}$$

Where M is the inertia constant, D is the damping coefficient, δ is the angular position , P_m is the generator power, and P_l is the load power. By using the relation between the angular position and frequency

$$\frac{d\delta}{dt} = \omega = 2\pi f \tag{2.2}$$

we obtain

$$\frac{M}{2\pi}\frac{df}{dt} = \frac{-D}{2\pi} f + P_m - P_l \tag{2.3}$$

When there are several generators connected to the same electrical grid, we can simply obtain the grid swing equation by summing up the generators

$$\sum_{i=1}^{n} M_i \frac{df}{dt} = -\sum_{i=1}^{n} D_i \cdot f + P_G - P_L \tag{2.4}$$

where M_i and D_i is the inertia constant and the damping constant of generator i with coefficients from (2.3) included, P_G is the total generated power, and P_L is the total consumed power in the grid. As one can see from equation (2.4), an imbalance between power generation and consumption will lead to a change in frequency. The magnitude of the frequency change depends on the system inertia constant and having a higher inertia constant will drive the rate of change in frequency (RoCoF) down.
The by far biggest contributor to the inertia is synchronous generators connected to the grid[1]. A synchronous generator is frequency coupled with the grid, which means that their frequencies will follow each other. Since a synchronous generator consists of several tonnes of steel that rotates, it takes a lot of energy to slow it down and it is this energy that is used to feed the power grid when the production is lower than the consumption.
Fossil energy resources use synchronous generators because they can easily control the engine which powers the generator. For RES, on the other hand, it is in most cases infeasible to use a grid-connected synchronous generator for reasons we will come back to. The exception is hydro-power where one can control the energy delivery by controlling the water flow down the pipes[49]. Unfortunately, hydro-power is limited to onshore locations with access to waterfalls or steep rivers.
Wind power generation is a possible RES for offshore isolated power systems. The wind turbine is also a rotating mass which implies that it could contribute to the grid stability even though the turbine is smaller than for fossil power generation. The problem is that wind power is usually generated from a DC generator and connected to the grid via an inverter[47]. One of the reasons for this is that the blades need to operate at a variable speed. To maximize the power transfer from the wind to the blades, the speed of the blades needs to vary with the wind speeds[47]. Therefore the power cannot be generated by a synchronous generator that operates at a constant frequency. Since there is no coupling between the wind power generator frequency and the grid frequency, the wind power generator does not contribute to the system inertia. Such types of inverter-connected power generation, which also includes solar power, is called inverter based generation (IBG)[41].

Due to the lack of system inertia in offshore isolated power systems, there is a need to virtually add this in the form of energy storage systems (ESS). There are several options, some of them are common in grids nowadays and others are in a research phase.
The battery energy storage system (BESS) is one of the chemical energy-storing options which is widely used, see for example [2], [3], [6], [10], and [34]. A large battery, typically a Lithium-ion battery, is connected to the grid and can be charged and discharged to maintain instant power balance. The advantage of the BESS is its low time constant which can give precise frequency control. The disadvantage is that the battery life cycle is not only dependent on the time in use (calendar aging), but also decreases non-linearly with the average depth of discharge (DoD) and the number of cycles.
Flywheel energy storage system (FESS) is another alternative being used today. An example of the modeling of a FESS can be found in [14]. A FESS stores energy in the mechanical energy of a rotating mass. The characteristics of the FESS are given by its mass and radius, and hence modeling it is quite straightforward. The FESS is widely used because of its simplicity and high efficiency of 80-90% [14].
Fuel cell (FC) is a developing technology that is introduced to some isolated power systems. A FC is a combustion unit that typically burns hydrogen and converts the chemical energy of the hydrogen to electrical energy through REDOX reactions[38]. This can be viewed as a battery where the energy comes from a continuous source of fuel instead of the chemicals stored in a BESS[38]. The advantage of the FC is its low environmental footprint with no need for rare chemicals and no carbon emissions.
The superconducting magnetic energy storage system (SMESS) is another alternative that is used in electrical grids. This is electrical energy stored in a superconducting coil, which can be released. The advantage of such systems is the high power which can be provided for a brief period of time with a short time constant[39]. An example of the implementation and modeling of such a system can be found in [15].
The supercapacitor (SC) is a similar alternative to the SMESS except it uses a capacitor instead of a coil to store the electrical energy. The SC has some of the same advantages as the SMESS when it comes to releasing a high amount of energy for a shorter time period and with a low time constant. The SC does not have a broad application in electrical grids today but is currently more used as a supplement to batteries in electrical vehicles[40].

2.2.3 Introduction of multiple control objectives

Adding different power production units and different ESS to the isolated power system also introduces new control objectives. A natural control objective when introducing RES is to minimize the fuel consumption of the fossil gas turbine power generator. This requires load

smoothing which adjusts the gas turbine power output to the RES generation to achieve maximal RES penetration.
The BESS also has its own control objectives. To prolong the lifespan of the battery and reduce costs, reducing the number of cycles, DoD, and charging/discharging rate must be balanced against the active frequency control to smoothen the frequency which also affects the stability of the system.

2.2.4 Constraints of the isolated power system

Another consequence of adding more units of power generation and energy storage, is more constraints[2]. We divide constraints into two main categories, soft and hard constraints.
Hard constraints are limits that should not be violated under any circumstances[50]. Some hard constraints cannot be broken, like limits from physical laws, others can be violated but causes system failure or unacceptable consequences. Examples of hard constraints in our case of the isolated power system are maximal fuel input to the gas turbine, maximal charge/discharge current of the battery, and SOC limits of the battery.
Soft constraints on the other hand are constraints that should be possible to violate if no other feasible solution can avoid it[50]. However, they should not be violated if there exist a feasible solution that does not violate the constraints. Examples of soft constraints in our case of the isolated power system are frequency deviation constraints from regulations and maximal fuel consumption from regulations. These constraints must be violated in extraordinary cases like huge power disruptions, which causes a large frequency deviation, so that the controller can restore the frequency in stead of failing because no feasible solution exist.
Another important distinction, which should be pointed out, is the difference between constraints on inputs and outputs. Our case of the isolated power system has both. Examples of input constraints are saturation limits of the battery and gas turbine. Examples of output constraints are SOC limits and frequency deviation limits. As we will discuss in section 2.3.2, MPC can handle both types while many other control algorithms like PID and LQR have no ability to adjust to output constraints.

2.3 Model predictive control

2.3.1 Basic concepts

Model predictive control is a control algorithm that solves a finite optimization problem for each time step. The optimization problem is a self selected criteria for which a model of the plant is simulated for a finite time horizon, and optimized with the control inputs as the free variables. The general optimization problem is expressed as

$$\min_u J = f(x,u), \text{ subject to} \tag{2.5}$$

$$G_I(x,u) \leq 0 \tag{2.6}$$

$$G_E(x,u) = 0 \tag{2.7}$$

where J is the cost function to be minimized under the inequality constraints G_I and equality constraints G_E. x is the state vector of the system and u is the control action vector. For every time step, the first control action is applied to the system, and the process is repeated for the next time step with new measurements of the states. This is called the receding horizon principle.
Two important parameters of MPC are the prediction horizon N_p and the control horizon N_c.

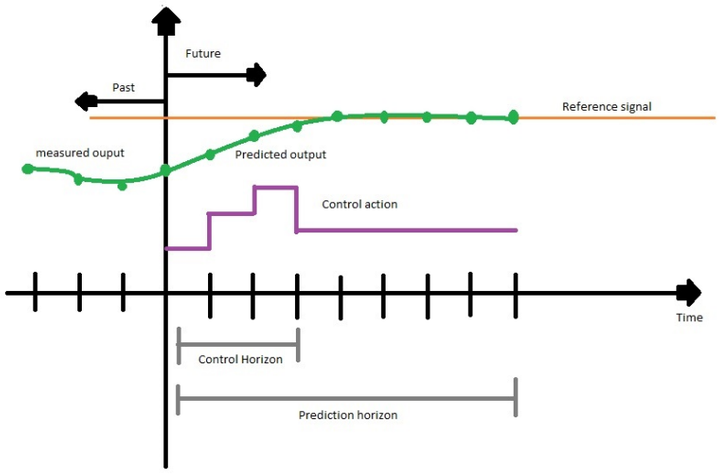


Figure 2.1: MPC parameters visualization

The prediction horizon is the number of time steps into the future for which the states are calculated and the objective function optimized. The prediction horizon has to be large enough for the system to react to upcoming events and is therefore dependent on the rise time of the system. However, a too large prediction horizon only adds computational complexity.
The control horizon is the number of time steps for which the optimal control action is calculated. This is effectively the number of free variables. The control horizon should be large enough to obtain the desired performance, but as for the prediction horizon, having a too large control horizon only adds computational complexity as the first couple of control actions are the most significant and only the first control action is actually applied. The choice of prediction horizon and control horizon is further discussed in subsection 3.2.
MPC is a discrete controller because the control action is held constant between each time step and this interval is used to calculate the next optimal input which is applied the next time step. The assumption of no change in states or control action between time steps, is called the zero-order hold principle.
A discrete MPC controller also requires a discrete state space (SS) model of the plant. Because of problem convexity, which is discussed in section 2.3.2, the plant model should preferably be a linear model on the form

$$x_{k+1} = A_d \cdot x_k + B_d \cdot u_k \tag{2.8}$$

$$y_k = C_d \cdot x_k + D_d \cdot u_k \tag{2.9}$$

where x_k is the states of the system at time step k, u_k is the control action at time step k, y_k is the system output at time k and A_d, B_d, C_d, and D_d are the state matrices.

2.3.2 Optimization problem

The optimization problem consists of two parts, the objective and the constraints. The objective is determined by the cost function J, and the choice of cost function must be carefully considered to obtain the desired performance. Typically there is a balance of complexity because the cost function needs to be complex enough to allow for precise tuning, but the more complex the cost function is, the harder the optimization problem is to solve.
However, the single most important feature to reduce complexity and solving time of a minimization problem is convexity. A convex function is defined in [33] as a function $f(\theta)$ where for every choice of $\theta^{'}$ and $\theta^{''}$ where f is defined

$$f(\alpha \cdot \theta^{'} + (1-\alpha) \cdot \theta^{''}) \leq \alpha \cdot f(\theta^{'}) + (1-\alpha) \cdot f(\theta^{''}),\ 0 \leq \alpha \leq 1 \tag{2.10}$$

This is visualized below for a single dimension function but applies for functions of all dimensions.

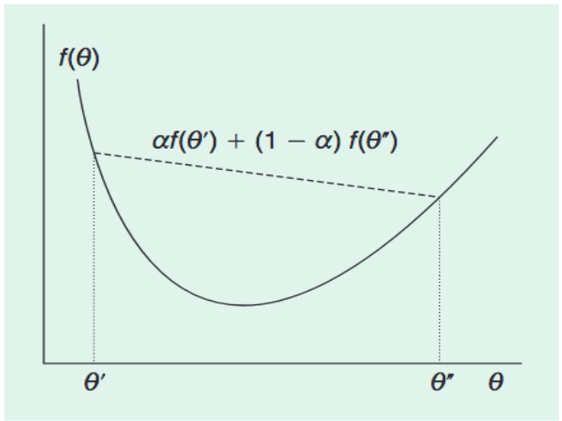


Figure 2.2: Convex function, reprinted from [4] p.41 and [44]

As one can see from figure 2.2, the local minimum is also a global minimum for the whole set where f is defined. As mentioned above, this applies also for multidimensional functions. If the function is not convex, the solver would need to find all the local minimums before it can be certain of the global minimum. This can be very time consuming.
The most common objective function for MPC is the quadratic cost function. The quadratic cost function is a weighted sum of squared error signals that can be expressed in its simplest form as

$$J(x,u) = \sum_{i=1}^{N_p} x_i^T Q x_i + \sum_{j=1}^{N_c} u_j^T R u_j \tag{2.11}$$

where Q and R are diagonal and positive definite weighting matrices. The observant reader will notice that this is the LQR cost function. Other terms can also be added to penalize specific features.
If the quadratic cost function is combined with linear constraints, the resulting problem is a QP problem which is convex.

2.3.3 Choice of solver

Solving the optimization problem is normally done by applying professional software specifically designed for the purpose. Some special cases exist where an analytical solution can be found, like the discrete MPC without constraints proposed in [16], but that strictly limits the plant model and objective function.
The optimal choice of solver is determined by the problem type. Examples of problem types, starting with the simplest form, are linear programs (LP), quadratic programs (QP), non-linear programs (NLP), bi-level programs (BLP), and mixed integer programming (MIP).

2.3.4 Advantages of MPC

Handling constraints Perhaps the foremost important feature of MPC is its ability to handle constraints. The constraints can be both linear and non-linear. Because the constraints are incorporated in the optimization problem, the solution is the optimal feasible solution and not the optimal non-feasible solution limited by the constraints. This is an important distinction from other controllers such as PID or loop shaping and is visualized below

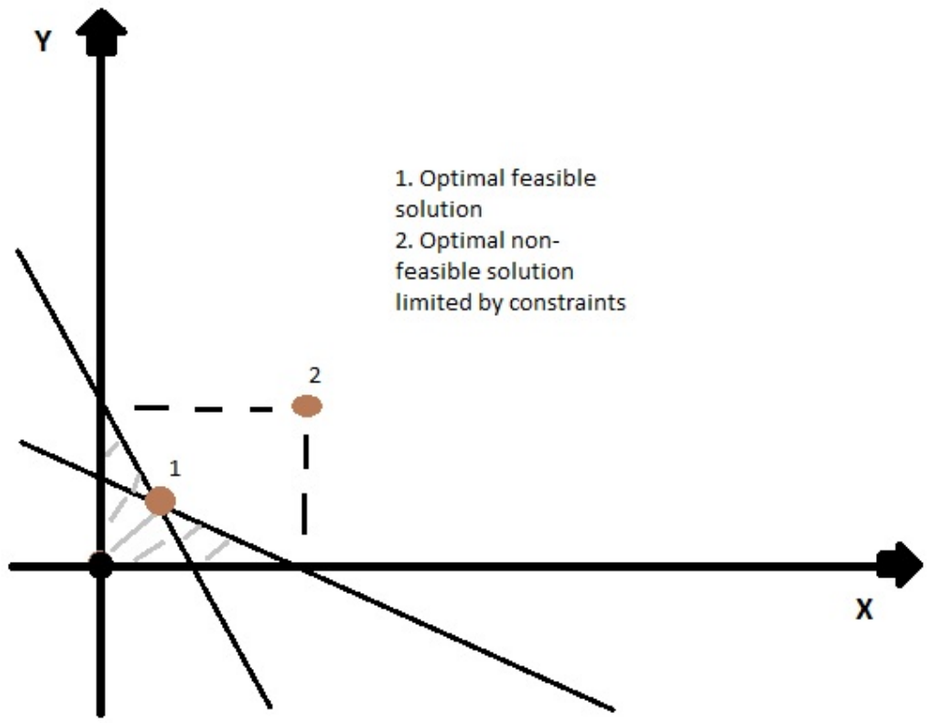


Figure 2.3: Optimal solution in feasible region vs optimal solution outside feasible region limited by constraints

MPC can also distinguish between soft and hard constraints. The definition of these two types was explained in subsection 2.2.4. The hard constraints are directly incorporated into the optimization problem via the inequality constraints G_I and equality constraints G_E. Hence the hard constraints defines the feasible region and no solution will violate these constraints.
Soft constraints, on the other hand, are not so straight forward since they must have the ability to be violated. This is often solved by adding a binary variable s and two large constants, M_1 and M_2, to the objective function and the inequality constraint. This results in

$$\min_{u,s} J(x,u,s) = \sum_{i=1}^{N_p} x_i^T Q x_i + \sum_{j=1}^{N_c} u_j^T R u_j + sM_1, \text{ subject to} \tag{2.12}$$

$$G_I(x,u) - M_2 s \leq 0 \tag{2.13}$$

$$M_1, M_2 \geq 0,\ s \in \{0,1\} \tag{2.14}$$

As one can see, setting the binary variable $s = 1$ increases the slack in the inequality constraint at the cost of adding M_1 to the cost function. As long as M_1 is sufficiently large, the binary variable will only be set to 1 if no feasible solution exist for $s = 0$. This approach is called the Big-M constraint.

Flexibility of cost function The flexibility of tailoring the cost function to the specific problem is in essence what allows us to replace multiple SISO controllers with a single MIMO controller. Because the cost function can penalize each error signal independently with the desired strength, and follow multiple references at the same time, we can achieve the same results and possibly better with one controller.

2.3.5 Challenges of MPC

Demand for computational power Perhaps the largest downside to MPC is the demand for computational power. This can be expensive because of the need for high speed processors. This downside is decreasing as modern processors become faster and cheaper but, at the time of this thesis, it is still an issue with MPC.
The more complex the optimization problem is, the more computational power is needed. Hence, reducing the problem from NLP to QP or LP, as mentioned before, is helpful. Increasing the time step can also help reduce the need for fast processing, but this might in turn affect the resolution and performance of the controller. Implementing scenario SMPC is also assumed to increase the need for computational power. This is because optimizing with regards to hundreds

of scenarios drastically increases the number of constraints and variables.
Another disadvantage is the need for an accurate model of the system. In contrast to "offline" controllers such as PID or $H\infty$, MPC predicts future behaviour and does therefore rely on a model of both plant and controller. The controller model is usually no problem since MPC is implemented as discrete controller where there is little additional time delay. The plant model on the other hand is often somewhat uncertain or inaccurate. There might be both unmodeled dynamics, which is deliberately left out due to unwanted complexity, and unknown dynamics that is too complex to be described mathematically or empirically.
Since MPC predicts future states, there is also a need to predict future disturbances or changes in the environment that affects the states. For example, in our case of the offshore isolated power system, the load condition might perform a step-increase when a large machine is connected. If the assumption of future load condition is not updated, the MPC will consequently predict the wrong frequency and not restore it to the nominal value. This introduces additional assumptions and conditions which might be more or less accurate.
The increased complexity of the MPC controller also means that there are more things that can go wrong. The physical equipment is harder to fix in case of a malfunction, and requires more educated and skilled technicians. Unexpected behaviour is also more likely to appear, which is unfortunate when a system must be reliable and robust to real-world events.

2.4 Handling uncertainty

Because the real-world is messy and complicated, there will always be uncertainty involved which needs to be taken into consideration when designing a controller. For some systems, the uncertainty is negligible and can simply be ignored without causing unwanted or unforeseen behaviour. For many systems however, the uncertainty demands a quantification and possibly a control action. We separate between two types of uncertainty: disturbance uncertainty and model uncertainty. These are explained in the next sections.
Handling uncertainty can also be categorized into two main approaches: the robust approach and the stochastic approach. The robust approach is the traditional method while the stochastic approach is a more recent development. These two main directions are discussed in section 2.5 and 2.6.

2.4.1 Disturbance uncertainty

Disturbance uncertainty is an unknown input to the system that cannot be controlled. Both the timing and magnitude might be unknown. Typically one separates disturbance uncertainty from noise. Even though this might be uncertainty originating from the same signal, they are usually of different frequencies and can therefore be handled differently. The point of this distinction will be revealed in section 2.5.2, when $H\infty$ control is introduced. Modeling disturbance uncertainty in a linear SS model yields

$$\dot{x} = Ax + Bu + Hw \tag{2.15}$$

where w is the disturbance and H is the matrix that determines the relation between w and the states in x.
Relevant examples for our case of the isolated power system are uncertain wind power generation, solar power generation, and grid power consumption.

2.4.2 Parametric uncertainty

Parametric uncertainty is inaccuracy or uncertainty in the model of the system resulting from either unknown dynamics or unmodeled dynamics. Certain dynamics can deliberately be left out to decrease the complexity of the model and allow for faster computation of the optimal solution. As we discussed earlier in section 2.3.2, the MPC solving time is drastically increased if the model is not convex, and therefore reducing the model complexity is necessary to obtain the optimal solution in the given time step.
Modeling parametric uncertainty in a linear SS formulation yields

$$\dot{x} = A(\delta)x + B(\delta)u + H(\delta)w \tag{2.16}$$

where one typically assume or claim that

$$\delta_{min} \leq \delta \leq \delta_{max} \tag{2.17}$$

where δ_{min} and δ_{max} are known boundaries or conservative estimates. If this assumption is not made, one cannot guarantee feasibility of any minimal invariant set.
Examples of unmodeled dynamics in mechanical systems can be air drag or friction. For our case of the isolated power system, the non-linear aspects of the battery model, which is simplified to the KiBaM in section 3.1.4, is an example of unmodeled dynamics.
Examples of unknown dynamics are wind turbulence around the wind turbine blades and parameters that are subject to measurement noise.

2.5 Robust control

Robust control is the conservative approach to handling uncertainty. It is conservative because it guarantees stability and/or a certain performance under the influence of the worst case uncertainty scenario. This scenario can be unlikely or even practically impossible. However, when violating constraints is unacceptable because of sever consequences or system failure, this approach is necessary. An important note is that robust control is based on the premise that the uncertainty is bounded and preferably with known limits. This is needed in order to predict or calculate the worst case uncertainty realization.
$H\infty$ loop shaping and RMPC are examples of such control strategies and are presented below.

2.5.1 Robust model predictive control

Robust model predictive control (RMPC) optimize the objective function under the influence of the worst case uncertainty scenario. This can be written mathematically as

$$\min_{u} \max_{w} J(x, u, w) \tag{2.18}$$

This approach is often denoted as the min-max optimization[25]. This approach ensures acceptable performance and guarantees that no constraints are violated when the worst case scenario occurs. However, this often leads to conservative results.
When finding the worst case realization of the disturbance, there are mainly two approaches. The theoretical approach and the scenario approach.
The theoretical approach calculates the worst case scenario by picking the disturbance that maximizes the objective function at each time step. This approach is often unnecessary conservative because it may include a combination of uncertainty realizations that is extremely unlikely or even impossible.
The scenario approach performs a random sampling over an unknown distribution, and the worst case scenario is extracted from this. This means that simulating more scenarios is more likely to obtain a worst case that is closer to the theoretical worst case.

2.5.2 Mixed sensitivity H∞ loop shaping

H∞ loop shaping is a robust control strategy from the frequency analysis domain. This method is a two stage process consisting of the open loop plant shaping and the robust controller design by $H\infty$-synthesis.
The plant shaping is based on the simple principles of loop shaping presented in [22] and summarized below.
Suppose we have a system

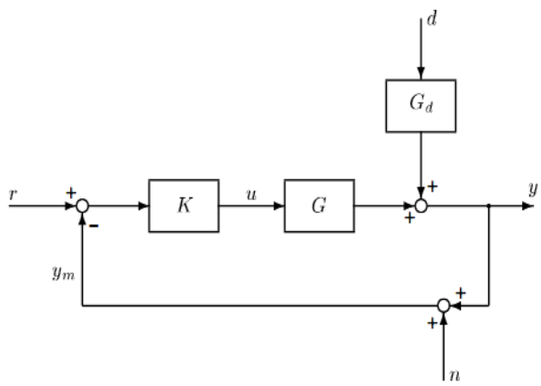


Figure 2.4: Closed loop feedback system, reprinted from [22] p.21

We can express y as

$$y = (r - y - n)GK + G_d d \tag{2.19}$$

Rewriting to isolate y gives

$$(I + GK)y = GKr - GKn + G_d d \tag{2.20}$$

$$y = \underbrace{(I + GK)^{-1}GK}_{\text{T}} r - \underbrace{(I + GK)^{-1}GK}_{\text{T}} n + \underbrace{(I + GK)^{-1}}_{\text{S}} G_d d \tag{2.21}$$

where S is called the sensitivity function and T is called the complementary sensitivity function.
When we introduce the general control objectives

1. following a reference value
2. rejecting disturbance
3. filter out noise

the fundamental aspect of feedback design arises because of the conflicting nature of these objectives. Fast disturbance rejection demands high S, fast reference tracking demands high T, and filtering out noise requires a low T. Hence there is a trade-off.
What makes loop shaping effective is the convenient fact that the different signals usually occurs at different frequencies. Reference signals and disturbance signals are usually low frequency signals, while noise is a high frequency signal. Take for instance the first order transfer function

$$K(s) = \frac{K_p}{s + 0,001}, \quad s = j\omega \tag{2.22}$$

It can be seen from inspection that $K(s) \to 1000 \cdot K_p$ when $\omega \to 0$ and that $K(s) \to 0$ when $\omega \to \infty$. Hence the desire for a low gain at high frequencies and a high gain at low frequencies is met.
For the mixed sensitivity $H\infty$ loop shaping, the controller K is obtained by solving a linear matrix inequality (LMI)

$$\left\| \begin{bmatrix} W_p S \\ W u K S \\ W_d T \end{bmatrix} \right\|_\infty \leq \gamma \tag{2.23}$$

where

$$\gamma \leq 1 \tag{2.24}$$

guarantees that the sensitivity function S, the complementary sensitivity function T, and the controller K is bounded by the weights W_p, W_d, and W_u respectively.
The choice of the weights is the demanding part of this algorithm. Especially for MIMO systems there are many considerations to make. However, the following general rules apply.
For fast and accurate reference tracking, S, which is upward bounded by $1/W_p$ given that equation (2.24) holds, should be high within the controller bandwidth. The controller bandwidth is the frequency range from $\omega = 0$ and up to the point where the controller gain is tending towards zero. Input signals outside the controller bandwidth does not affect the controller output and is simply ignored by the controller. Such signals are preferably noise signals.
For noise attenuation and disturbance rejection, T, which is upward bounded by $1/W_d$ given that equation (2.24) holds, should be high outside the controller bandwidth.
For limiting the control action within a certain frequency band, KS, which is upward bounded by $1/W_u$ given that equation (2.24) holds, should be low within the selected frequency band.
By requiring that equations (2.23) and (2.24) holds for all realizations of the uncertain plant dynamics $A(\delta)$ and $B(\delta)$, the $H\infty$ mixed sensitivity loop shaping method is robust to the parametric uncertainty and guarantees an upper bound on S, T, and KS. This can be translated into for example guarantees of maximum frequency deviation magnitude, resulting from a certain magnitude of load increase.
One big advantage of the $H\infty$ control strategy is the low computational cost. Since the controller is designed offline and only applies a pre determined control action, the time delay from input error signal to controller output is negligible. As discovered later in this thesis, MPC can be infeasible for applications with short time steps. Hence, $H\infty$ can be a good alternative strategy in those cases.
Another advantage is that the $H\infty$ control algorithm does not require a discrete model of the plant. Since the discretization leads to deviations from the real continuous plant, which is dependent on the time step and which discretization topology is used, omitting this step leads to fewer possible causes of instability.
For SISO systems, the control design is straight forward because of the power of frequency analysis. Obtaining stability can be done via analysing Bode plots or Nyquist plots, and placing the poles to achieve robust stability. However, when dealing with MIMO systems it is not so trivial. Pair vice Input-Output stability obtained by Bode plots, only guarantees stability between the given pair of input and output. However, because one input can affect several outputs, overall stability is not guaranteed. A good alternative for analysing MIMO systems is maximum singular value (MSV) plots. For further details see Skogestad [22].

2.6 Stochastic control

Stochastic control is a control strategy that does not guarantee constraint satisfaction. Instead, stochastic constraints are introduced, which allows for violation of the constraints given that it is sufficiently unlikely. The probability of violation is a design parameter that adjusts the robustness of this approach. This modern control strategy generally gives less conservative results and is suited for many types of problems. Our case of frequency control might be

one of them, because an unlikely frequency deviation outside the allowed range for a short time period, is acceptable for most systems. As opposed to robust control, stochastic control does not need to rely on the premise of a bounded uncertainty with known limits. However, in the stochastic MPC with theoretical scenario generation, assumptions on the uncertainty probability distribution are made to generate the scenarios.

2.6.1 Stochastic MPC

For stochastic MPC (SMPC), the general formulation in (2.5) to (2.7) is replaced by

$$\min_u J = f(x,u), \quad \text{subject to} \tag{2.25}$$

$$P\{G_I(x,u) \leq 0\} \geq 1-\epsilon, \quad 0 \leq \epsilon \leq 1 \tag{2.26}$$

$$G_E(x,u) = 0 \tag{2.27}$$

where ϵ is a design parameter that determines the robustness of the controller. The cost function is usually replaced by the expected value of the cost function resulting in

$$\min_u E[J] = \min_u E[f(x,u)] \tag{2.28}$$

This means that the constraints are allowed to be violated given that the probability for that to happen is less than ϵ. This leads to a less conservative controller, which presumably improves the average performance. Additionally, this can in some cases avoid infeasibility and able the solver to find a solution.
The problem with the stochastic constraints is that they in general are non-convex. The stochastic constraint is equivalent to having an infinite number of constraints. Solving an optimization problem with stochastic constraints is therefore often infeasible, at least for short time step applications such as frequency control. For this reason, the scenario approach is introduced.

Scenario optimization In the scenario approach, the chance constraints are replaced by a finite number of deterministic constraints where each constraint represent one realization, or possible realization, of the uncertainty. This transformation is validated by rigorous theory, which is presented in the work of Campi et Al. in [4] and [24]. The explanation is reproduced in this thesis and based on the same theorems and visualizations.
Theorem 1 from [24] p.5 states that

$$P\{V(u^*) > \epsilon\} \leq \sum_{i=0}^{d-1} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \tag{2.29}$$

Where u^* is the optimal solution to

$$\min_u E[f(x,u)], \quad \text{subject to} \tag{2.30}$$

$$G_{I,i}(x,u) \leq 0, \quad i \in \{1,2,...,N\} \tag{2.31}$$

$V(u^*)$ is the probability of constraint violation given the optimal solution u^*, N is the number of scenarios, and d is the number of optimization variables. By then claiming

$$\sum_{i=0}^{d-1} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \leq \beta \tag{2.32}$$

we can state that, with a probability $1-\beta$, it holds that (2.30) to (2.31) is a feasible solution to the original chance constrained problem (2.25) to (2.27). Because of the relationship between the probability β and the number of scenarios N, β is usually chosen to be in the region of 10^{-6}.
One of the main findings of the scenario approach is that one can "guarantee", with a certainty of $1-\beta$, a maximum constraint violation without knowing or assuming anything about the uncertainty distribution. The fact that it is a purely mathematical relationship between β and N, means that the scenario approach can be applied to a range of systems. At least this is true for systems where one is able to collect different scenarios from historical data. In some cases however, as mentioned in section 2.6, it is not possible to collect scenarios and they have to be generated based on an assumed probability distribution of the uncertainty.
A key distinction in scenario optimization is the one between online and offline scenario sampling. In online scenario sampling, the scenarios are influenced by the current state of the uncertainties and is therefore updated at every time step. For the offline scenario sampling, this is done beforehand, or with certain intervals, to avoid doing this at every time step. Online sampling is preferable if it is feasible with regards to the time step of the application. For our purpose of the frequency control, offline scenario sampling is performed as described in section 3.3.

Control parameterization When designing a SMPC controller, it is common to parameterize the control action as is done in [24], [27], [26], and [25]. This has been found to increase the performance of the controller in the mentioned papers for several reasons.
One advantage of parameterizing the control action, is that it limits the search space for the optimal solution, and can lead to lessening the computational burden and increase the speed of optimization.
Secondly, a structure of the control action involving a disturbance or state feedback gives a feedback structure where the control action is influenced by the state of the plant. Feeding back information in this way helps adjusting the control action for different realizations of the uncertainty and has been found to increase the effect of the number of scenarios collected[24].
The downside of some control parameterizations is the increased number of variables. There is a balance between the level of information in the parameterization and computational speed, which is discussed in [24]. In [24], they introduce four different disturbance feedback control parameterizations with different number of control variables. In this thesis, the two endpoints of these parameterizations are investigated, namely the "full parameterization" (FP) and "no parameterization" (NP).
Lastly, it should be mentioned that an important feature of the control parameterization is to preserve convexity. For this reason, a linear structure is preferred where there are no cross multiplying terms with control variables.

Scenario reduction algorithms A common strategy in the scenario approach is to perform a scenario reduction. As shown in [24], it can often improve the performance of the controller by a significant fraction while only increasing the constraint violation probability by a small amount. Often, there are only a few constraints, which are unlikely to be realized, that limit the performance of the controller. Hence, removing these will give increased performance.
There exists several scenario reduction algorithms. In [37] they define three types of scenario removal algorithms. Optimal removal, greedy removal and marginal removal. For explanation of the definitions, see [37] section 3.3. The algorithm considered for our application is the greedy algorithm presented in [24] which has the following steps

---

1. Solve the initial optimization problem
2. Find the violated constraints(zero at first interation). If number of violated constraints is equal to removal target, finish algorithm.
3. Find the active constraints.
4. For every active constraint:
Solve problem with the active constraint removed. Update optimal solution if it is lower than previous optimal solution.
end
5. Go to step 2

This algorithm is greedy because it always removes the constraint that improves the optimal solution the most. This is an effective strategy in terms of improving the performance, but as with other greedy algorithms, it performs many iterations and can be time consuming.
This algorithm is designed to be executed online at every iteration. Because the algorithm is dependent on which constraints are active, it cannot be performed offline when perfect knowledge of future events is not known. This presents a problem for applications where the time step is short like in our case of frequency control. Because the time is such a limiting factor when performing the scenario approach, such online algorithms are often infeasible.
A simple alternative to the scenario reduction is proposed. It is assumed that both the parametric and disturbance uncertainty are behaving as random Gaussian variables, and that one therefore can remove a predetermined number of random constraints and get a constraint violation probability that can be predicted by using the scenario optimization theory presented above. This algorithm is not tested in practice here, but is mentioned in section 5.6 for future extensions of the study.

2.7 Monte Carlo simulation

The Monte Carlo simulation is used for testing the control algorithms performance with respect to uncertainty. The principle of the Monte Carlo simulation is simple, it performs n number of simulations where each simulation is affected by a random realization of the uncertain parameters or disturbances. Typically, one would like to run at least 1000 simulations [42] to get the most solid results, but due to the time consuming nature of the simulations in this thesis, 100 simulations was considered the highest tolerable number of simulations. Running 100 simulations for the slowest controller took 8-12 hours on a 2010 MacBook Pro with 2,4 GHz Intel Core 2 Duo processor. Please note that this is a possible source of errors and must be taken into account when discussing the results.

2.8 Related research

Exploring energy management for isolated power systems and the stochastic MPC algorithm is currently an active field of research. Many studies are carried out across the globe in various scientific environments. In this section, some of the related studies which this thesis is inspired by and built upon are summarized.

In [2], they apply a standard MPC algorithm on a convex QP problem to control an isolated power system containing critical and non-critical loads, diesel generators, BESS, and RES in the form of photovoltaics and wind power generation. The results in this paper demonstrates how the MPC algorithm effectively manages to preserve several objectives of preserving power balance in the grid, while reducing the fossil fuel consumption of the diesel generators. They simulate scenarios with different level of accuracy for the load profile predictions ranging from a perfect forecast and up to approximately 10% deviation. [2] mirrors the study of this thesis in the way they model an isolated power system and the design of the MPC controller. However, a key difference is that [2] considers the power balance in a 24h window, while this thesis considers frequency control for a time frame of seconds to a few minutes.

In [34], the authors develop a scheduling algorithm for an isolated power system with high penetration RES, which controls the fossil energy production and the power transactions with the main grid in order to maintain power balance and maximize the RES penetration. This paper serves as an alternative strategy to the MPC algorithm and showed good results in simulations with a time frame of a day with uncertain forecasts of wind speeds and load profiles.

The paper by Patrinos et al.[35] also tries to optimize the fossil energy production and power transactions in the real time market, as in [34], but they instead develop a scenario MPC algorithm to do so. The scenarios are generated from a scenario tree to capture the additive feature of the uncertainty and avoid infeasibility. They compare their algorithm to so-called prescient optimal control which assumes perfect knowledge of future realizations of the uncertainty, and certainty equivalent MPC where uncertain parameters are substituted by their average values calculated from historical data. Results showed a large cost saving for their algorithm.

[25] performs a comparative study of what they call stochastic MPC (referred to as SMPC in this paragraph) and scenario MPC (referred to as SCMPC in this paragraph). The difference between the two methods is that the SMPC method converts the probabilistic constraints to deterministic ones, using knowledge of the co-variance of the random variables and their propagation along the prediction horizon, while the SCMPC computes scenarios and form a scenario tree, as in [35], from the probability distribution of the uncertain variables. The results show that SCMPC generate more realistic scenarios than SMPC because it uses information gathered online to adjust scenario predictions. This accuracy comes at the cost of increased computational strain.

In [24], the authors apply a scenario SMPC algorithm to a finite horizon optimization problem of a reference tracking for a mass-spring system. They compare the constraint violation probability of the scenario SMPC with various number of scenarios and the linear quadratic Gaussian (LQG) problem, which is the optimization problem without constraints. Scenario removal is done by the greedy algorithm described above in the **Scenario reduction algorithms** paragraph. Results show how effective the scenario reduction algorithm is in reducing the cost function while preserving the constraint violation probability.

In [36], the authors develop a scenario SMPC for hybrid vehicles with the goal of improving fuel efficiency while obeying constraints on battery SOC and power, and following the driver power request. However, the scenario SMPC is modified to only generate scenarios that are feasible and likely. The disturbance, which is the driver power request, is estimated via a Markov chain that predicts the future driver inputs by learning the previous request pattern in real time. Results showed that their SMPC with learning outperformed the classical MPC and, in many simulations, it was close to the performance of an MPC with perfect knowledge of future realizations of the disturbance.

3 Method

As for the theoretical section, some subsections of the methodology section are similar to the project leading to this thesis[44]. This includes section 3.1 and 3.2 because the plant model is identical, and the DMPC implementation is similar to the one in [44].

3.1 Model description

The model presented in this thesis is build in MATLAB™and Simulink™. The model of the plant is composed of Simulink™transfer functions and the MPC controller, which is implemented as an interpreted MATLAB™function. The reason for choosing MATLAB™and Simulink™, is both their simple and intuitive programming languages, and the detailed documentation from Mathworks. Another important factor is the vast example database and forums online, which provides guides to most problems. This makes the practical implementation run smoothly.

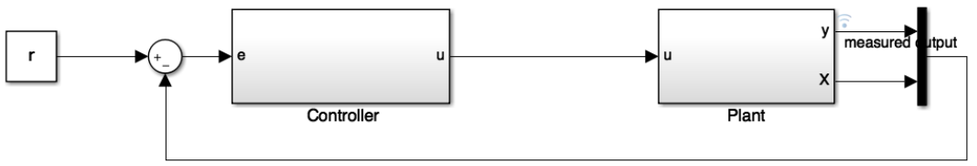


Figure 3.1: Closed loop system in Simulink™

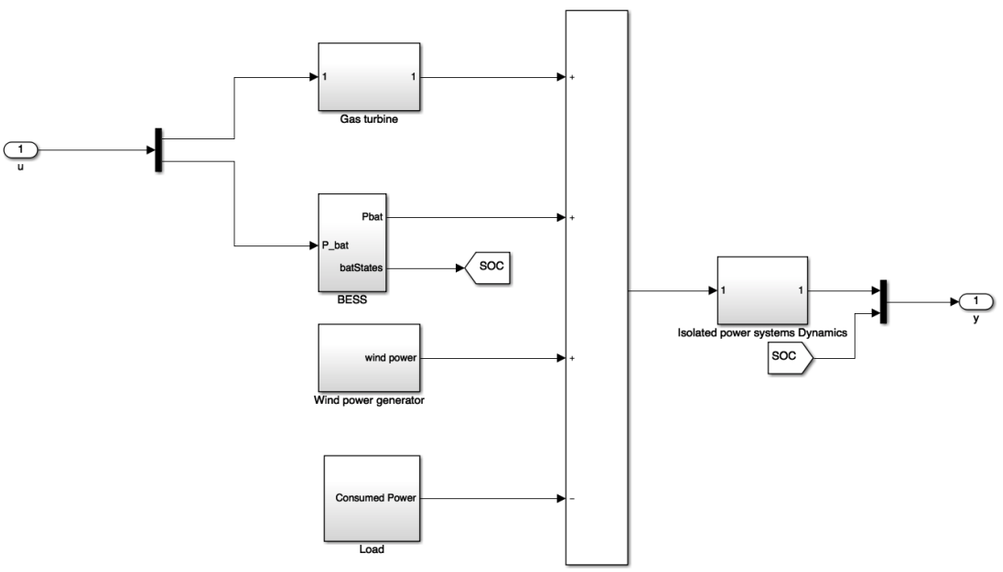


Figure 3.2: Plant model in Simulink™

Figure 3.1 shows the close loop overall structure. The only difference, compared to the topology in the report, is that the unit time delay is removed from the feedback signal. This was found to be a problem for the constraint handling of the MPC, because the feedback signal used for prediction was one time step old and hence not accurate.
All the signals are measured and scaled to a per unit (p.u.) representation. The p.u. representation scales all values relative to their nominal values. For example, a frequency deviation of 0.1 p.u. is equivalent to $0.1 \cdot 50Hz = 5Hz$ if the nominal frequency is $50Hz$. When this is done for all the signals, it absorbs large numerical differences between signals which can give numerical issues.
Another important note is that frequency, gas turbine generation, BESS charge or discharge power, and wind power generation signals are given in deviation from their nominal values. Hence, a frequency of 0 is equal to the desired nominal frequency of $50Hz$. This choice is made for simplicity. For instance, when a negative gas turbine power output occurs, this results in a negative frequency and vice versa. Another benefit is that most reference signals is 0, which allows for simplification of some MPC cost function and constraint expressions.
The isolated power system dynamics, gas turbine generator, BESS transfer function, and wind turbine generator transfer function are all based on the simple models from [3]. The advantage of using this model is its simplicity and the fact that the parameters are given. However, for

analysing a real-world example, the model should be tailor made to match the actual plant. This model is a general one, which is best suited for analysing the general case of the isolated power system.

3.1.1 Isolated power system dynamics

By applying the Laplace transform on the grid swing equation (2.4) we obtain

$$M \cdot f \cdot s = -D \cdot f + P_g - P_L \tag{3.1}$$

By rearranging and substituting $P_G - P_L$ with ΔP we get

$$\frac{f}{\Delta P} = H_{GD}(s) = \frac{1}{D + M \cdot s} \tag{3.2}$$

where f is the frequency deviation, D is the total grid damping constant, and M is the total grid inertia constant. D and M have nominal values $D_0 = 1$ p.u/Hz., $M_0 = 3$ p.u. s/Hz and an uncertain part D_δ, M_δ. Hence

$$\begin{aligned} D &= D_0 \pm D_\delta \\ M &= M_0 \pm M_\delta \end{aligned} \tag{3.3}$$

After converting the continuous model to a discrete transfer function using the MATLAB™function $c2d(tf, \Delta t)$, which uses zeros order hold and the equations presented in section 3.1.6, we implement the transfer function using a Simulink™discrete transfer function block.

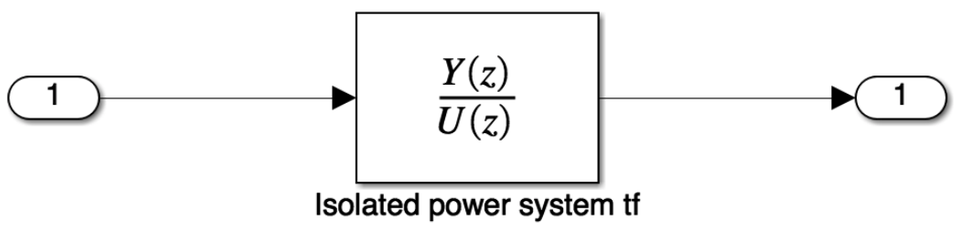


Figure 3.3: Isolated power system subsystem in Simulink™

3.1.2 Gas turbine

The model of the gas turbine generation unit is a simplified version of the units presented in [3] where the droop control, which is a primary frequency control strategy, is left out. This is done for simplicity. The gas turbine generation unit then consist of two parts. The governor and the turbine generator. Each with its own time constant and hence it is modeled as two separate first order transfer function connected in series

$$H_{GT}(s) = H_G(s) \cdot H_T(s) = \frac{1}{1 + T_g \cdot s} \times \frac{1}{1 + T_t \cdot s} \tag{3.4}$$

where T_g is the governor time constant and T_t is the turbine generator time constant. The parameters T_g and T_t have the same uncertainty topology as for the isolated power system constants

$$\begin{aligned} T_g &= T_{g,0} \pm T_{g,\delta} \\ T_t &= T_{t,0} \pm T_{t,\delta} \end{aligned} \tag{3.5}$$

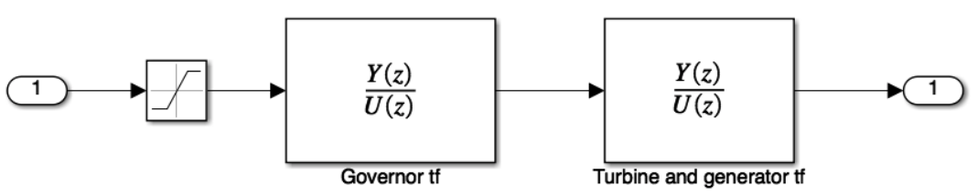


Figure 3.4: Gas turbine generator subsystem in Simulink™

3.1.3 Wind Power generation

The wind power generation in this model consist of two separate parts, the wind speed generator and a first order wind speed to power transfer function, which is presented in [3]. The choice of this topology was made to replicate the structure of the real-world wind power generation and to be able to simulate specific wind conditions if wanted in future research.
The random wind speed generator takes an average wind speed and generates a realistic wind speed sample based on a model of the hydrodynamic effects locally around the wind turbine and rotor blades. The model includes parameters such as the wind turbine hub height, rotor radius, turbulence intensity, and turbulence length scale parameter to name a few. The overall structure of the wind speed generator is provided below in figure 3.5 and further documentation is given in appendix A. Please note that the wind speed generator is not developed by the author. It is developed by Spyridon Chapaloglou based on the theory presented in [20] and [21].

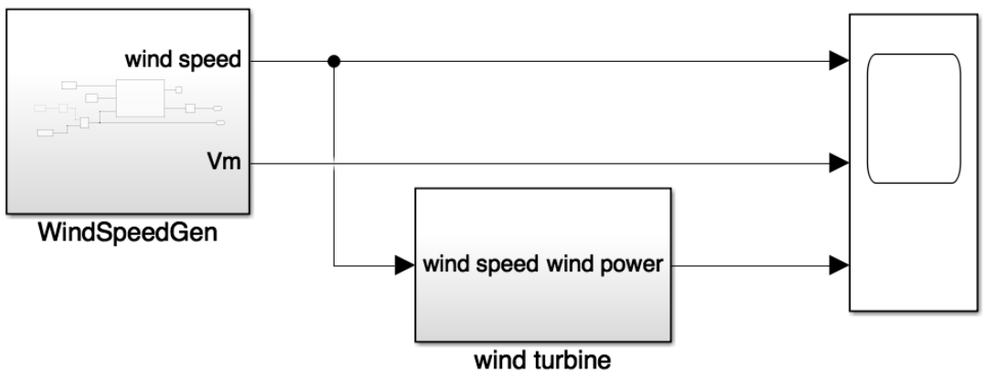


Figure 3.5: Wind speed generator in Simulink™

An added layer in the wind speed to wind power conversion is the cut inn and cut off speeds. These are important features of a wind turbine. The cut inn (low cut) speed is the lowest wind speed where the torque from the rotor blades is high enough to start spinning the turbine. Hence, no power is provided at lower wind speeds. The cut off (high cut) wind speed is the highest wind speed the wind turbine is designed to handle and not be damaged. It is not economically viable to scale the turbines to withstand the massive forces from a storm if this seldom occurs, and to obtain the maximal efficiency at the nominal wind speeds the turbine is scaled thereafter. When the wind speeds is too high, the blades are pitched to give no torque on the turbine and therefore no electrical power. A typical wind-power curve is presented in [18] and reprinted below

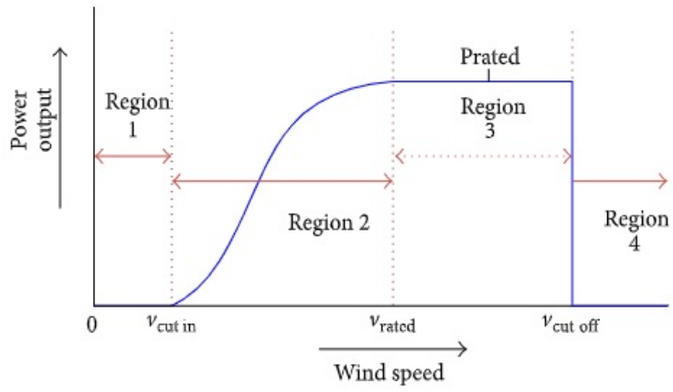


Figure 3.6: Typical wind-power curve, reprinted from [18] and [44]

The non-linear region 2 in figure 3.6 can be modeled in several different ways. Some of whom are presented in [18] and includes linear, quadratic, binomial, cubic, and exponential. The optimal approximation depends on the shape of the wind turbine, the local terrain, and wind speed profiles to name a few. For this model the quadratic approximation was chosen because it was considered the best trade off between simplicity and accuracy. The biggest jump in accuracy was considered to be from the linear approximation to the quadratic one. This results

in the following wind speed to wind power transformation

$$P_{WT} = \begin{cases} 0 & 0 \leq V \leq V_{ci} \text{ or } V \geq V_{co} \\ P_{r,WT} \times (\frac{V - V_{ci}}{V_r - V_{co}})^2 & V_{ci} \leq V \leq V_r \\ P_{r,WT} & V_r \leq V \leq V_{co} \end{cases} \tag{3.6}$$

where V is the wind speed, V_{ci} is the cut inn speed, V_{co} is the cut off speed, V_r is the rated wind speed, and $P_{r,WT}$ is the nominal power generation at the rated wind speed. Equation (3.6) was implemented in Simulink™by making a block diagram which is presented in figure 3.7.
The last piece of the wind turbine is the actual turbine and generator transfer function. This was extracted from [3] and yields

$$H_{WTG}(s) = \frac{1}{1 + T_{WTG} \cdot s} \tag{3.7}$$

where T_{WTG} is the wind turbine generator time constant. T_{WTG} has the same uncertainty topology as for the isolated power system

$$T_{WTG} = T_{WTG,0} \pm T_{WTG,\delta} \tag{3.8}$$

The overall wind speed to wind power transfer function block is then

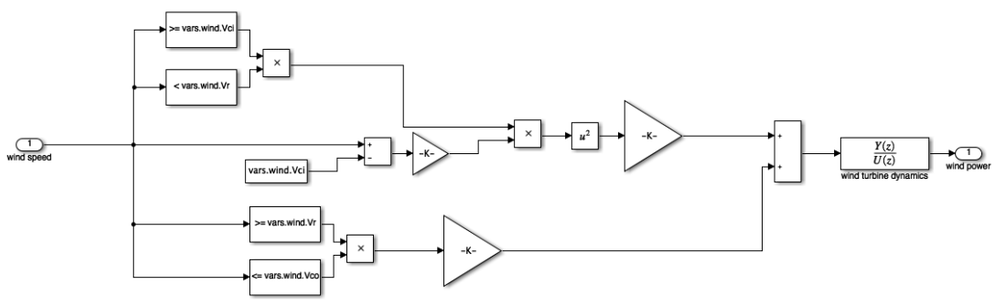


Figure 3.7: Wind speed to wind power transfer function in Simulink™

3.1.4 Battery energy storage system

The simulated battery energy storage system (BESS) consist of an interpreted MATLAB™function and a first order transfer function from requested power output to actual power output. This transfer function is extracted from [11] and yields

$$H_{BESS}(s) = \frac{1}{1 + T_{BESS} \cdot s} \tag{3.9}$$

where T_{BESS} is the BESS time constant. T_{BESS} has the same uncertainty topology as for the isolated power system

$$T_{BESS} = T_{BESS,0} \pm T_{BESS,\delta} \tag{3.10}$$

The interpreted MATLAB™function, which is the modeling of the battery state of charge (SOC), is derived from the so called kinetic battery model (KiBaM) presented in [19] and summarized below. This summary is based on the same equations as those stated in the project that this thesis is a continuation of.
The battery is modeled as a simple circuit with a voltage source and a resistance.

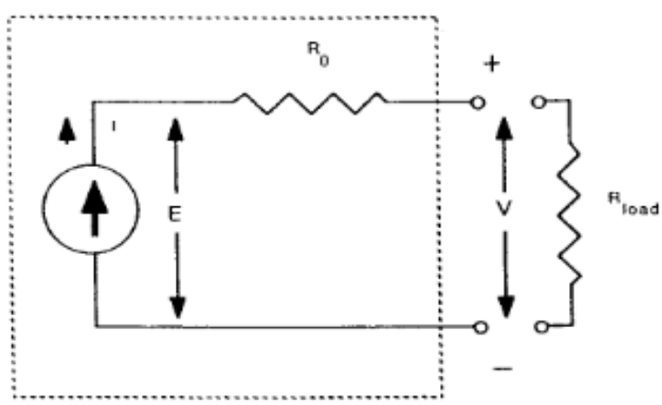


Figure 3.8: Battery model circuit. Reprinted from [19]

In figure 3.8, E is the battery internal voltage, R_0 is the battery internal resistance, and R_{load} is the connected load with the terminal voltage V.
The idea of the KiBaM model is to include the effect of the chemical reaction from chemically bounded energy to available charge that can be used by the load. The voltage source is therefore modeled to include two tanks, one for the chemically bounded charge, and one for the available charge. The tanks are separated by a conductance, which determines the flow rate from one tank to the other. This is visualized below

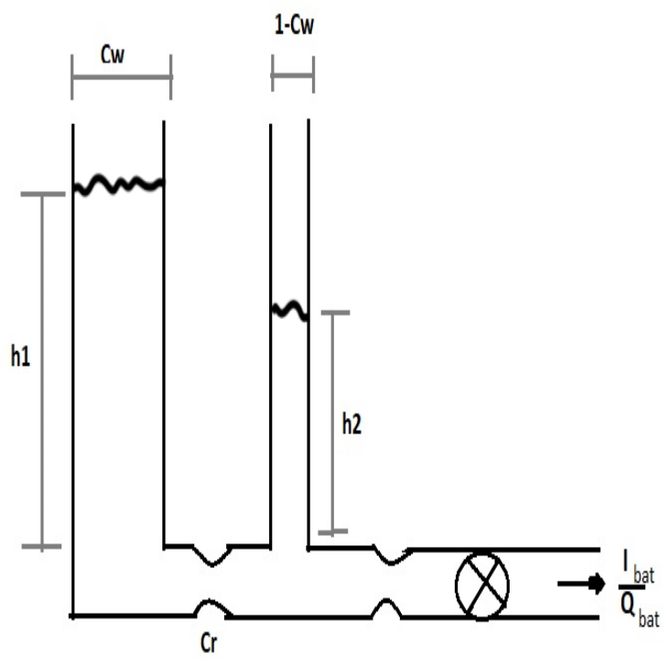


Figure 3.9: Drawing of the KiBaM tank model according to [19]

The equations describing the tank in figure 3.9 are

$$\dot{x_1} = -\frac{I_{bat}}{Q_{bat}} - c_r \cdot (h_1 - h_2) \tag{3.11}$$

$$\dot{x_2} = c_r \cdot (h_1 - h_2) \tag{3.12}$$

where x_1 and x_2 are the volumes of the two tanks, c_r is the fixed conductance between the two tanks, I_{bat} is the output current flow, and Q_{bat} is the maximal current flow. By using the relation

$$h_1 = \frac{x_1}{c_W} \tag{3.13}$$

$$h_2 = \frac{x_2}{1 - c_W} \tag{3.14}$$

and inserting this into (3.11) and (3.12) we get

$$\dot{x_1} = -\frac{I_{bat}}{Q_{bat}} - c_r \cdot (\frac{x_1}{c_W} - \frac{x_2}{1 - c_W}) \tag{3.15}$$

$$\dot{x_2} = c_r \cdot (\frac{x_1}{c_W} - \frac{x_2}{1 - c_W}) \tag{3.16}$$

which we can rewrite to a linear SS model

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \mathbf{A}_{bat} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -Q_{bat}^{-1} \cdot I_{bat} \\ 0 \end{bmatrix} \tag{3.17}$$

where

$$\mathbf{A}_{bat} = \begin{bmatrix} -\frac{c_r}{c_W} & \frac{c_r}{1-c_W} \\ \frac{c_r}{c_W} & -\frac{c_r}{1-c_W} \end{bmatrix} \tag{3.18}$$

The relation between the battery power and the battery current can be obtained by using the electric power law on the circuit in figure 3.8

$$P_{bat} = V \cdot I_{bat} - R_0 \cdot I_{bat}^2 \tag{3.19}$$

where P_{bat} is the battery power and I_{bat} is the battery current. This non-linear relation can be solved for I_{bat} and then linearized around the real roots to give the following relation

$$I_{bat} \approx \frac{\eta}{V \cdot P_{bat}} \tag{3.20}$$

where η is the charge/discharge efficiency. A deeper understadning of why and how the linearization is performed can be found in [11]. By combining (3.20) and the definition

$$C_{bat} = Q_{bat} \cdot V \tag{3.21}$$

where C_{bat} is the battery capacity, and inserting this into (3.17) we obtain the KiBaM

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \mathbf{A}_{bat} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + C_{bat}^{-1} \cdot \begin{bmatrix} -\eta_{gen}^{-1} & \eta_{load} \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} u_{gen} \\ u_{load} \end{bmatrix} \tag{3.22}$$

$$u_{gen} = \begin{cases} 0 & P_{bat} < 0 \\ P_{bat} & P_{bat} \geq 0 \end{cases} \tag{3.23}$$

$$u_{load} = \begin{cases} 0 & P_{bat} > 0 \\ -P_{bat} & P_{bat} \leq 0 \end{cases} \tag{3.24}$$

where η_{gen} is the discharge efficiency, η_{load} is the charge efficiency, u_{gen} is the discharging power, and u_{load} is the charging power.
Due to a non-linearity in the constraint formulation for the charge and discharging power when implementing the MPC, the charge/discharge efficiencies were neglected, which allows for combining the charge and discharge power into P_{bat}. In short, a binary variable had to be introduced to ensure that charging and discharging could not be done simultaneously and the multiplication term of the binary variable and $u_{gen/load}$ was non-convex. Hence, we get the following simplified KiBaM

$$\dot{\mathbf{x}} = \mathbf{A}_{bat} \cdot \mathbf{x} - \begin{bmatrix} C_{bat}^{-1} \\ 0 \end{bmatrix} \cdot P_{bat} \tag{3.25}$$

where

$$SOC = x_1 + x_2 \tag{3.26}$$

In this model c_r, c_W, and C_{bat} are uncertain parameters and have the same uncertainty topology as the other parameters

$$\begin{aligned} c_r &= c_{r,0} \pm c_{r,\delta} \\ c_W &= c_{W,0} \pm c_{W,\delta} \\ C_{bat} &= C_{bat,0} \pm C_{bat,\delta} \end{aligned} \tag{3.27}$$

Please note that the fact that C_{bat} is uncertain in both directions is somewhat unrealistic. In reality, the upper limit to C_{bat} is often known, while the reduction in C_{bat} due to battery degradation leads to uncertainty in how much of the capacity is lost. This is an improvement which should be made for future work.
An important feature for reducing the cost of the BESS, is to reduce the cycling aging. In addition to reducing the DoD, this means penalizing the cycling of the battery. As for many other aspects of modeling, this is a trade off between capturing the cycling accurately and preserving convexity and simplicity in the model.
To preserve convexity, a simple battery cycling penalty mechanism is chosen for this model. Namely the SOC standard deviation. The analytical expression for the standard deviation is given in the cost function description in section 3.2.2. The standard deviation metric captures the cycling as variation and penalizes this. Hence, the number of cycles is not counted and DoD is not explicitly penalized. However, since a large DoD also implies a large variation given that the average SOC is 50%, DoD is implicitly penalized.
Some alternatives are penalizing the integral of the SOC deviation, penalizing the battery charge/discharge power (also included in this thesis), and the Rainflow algorithm. The Rainflow algrithm is a more advanced technique for fatigue analysis that counts the number of cycles from a finite history of values, which may contain both large and small cycles[28]. This method is refined over many years by a range of industries. The problem with such algorithms is that they are almost exclusively non-convex algorithms. Another problem is that they typically are designed for a time horizon that is much longer and contains more cycles than our purpose of frequency control with a time horizon of seconds. Hence, they are not incorporated into this model. For further detail of the Rainflow algorithm applied to SOC profiles, see [28].
An alternative idea to Rainflow counting, is to develop a linear regression model between one or more SOC metrics and the cycle count. This could be done by retrieving a large data set of SOC profiles and applying the Rainflow algorithm to each of these profiles. Hopefully, it is possible to observe and estimate a linear or convex relationship between metrics like SOC standard deviation or curve integral, and the cycle count. This idea is not explored or tested further in this thesis for time limiting reasons, but is discussed in section 5.6 as a topic for further research.

3.1.5 Load profile

A study on what is a typical load profile for an isolated power system is performed by Hartvigsson and Ahlgren in [29]. The example profile they ended up with is shown below in figure 3.10. The load profile is the thick black line and it is plotted for 24 hours with one minute resolution.

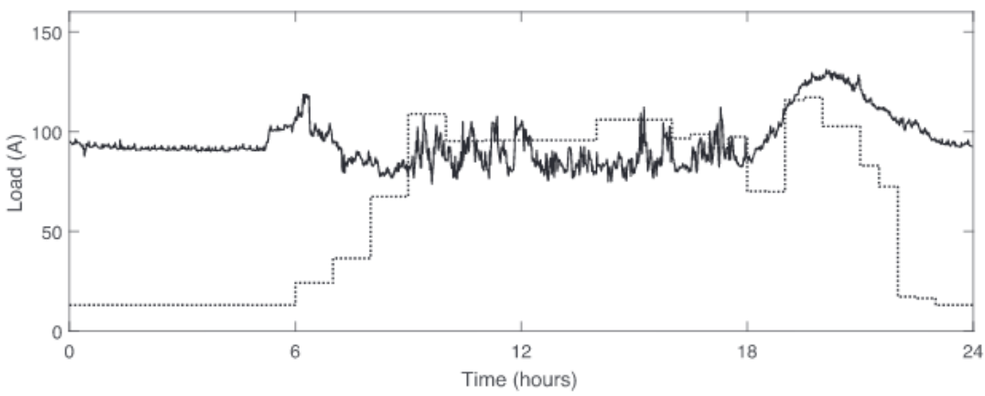


Figure 3.10: Typical 24 hours Load profile for electrical grid. Reprinted from [29]

As we can observe, there are some characteristics that are likely to repeat, like the fact that the consumption rises in the morning around 6-8, and in the evening from around 18-22. This makes perfect sense because the average man typically wakes up at 6 and cooks breakfast, goes to work at 8, and comes home to make dinner and watch TV at 18. These types of trends are also likely to appear for offshore isolated power systems. However, what we can also observe from the profile is its stochastic behaviour in the short time frame. Since frequency control is dependent on generation/consumption balance in the time frame of seconds, it is the short time frame that is relevant for our case. The stochastic behaviour we observe in figure 3.10 can be modeled as Gaussian white noise[29].
In this thesis, the load is modeled as a constant reference signal. This simplification was made to avoid introducing too many variables of uncertainty at once, because separating the effect of them would be more difficult. The uncertainty of wind power generation and in model parameters was considered to be a sufficient test of the MPC controllers. The load modeling also include step increases/decreases. This simulates connecting or disconnecting a big load like a drilling machine for instance, which is a likely scenario for an offshore oil & gas platform.

Figure 3.11: Load subsystem in Simulink™

3.1.6 State space representation

As discussed in section 2.3.1, to calculate future states it is convenient to express the model as a state space (SS) formulation. A SS formulation is not unique and for a given system there may exist several SS formulations. A challenge when transforming the plant from a transfer function topology to a SS model, was to obtain the desired set of states. The MATLAB™function $tf2ss(sys)$ outputs a SS model, but no additional information on what the states are in correspondence with the transfer function topology. The desired states are defined in figure 3.12 below

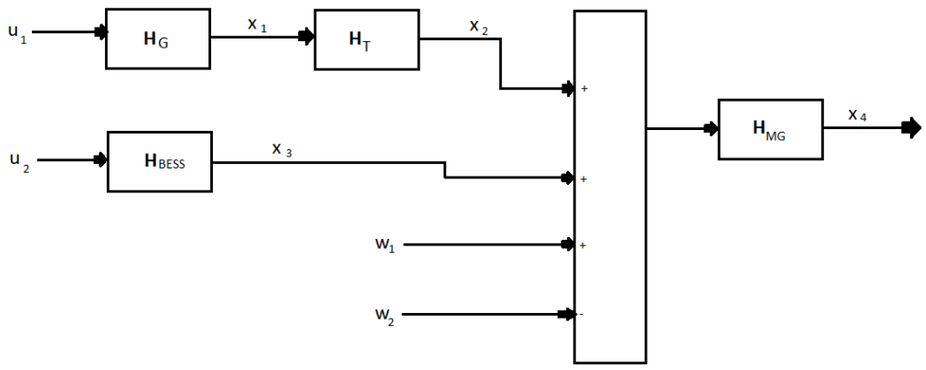


Figure 3.12: Overview of system topology with transfer function, states, inputs, and disturbances

This was solved by applying $tf2ss(sys)$ to every singe transfer function. Since they are of single order, the resulting SS is also of single order and the states are easily identified. The result of this is shown below

$$\begin{aligned} \dot{x_1} &= -\frac{1}{T_g}x_1 + \frac{1}{T_g}u_1 \\ \dot{x_2} &= -\frac{1}{T_t}x_2 + \frac{1}{T_t}x_1 \\ \dot{x_3} &= -\frac{1}{T_{BESS}}x_3 + \frac{1}{T_{BESS}}u_2 \\ \dot{x_4} &= -\frac{1}{M}x_4 + \frac{1}{M}(x_2 + x_3 + w) \end{aligned} \tag{3.28}$$

Note that for all the first order SS models, $y_i = x_i$ and hence $C = 1$. This can be transformed into a single SS representation where we also include the battery states x_5 and x_6

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{H}\mathbf{w} \tag{3.29}$$

with

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}, \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} -\frac{1}{T_g} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{T_t} & -\frac{1}{T_t} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{T_BESS} & 0 & 0 & 0 \\ 0 & \frac{1}{M} & \frac{1}{M} & -\frac{1}{M} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{c_r}{c_W} & \frac{c_r}{1-c_W} \\ 0 & 0 & 0 & 0 & \frac{c_r}{c_W} & -\frac{c_r}{1-c_W} \end{bmatrix} \tag{3.30}$$

$$\mathbf{B} = \begin{bmatrix} \frac{1}{T_g} & 0 \\ 0 & 0 \\ 0 & \frac{1}{T_BESS} \\ 0 & 0 \\ 0 & -\frac{1}{C_{bat}} \\ 0 & 0 \end{bmatrix}, \mathbf{C}^T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{M} & -\frac{1}{M} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Since the parameters are uncertain, the SS formulation is expressed in the same topology, namely

$$\begin{aligned} \mathbf{A} &= \mathbf{A_0} \pm \mathbf{A}_\delta \\ \mathbf{B} &= \mathbf{B_0} \pm \mathbf{B}_\delta \\ \mathbf{H} &= \mathbf{H_0} \pm \mathbf{H}_\delta \end{aligned} \tag{3.31}$$

where the procedure above is applied when finding the nominal matrices and uncertain matrices with the nominal parameters and uncertain parameters respectively. The resulting system is

then transformed from the continuous system above, to the discrete system presented in section 2.3.1, by assuming zero-order hold

$$\begin{aligned}\mathbf{A_d} &= e^{\mathbf{A}T} = \mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{A})^{-1}\} \\ \mathbf{B_d} &= \mathbf{A}^{-1}(\mathbf{A_d} - \mathbf{I})\mathbf{B} \\ \mathbf{C_d} &= \mathbf{C} \\ \mathbf{D_d} &= \mathbf{D}\end{aligned} \tag{3.32}$$

where T is the time step, and $\mathcal{L}^{-1}$ is the inverse Laplace transform. This type of uncertainty, where the parameters have bounded uncertainty is called structured uncertainty. The counterpart is when one have bounded uncertainty in the total system itself, but no knowledge of the individual parameter uncertainty. The latter is a more conservative approach because it may include combinations of parameter uncertainty that is not even possible.

3.2 MPC implementation

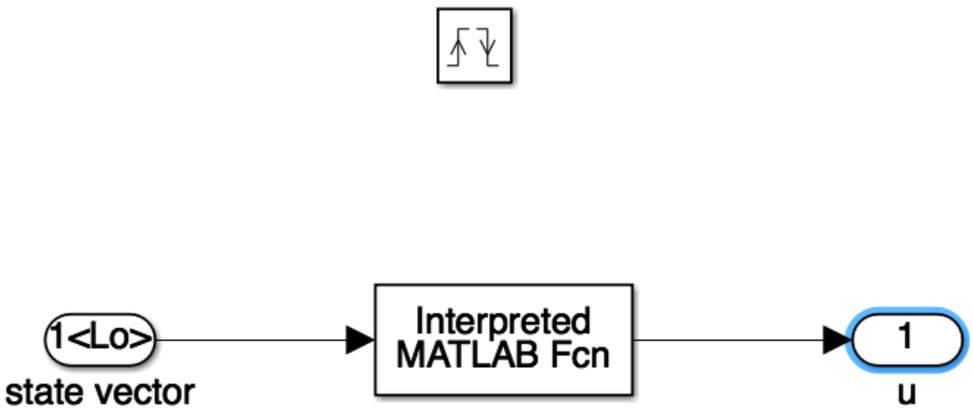


Figure 3.13: MPC controller subsystem in Simulink™

The MPC controller was implemented in simulink using an Interpreted MATLAB™function, as seen above in figure 3.13. The subsystem also includes a trigger function, which calls the Interpreted MATLAB™function every time step, and a latch on the input signal to avoid an algebraic loop.

3.2.1 MPC parameters

The choice of MPC parameters was thoroughly discussed in the project leading to this thesis. The MPC parameters were there selected based on some rules of thumb, which is presented by Mathworks in [30]. They state that the time step Δt should be in the region 10-20% of the system rise time. The rise time was found to be approximately 7 seconds for our plant. Therefore Δt was initially chosen to be $\Delta t = 0.7s$. However, in order to come closer to feasibility for the scenario SMPC strategies, this was increased to $\Delta t = 1.5s$ which is 21% of the rise time and therefore at the upper limit of the rule of thumb
The guideline for the control horizon is to cover the rise time of the system at least one time. This is the lower limit because it is crucial to have enough time to react to future events. To make sense of this rule, imagine a car cruising at speed. If the breaking distance of the car is 40 meters, then the prediction horizon needs to cover 40 meters into the future, or else the car will not be able to stop if for instance a traffic junction appears. For our case, $N_p = 10$ was chosen which means that the rise time is covered approximately two times. Have in mind that a too long prediction horizon gives unnecessary high computational cost, so it is important to keep the prediction horizon low.
When it comes to choosing the control horizon, the rule of thumb states that it should be approximately 10-20% of the prediction horizon. As for the prediction horizon, having a too high control horizon only adds computational cost. Having a too low control horizon on the other hand, reduces the degrees of freedom with regards to the control action and will give bad resolution and accuracy in the reference tracking. For our system, $N_c = 4$ was chosen as for the project. This is definitely in the upper end of the advised region, and is perhaps a topic for further studies in order to reduce the calculation time.

3.2.2 Cost function formulation

The quadratic cost function is by far the most common structure because it allows for intuitive and simple tuning. But more importantly because it is a convex function given that the weights are positive, or equivalently the weighting matrices are positive definite. This choice was considered obvious and therefore, no other cost function structures were explored.
The first objective included in the cost function is penalizing the frequency deviation. This is the main objective of the controller and must therefore be accompanied with a high cost. The frequency is given by the state x_4 which gives the term $\mathbf{x_4^T Q x_4}$ where

$$\mathbf{x_4^T} = [x_{4,1}, x_{4,2}, \ldots, x_{4,N_p}] \tag{3.33}$$

and

$$\mathbf{Q} = diag([Q_1, Q_2, \ldots, Q_{N_p}]) \tag{3.34}$$

Secondly we want to penalize SOC deviation from the reference value, to minimize battery degradation. SOC is found by equation (3.26) and gives the term $\mathbf{SOC^T \cdot L \cdot SOC}$ where

$$\mathbf{SOC^T} = [SOC_1, SOC_2, \ldots, SOC_{N_p}] \tag{3.35}$$

and

$$\mathbf{L} = diag([L_1, L_2, \ldots, L_{N_p}]) \tag{3.36}$$

To penalize battery degradation more accurately, penalizing the cycling is needed because it is an accelerating factor for the degradation. This is done by penalizing the standard deviation of the SOC. This gives the term

$$M \cdot \sqrt{\frac{1}{N_p}\sum_{k=1}^{N_p}\left(SOC_k - \frac{1}{N_p}\sum_{k=1}^{N_p} SOC_k\right)^2} \tag{3.37}$$

where M is a constant scalar.
Lastly we want to penalize the control action. This is often included as a standard term in cost functions because in reality, the control action results in fatigue on the system input and possibly the system itself. For our system, the change in power output from the gas turbine leads to mechanical fatigue on the governor, and the change in BESS output power lead to battery degradation. For our purpose it is also a goal to minimize the fuel consumption of the gas turbine. Because the average of the gas turbine power output and the wind turbine power output combined has to match the average consumed power, it is not appropriate to penalize the absolute value of the gas turbine input. This would lead to a trade off between frequency tracking and gas turbine input in the case of a step change in the power consumption, which would result in a steady state frequency deviation.
Because of this and because the change in gas turbine input often affect the fuel consumption more than the absolute value of the gas turbine input because of the low steady state fuel consumption of the gas turbine, the change in control action Δu is penalized. The control action penalizing term is included in the cost function as $\mathbf{\Delta u^T R \Delta u}$ where

$$\mathbf{\Delta u^T} = [\mathbf{\Delta u_1}, \mathbf{\Delta u_2}, \ldots, \mathbf{\Delta u_{N_c}}] \tag{3.38}$$

and

$$\mathbf{R} = diag([[R_{1,1}R_{2,1}], [R_{1,2}R_{2,2}], \ldots, [R_{1,N_c}R_{2,N_c}]]) \tag{3.39}$$

For the BESS however, penalizing the absolute value might have been more appropriate. This is a topic for further discussion.
Hence we have the following cost function objectives

- minimize frequency deviation from reference
- minimize SOC deviation from reference
- penalize the BESS degradation
- penalize the control action

and the resulting cost function J can then be expressed as

$$J = \mathbf{x_4^T Q x_4} + \mathbf{SOC^T \cdot L \cdot SOC} + M \cdot \sqrt{\frac{1}{N_p}\sum_{k=1}^{N_p}\left(SOC_k - \frac{1}{N_p}\sum_{k=1}^{N_p} SOC_k\right)^2} + \mathbf{\Delta u^T} R \mathbf{\Delta u} \tag{3.40}$$

3.2.3 Recursive elimination

As we can see from the expressions above, the future N_p states from x_0 needs to be calculated at every time step. This can be solved easily by recursive elimination of the states.

$$\begin{aligned}\mathbf{x_1} =& \mathbf{A x_0} + \mathbf{B}(\mathbf{u_0} + \Delta\mathbf{u_1}) + \mathbf{H w_1} \\ \mathbf{x_2} =& \mathbf{A x_1} + \mathbf{B}(\mathbf{u_1} + \Delta\mathbf{u_2}) + \mathbf{H w_2} = \\ & \mathbf{A}(\mathbf{A x_0} + \mathbf{B}(\mathbf{u_0} + \Delta\mathbf{u_1}) + \mathbf{H w_1}) + \mathbf{B}(\mathbf{u_0} + \Delta\mathbf{u_2} + \Delta\mathbf{u_2}) + \mathbf{H w_2}\end{aligned} \tag{3.41}$$

This pattern continues and we end up with the following expression

$$\mathbf{X} = \mathbf{A_e x_0} + \mathbf{B_0 u_0} + \mathbf{B_e \Delta U} + \mathbf{H_e W} \tag{3.42}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \vdots \\ \mathbf{x_{N_p}} \end{bmatrix}, \mathbf{U} = \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \\ \vdots \\ \mathbf{u_{N_c}} \end{bmatrix}, \mathbf{W} = \begin{bmatrix} \mathbf{w_1} \\ \mathbf{w_2} \\ \vdots \\ \mathbf{w_{N_p}} \end{bmatrix} \tag{3.43}$$

and

$$\begin{aligned}\mathbf{A_e} &= \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \vdots \\ \mathbf{A}^{N_p} \end{bmatrix}, \mathbf{B_0} = \begin{bmatrix} \mathbf{B} \\ \mathbf{AB} + \mathbf{B} \\ \vdots \\ \mathbf{A}^{N_p-1}\mathbf{B} + \mathbf{A}^{N_p-2}\mathbf{B} + \cdots + \mathbf{A}^{N_p-N_p}\mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{B} \\ \mathbf{AB} + \mathbf{B} \\ \vdots \\ \sum_{i=1}^{N_p} \mathbf{A}^{N_p-i}\mathbf{B} \end{bmatrix}, \\ \mathbf{B_e} &= \begin{bmatrix} \mathbf{B} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{AB} + \mathbf{B} & \mathbf{B} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{N_p} \mathbf{A}^{N_p-i}\mathbf{B} & \sum_{i=2}^{N_p} \mathbf{A}^{N_p-i}\mathbf{B} & \dots & \sum_{i=1+N_c}^{N_p} \mathbf{A}^{N_p-i}\mathbf{B} \end{bmatrix}, \\ \mathbf{H_e} &= \begin{bmatrix} \mathbf{H} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{AH} & \mathbf{H} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{N_p-1}\mathbf{H} & \mathbf{A}^{N_p-2}\mathbf{H} & \dots & \mathbf{A}^{N_p-N_p}\mathbf{H} \end{bmatrix}\end{aligned} \tag{3.44}$$

3.2.4 Constraints formulation

There are a few hard physical constraints in this model. This includes the BESS SOC limits, the maximal battery current, the maximal gas turbine input rate of change, and the maximal output power generation from the gas turbine and BESS. These can be formulated as follows

$$0 \leq \mathbf{SOC} \leq 1 \tag{3.45}$$

$$\mathbf{\Delta u_{1,min}} \leq \mathbf{\Delta u_1} \leq \mathbf{\Delta u_{1,max}} \tag{3.46}$$

$$\mathbf{\Delta u_{2,min}} \leq \mathbf{\Delta u_2} \leq \mathbf{\Delta u_{2,max}} \tag{3.47}$$

$$\mathbf{u_{1,min}} \leq \mathbf{u_1} \leq \mathbf{u_{1,max}} \tag{3.48}$$

$$\mathbf{u_{2,min}} \leq \mathbf{u_2} \leq \mathbf{u_{2,max}} \tag{3.49}$$

where the values of $\mathbf{\Delta u_{1,min}}$, $\mathbf{\Delta u_{1,max}}$, $\mathbf{\Delta u_{2,min}}$, $\mathbf{\Delta u_{2,max}}$, $\mathbf{u_{1,min}}$, $\mathbf{u_{1,max}}$, $\mathbf{u_{2,min}}$, and $\mathbf{u_{2,max}}$ are all given in table 3.1.
In addition to these hard constraints, two constraints which must be considered as soft constraints are introduced. One for the maximal specified frequency deviation and one for the maximal specified SOC deviation

$$-\mathbf{x_{4,max}} \leq \mathbf{x_4} \leq \mathbf{x_{4,max}} \tag{3.50}$$

$$-\mathbf{SOC_{max}} \leq \mathbf{SOC} \leq \mathbf{SOC_{max}} \tag{3.51}$$

where the values of $\mathbf{x_{4,min}}$, $\mathbf{x_{4,max}}$, $\mathbf{SOC_{min}}$, and $\mathbf{SOC_{min}}$ are also given in table 3.1. Note that the constraint in equation (3.51) overwrites the constraint in equation (3.45). These soft constraints are made hard by implementing them in this way because they restrict the feasible region and a solver will not return a solution outside of the feasible region. This is a problem because the controller will brake down in the case of a massive power cut which drives the frequency way outside the frequency constraint. However, the reason for implementing the constraints in this way is to highlight the SMPC ability to respect constraints under the influence of uncertainty.

3.2.5 Choice of solver

In the project leading to this thesis, the MATLAB™solver *fmincon* was used. This solver can handle both linear and non-linear problem types. As discussed in the project, *fmincon* was used because of its simplicity in implementing constraints by formulating them as functions. However for this thesis, the problem size increased to the degree that the general solver *fmincon* was no longer efficient enough and the change to a convex QP specific solver was made. The choice fell on SDPT3 after considering both Mosek and *Quadprog*.
For simplicity in implementation, CVX was used as the interface between MATLAB™and SDTP3. CVX is a modeling language[31] which means that one can implement the optimization problem in a user friendly and intuitive way in MATLAB™. CVX inputs the optimization problem into the solver which needs to be a specific structure. The disadvantage with this approach is that this added feature also adds time to the solving of the optimization problem. In a real-world application, this might be costly if time is a limiting factor. However, for research purposes such as this thesis it was considered to be convenient.
Another benefit from the CVX modeling language is that it follows the DCP rule set. This is a strict rule set on the problem formulation which assures convexity[31]. Inputting a non-convex problem to CVX will yield an error message and therefore using CVX gives a guarantee on the convexity of the problem.

3.3 Stochastic MPC implementation

The stochastic MPC in this thesis was implemented using scenario optimization. As discussed in section 2.6 under the paragraph **scenario optimization**, the distinction between collecting and generating scenarios is important because the latter one introduces assumptions about the uncertainty probability distribution. For our case, the wind speed generator described in section 3.1.3 can also be used to collect different scenarios and hence, no assumptions on the disturbance uncertainty is made.
For our case of frequency control with a time step of $\Delta t = 1.5s$, online scenario sampling was considered to be infeasible. Therefore, the scenarios were generated a priori and not updated between time steps. Since the scenario generation is a random sampling process in our case, the online and offline scenario sampling is practically the same because no past information is used when generating scenarios.

3.3.1 Cost function

As presented in section 2.6.1 and equation (2.28) the cost function from DMPC is replaced with the expected value of the cost function. For the SMPC, this is usually done by taking the average of the scenarios

$$E[J(u,w)] = \frac{1}{N}\sum_{i=1}^{N} J(u_i, w_i) \tag{3.52}$$

where N is the number of scenarios, u_i, and w_i is the optimal input and the realization of the disturbance for scenario i. It is well known that the expected value operator $E[-]$ preserves convexity and hence the optimization problem is still convex. Proof for this can be found in [33] starting at p.77.

3.3.2 Constraints

As explained in section 2.6.1, for every collected scenario a set of constraints is added to the optimization problem. This gives the following constraints

$$\mathbf{\Delta u_{1,min}} \leq \mathbf{\Delta u_{1,i}} \leq \mathbf{\Delta u_{1,max}}, \quad i \in \{1,2,...,N\} \tag{3.53}$$

$$\mathbf{\Delta u_{2,min}} \leq \mathbf{\Delta u_{2,i}} \leq \mathbf{\Delta u_{2,max}}, \quad i \in \{1,2,...,N\} \tag{3.54}$$

$$\mathbf{u_{1,min}} \leq \mathbf{u_{1,i}} \leq \mathbf{u_{1,max}}, \quad i \in \{1,2,...,N\} \tag{3.55}$$

$$\mathbf{u_{2,min}} \leq \mathbf{u_{2,i}} \leq \mathbf{u_{2,max}}, \quad i \in \{1,2,...,N\} \tag{3.56}$$

$$\mathbf{x_{4,min}} \leq \mathbf{x_{4,i}} \leq \mathbf{x_{4,max}}, \quad i \in \{1,2,...,N\} \tag{3.57}$$

$$\mathbf{SOC_{min}} \leq \mathbf{SOC_{i}} \leq \mathbf{SOC_{max}}, \quad i \in \{1,2,...,N\} \tag{3.58}$$

A final remark to be made on the implementation of the constraints and the cost function for this approach with N terms in the cost function and N set of constraints, is to avoid FOR loops when building the problem for your selected solver. This can be a tempting solution as it is easy to implement and intuitive to understand, but it will drastically increase computational time non linearly with N. The solution to this problem is to build matrices, with one column for each scenario and use matrix multiplication to achieve the desired terms in the constraints and cost function.

3.3.3 Control action parameterization

Affine disturbance feedback All the different control parameterizations presented in this thesis are also discussed in [24]. The affine disturbance feedback parameterization have the following structure

$$u_i = \gamma_i + \sum_{j}^{i-1} \theta_{i,j} w_{0+j} \tag{3.59}$$

reorganizing this into recursive eliminated vector form and integrating the number of states, control actions, and disturbances yields

$$\mathbf{U} = \Gamma + \Theta \mathbf{W} \tag{3.60}$$

where

$$\begin{aligned} \mathbf{U}^T &= [\mathbf{u_1}, \mathbf{u_2}, \dots, \mathbf{u_{N_c}}] \\ \mathbf{W}^T &= [\mathbf{w_1}, \mathbf{w_2}, \dots, \mathbf{w_{N_c}}] \\ \Gamma^T &= [\gamma_\mathbf{1}, \gamma_\mathbf{2}, \dots, \gamma_{\mathbf{N_c}}] \\ \Theta &= \begin{bmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \theta_{\mathbf{1},\mathbf{0}} & \mathbf{0} & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \theta_{\mathbf{N_c-1},\mathbf{0}} & \dots & \theta_{\mathbf{N_c-1},\mathbf{N_p-2}} & \mathbf{0} \end{bmatrix} \\ {\gamma_\mathbf{i}}^T &= [\gamma_{i,1}, \gamma_{i,2}] \\ {\theta_\mathbf{i,j}}^T &= [\theta_{i,j,1}, \theta_{i,j,2}] \end{aligned} \tag{3.61}$$

The total number of variables is then

$$d = N_c B_y + B_y H_y \frac{(N_c - 1)N_c}{2} = 4 \cdot 2 + 2 \cdot 2 \frac{(4-1)4}{2} = 32 \tag{3.62}$$

This control action parameterization is referred to as the full parameterization (FP).

State feedback The state feedback control parameterization have the following structure

$$u_i = \gamma_i - K x_i \tag{3.63}$$

where K is fixed and often implemented as the LQR feedback gain of the infinite unconstrained optimization problem. This structure has been found to improve performance over the non-parametrisized control action in [26] and it is therefore interesting to observe the performance of this parameterization for our case of the isolated power system. Note that K is fixed and not an optimization variable. This is due to a non-linearity which arises when performing the recursive elimination of the states, because the future states x are dependent on the optimization variables. Solutions to this problem exist, see for example [27] where the authors reformulate the problem as a LP problem by exploiting geometrical properties of K and solving a disturbance feedback problem. However, this was considered to be out of the scope of this thesis.
As mentioned above, K is implemented as the solution to the LQR problem

$$\begin{aligned} \min J = &\int_0^\infty x^T Q x + u^T R u 2 x^T N u, \quad \text{subject to} \\ &u = -Kx \\ &\dot{x} = Ax + Bu \end{aligned} \tag{3.64}$$

which is found by the applying the MATLAB™function $lqr(A, B, Q, R)$ which solves the Ricatti equation with respect to S

$$A^T S + SA - (SB + N)R^{-1}(B^T S + N^T) + Q = 0 \tag{3.65}$$

and calculates

$$K = R^{-1}(B^T S + N^T) \tag{3.66}$$

The weighting matrices Q and R are identical with the ones used in the resulting MPC controller. The total number of variables is then

$$d = N_c B_y = 4 \cdot 2 = 8 \tag{3.67}$$

No parameterization The last control action configuration investigated in this thesis is the non-parametrisized one. This is simply be expressed as

$$u_i = \gamma_i \tag{3.68}$$

which gives the same number of variables as for the state feedback parameterization.

3.3.4 Choosing the number of scenarios

Choosing the number of scenarios N will in most cases be determined by what "guaranteed" percentage of constraint satisfaction is desired. After determining the constraint violation probability ϵ, and the degree of certainty β, the number of scenarios can be calculated from equation (2.32) by solving it using the bisection method or graphically. The graphical solution for the three different control action parameterizations in this thesis is shown below in figure 3.15

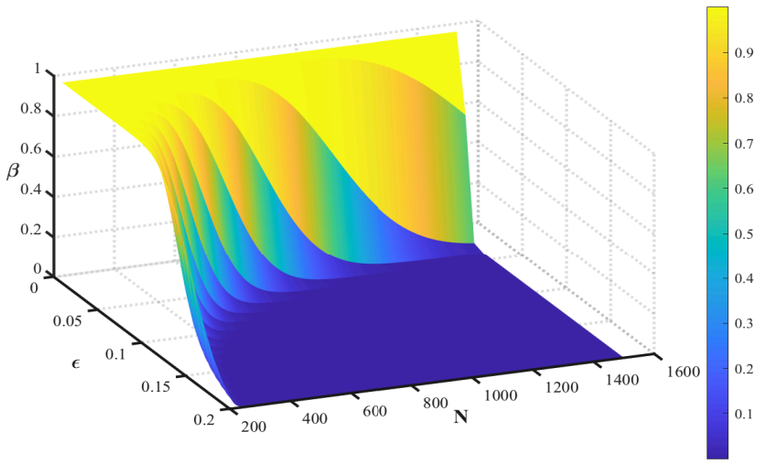
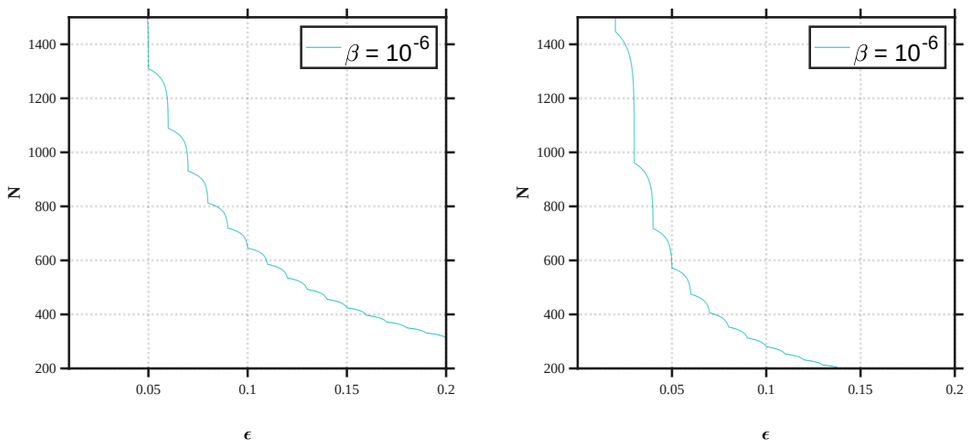


Figure 3.14: 3D plot of equation (2.32), $\beta(\epsilon, N) = 10^{-6}$



(a) $d = 32$ (b) $d = 8$

Figure 3.15: Contour plots of $\beta(\epsilon, N) = 10^{-6}$

3.4 $H\infty$ implementation

For this thesis, the mixed sensitivity $H\infty$ control strategy is used as a benchmark robust control algorithm. The overall approach is described in section 2.5.2 and in short it consists of choosing the appropriate weights W_p, W_u, and W_d and then solve the LMI in equation (2.23). The latter is done by using the MATLAB™function $[K] = mixsyn(G, W_p, W_u, W_d)$. This section is focused on the choice of the weights.
From equation (2.23) and (2.24) we can obtain the following expressions by rewriting the equations

$$\begin{aligned} |S(j\omega)| &\leq |W_p(j\omega)^{-1}| \\ |KS(j\omega)| &\leq |W_u(j\omega)^{-1}| \\ |T(j\omega)| &\leq |W_d(j\omega)^{-1}| \end{aligned} \tag{3.69}$$

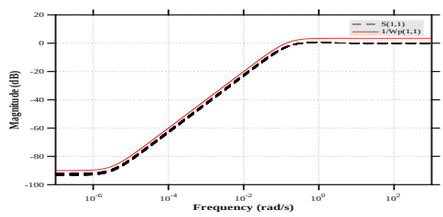
Hence, we can design the weights W_p, W_u, and W_d to limit the sensitivity function, control action, and complementary sensitivity function respectively. The general rules of thumb when tuning the weights are listed in section 2.5.2 and was used for the tuning for this application. By an iterative process of trial and error which consisted of choosing weights, and checking the continuous model performance, the following weights were obtained

$$W_p = \begin{bmatrix} \frac{\frac{2}{3}s+0.01}{s+3.15*10^{-6}} & \frac{\frac{2}{3}s+0.01}{s+3.15*10^{-6}} \\ \frac{\frac{2}{3}s+0.50}{s+4.99*10^{-6}} & \frac{0.32s+0.05}{s+4.74*10^{-6}} \end{bmatrix} \tag{3.70}$$

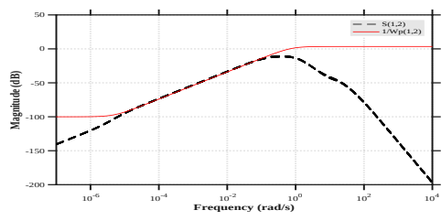
$$W_u = [-] \tag{3.71}$$

$$W_d = \begin{bmatrix} \frac{0.32s+0.018}{s+1.77} & \frac{10s+3.33}{s+33.34} \\ \frac{s+1.00*10^{-8}}{s+100} & \frac{s+3.17}{s+31.65} \end{bmatrix} \tag{3.72}$$

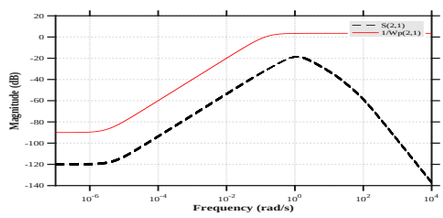
In figure 3.16 and 3.17 below, the criteria in (3.69) is visualized for 30 different realizations of the plant with 10% parametric uncertainty. As one can see, the black dotted lines which represent the different realizations of the plant uncertainty are never above the inverse of the weights with one exception. The only exception is for the complementary sensitivity function at frequencies below 10^{-10}rad/s which is not realistic frequencies of a disturbance or noise signal and hence not relevant.
An important note is that the bandwidth of the controller had to be adjusted when implementing the discrete controller. Because of sampling issues with $\Delta t = 1.5s$ the bandwidth was reduced significantly for the discrete implementation in order to obtain a stable controller. Since the MPC controller is implemented with $\Delta t = 1.5s$ it was considered important to have the same time step for the $H\infty$ controller to obtain a good comparison. For a real-world application, the time step would probably have been reduced to increase the bandwidth of the controller and achieve better reference tracking.



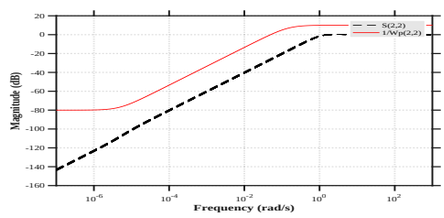
(a) input 1, to output 1



(b) input 1, to output 2

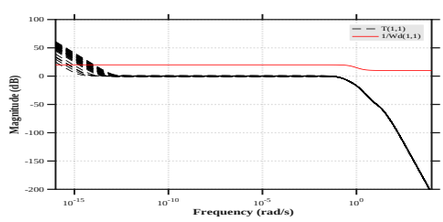


(c) input 2, to output 1

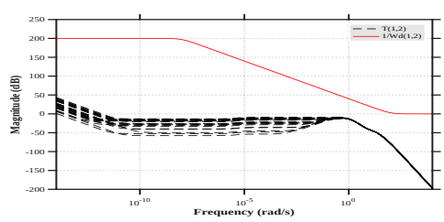


(d) input 2, to output 2

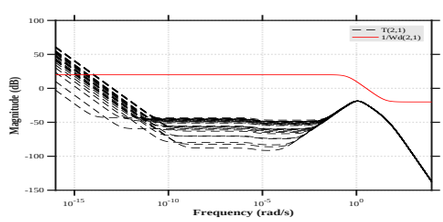
Figure 3.16: Bode plot for $1/W_p$ and the sensitivity function S



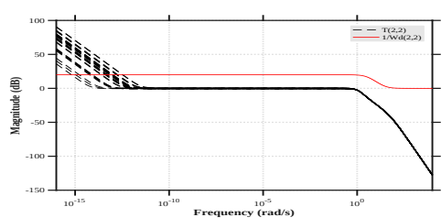
(a) input 1, to output 1



(b) input 1, to output 2



(c) input 2, to output 1



(d) input 2, to output 2

Figure 3.17: Bode plot for $1/W_d$ and the complementary sensitivity function T

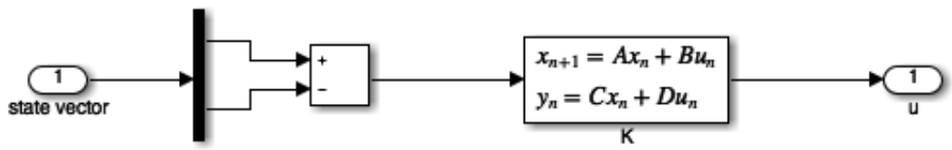


Figure 3.18: $H\infty$ controller subsystem in Simulink™

The $H\infty$ controller was implemented in Simulink™using a discrete state space block as can be seen from figure 3.18.

3.5 Weather forecast

Because the wind generation is a result of the weather conditions, a necessary tool for predicting the future states of the model is the weather forecast. If there is an offset between the predicted average wind power generation and the actual average wind power generation this would lead to an increasing frequency deviation. For a control problem where the time scale is hours or days, developing accurate forecasts is feasible but for our case of frequency control where the time scale is seconds it is not feasible. For such short horizons, the weather conditions seems to behave as a normal distributed stochastic variable and can be modeled thereafter. The wind speed generator described in section 3.1.3 reflects this.

3.6 Performance metrics

To be able to demonstrate the strengths and weaknesses of different control algorithms more intuitively, some performance metrics were introduced. These were designed to reflect real-world performance metrics.
To measure the RES penetration, the total units of fuel is calculated. As mentioned earlier in section 3.2.2, the variation in gas turbine power generation has a large impact on the fuel consumption. For this reason, we penalize the increased fuel consumption from the gas turbine generation variation by penalizing $\mathbf{\Delta u_1}$. Hence it makes sense to also define the fuel consumption as the variation in $\mathbf{u_1}$ and we get

$$\text{Total units fuel} = \sum_{k=1}^{N_c} \Delta u_{1,k} \tag{3.73}$$

The second performance metric is defined as the average frequency deviation. This is a natural metric to include because the main objective is to preserve the frequency around its specified value.

$$\text{Average frequency deviation} = \frac{1}{N_p} \sum_{k=1}^{N_p} x_{4,k} \tag{3.74}$$

Thirdly, to measure the battery degradation two metrics are defined. The average SOC deviation and the SOC standard deviation.

$$\text{Average SOC deviation} = \frac{1}{N_p} \sum_{k=1}^{N_p} SOC_k - SOC_{ref,k} \tag{3.75}$$

$$\text{SOC standard deviation} = \sqrt{\frac{1}{N_p} \sum_{k=1}^{N_p} \left(SOC_k - \frac{1}{N_p} \sum_{k=1}^{N_p} SOC_k \right)^2} \tag{3.76}$$

3.7 Simulations

The three questions from section 1 were addressed by performing a Monte Carlo simulation with 100 samples, for the step change in load response of our closed loop controller and plant model. This was regarded as a realistic scenario because loads can be connected and disconnected from the grid instantaneously. We then observed how the frequency, SOC, and control action usage developed over the course of a 60 seconds time horizon. This simulation was performed for the nominal model with no parametric uncertainty, for 10% parametric uncertainty, and for 50% uncertainty.

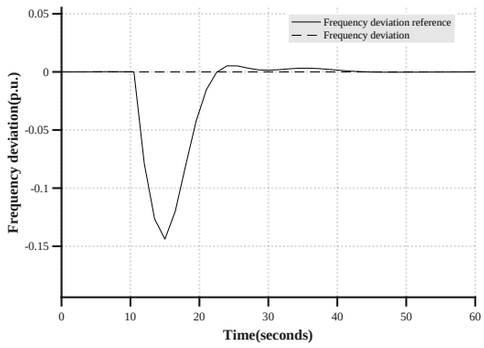
Parameter	Symbol	Value	Units
Nominal governor time constant	$T_{g,0}$	0.05	[s]
Nominal gas turbine time constant	$T_{t,0}$	0.5	[s]
Nominal wind turbine generator time constant	$T_{WTG,0}$	0.04	[s]
Nominal battery time constant	$T_{BESS,0}$	0.1	[s]
Nominal MG load damping coefficient	D_0	1	[p.u./Hz]
Nominal MG inertia constant	M_0	3	[p.u. s/Hz]
Control horizon	N_c	4	[-]
Prediction horizon	N_p	10	[-]
Discrete time step	Δt	1.5	[s]
Simulation length	n_{sim}	60	[s]
Nominal battery capacity	$C_{bat,0}$	1	[p.u.]
KiBaM charging well width	$c_{W,0}$	0.93	[-]
KiBaM charging well conductance	$c_{r,0}$	$2.24{\cdot}10^{-5}$	[-]
Wind turbine cut inn speed	V_{ci}	13	[m/s]
Wind turbine cut off speed	V_{co}	17	[m/s]
Wind turbine rated wind speed	V_r	15.5	[m/s]
Wind turbine rated power	P_r	0.2	[p.u.]
Maximal gas turbine input	$\mathbf{u_{1,max}}$	1	[p.u.]
Minimal gas turbine input	$\mathbf{u_{1,min}}$	-1	[p.u.]
Maximal BESS input	$\mathbf{u_{2,max}}$	1	[p.u.]
Minimal BESS input	$\mathbf{u_{2,min}}$	-1	[p.u.]
Maximal gas turbine input change	$\mathbf{\Delta u_{1,max}}$	0.1	[p.u.]
Minimal gas turbine input change	$\mathbf{\Delta u_{1,min}}$	-0.1	[p.u.]
Maximal BESS output power	$\mathbf{\Delta u_{2,max}}$	0.2	[p.u.]
Minimal BESS output power	$\mathbf{\Delta u_{2,min}}$	-0.2	[p.u.]
Maximal frequency deviation	$\mathbf{x_{4,max}}$	0.2	[p.u.]
Maximal SOC deviation	$\mathbf{SOC_{4,max}}$	0.3	[p.u.]

Table 3.1: Table of Constants

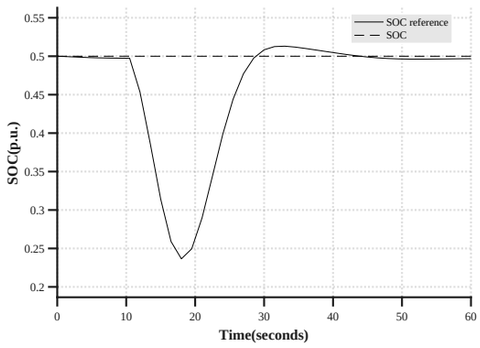
4 Results

4.1 Nominal behaviour

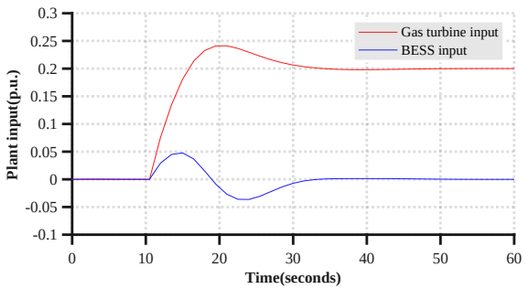
The nominal behaviour, which is the behaviour with no parametric or disturbance uncertainty is included in this section to demonstrate the tuning of the different controllers. The load condition is a constant signal which performs a step increase at $t = 10s$. This can simulate connecting a large load to the grid such as a drilling machine for the offshore isolated power system.



(a) Frequency deviation

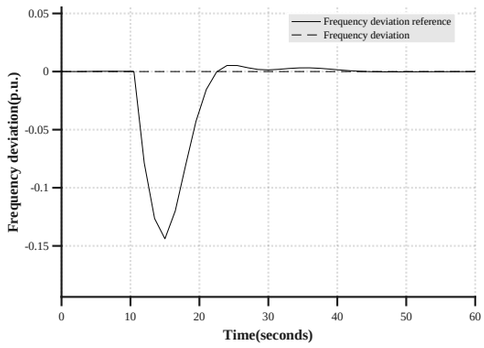


(b) SOC deviation

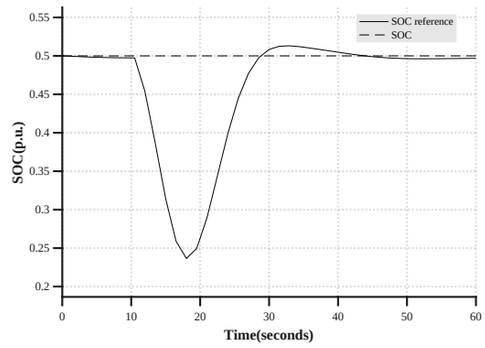


(c) Gas turbine and BESS input

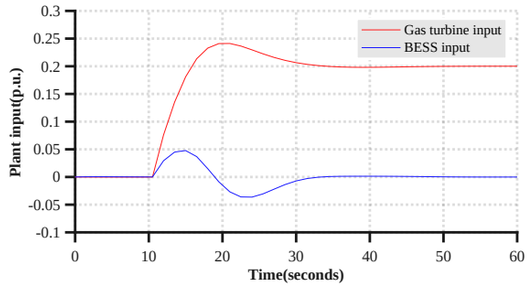
Figure 4.1: Nominal behaviour for DMPC



(a) Frequency deviation

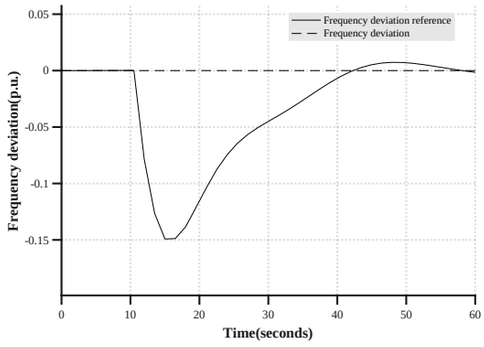


(b) SOC deviation

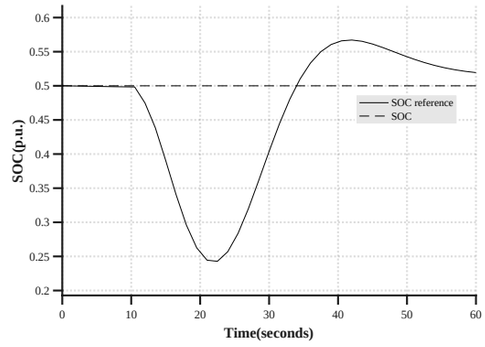


(c) Gas turbine and BESS input

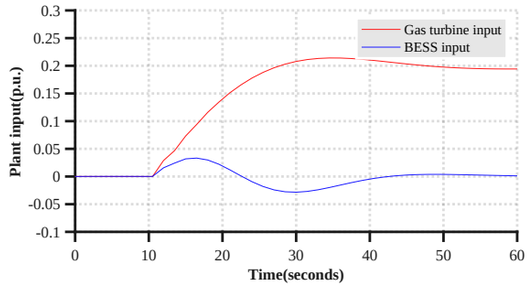
Figure 4.2: Nominal behaviour for SMPC FP



(a) Frequency deviation

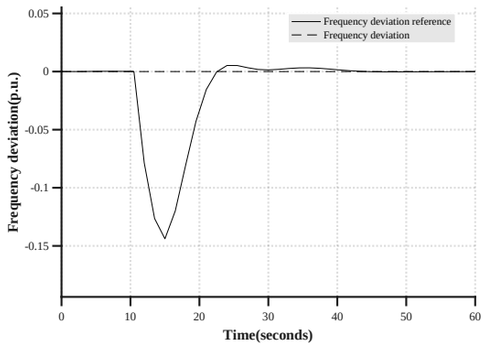


(b) SOC deviation

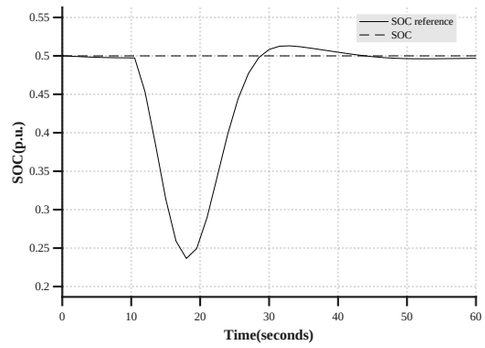


(c) Gas turbine and BESS input

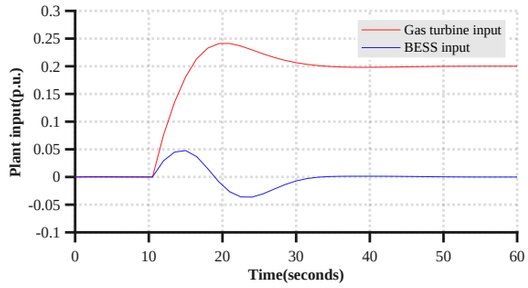
Figure 4.3: Nominal behaviour for SMPC SF



(a) Frequency deviation

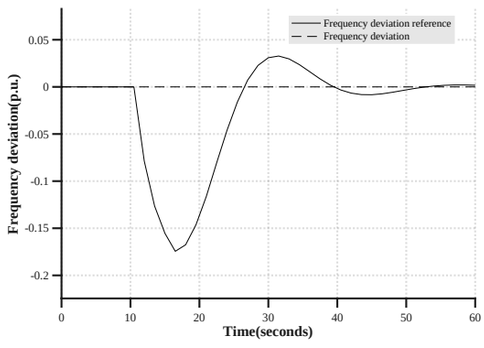


(b) SOC deviation

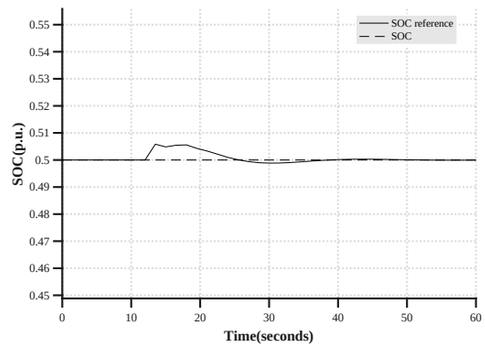


(c) Gas turbine and BESS input

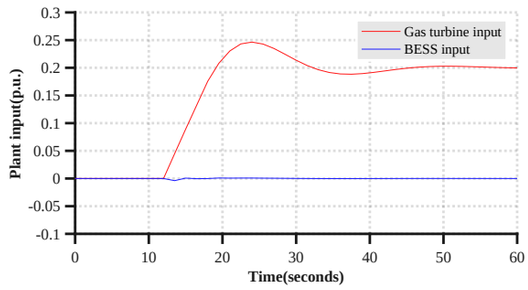
Figure 4.4: Nominal behaviour for SMPC NP



(a) Frequency deviation



(b) SOC deviation



(c) Gas turbine and BESS input

Figure 4.5: Nominal behaviour for $H\infty$

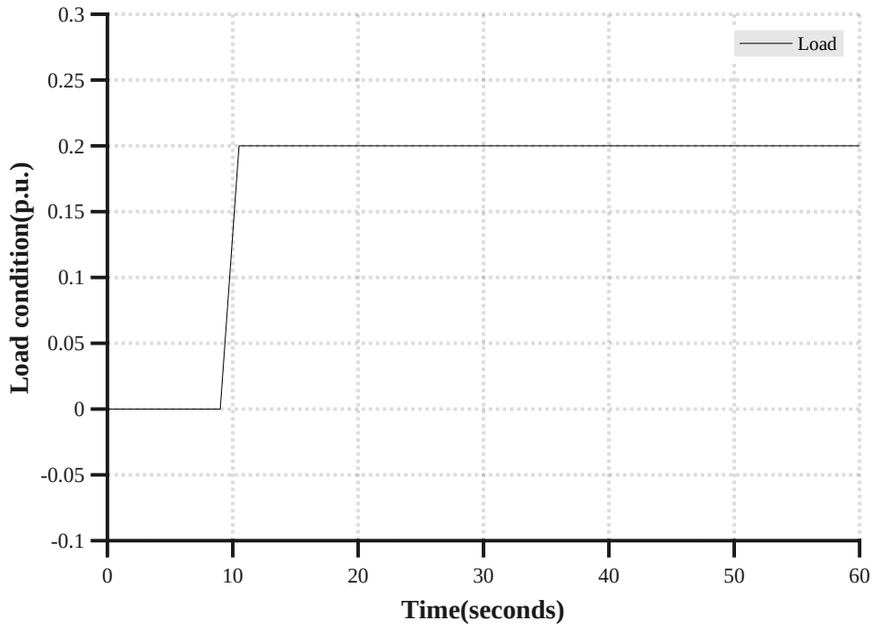


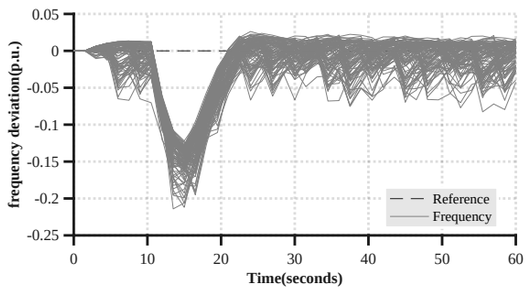
Figure 4.6: Load profile

	DMPC	SMPC FP	SMPC SF	SMPC NP	H∞
Average frequency deviation	0.0157	0.0157	0.0354	0.0157	0.0326
Average SOC deviation	0.0433	0.0433	0.0737	0.0433	$9.89 * 10^{-4}$
Units fuel	0.2001	0.2001	0.1943	0.2001	0.1997
SOC standard deviation	0.0805	0.0805	0.1029	0.0805	$1.90 * 10^{-3}$

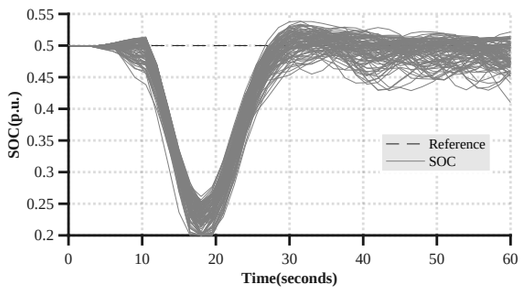
Table 4.1: Performance metrics with no parametric uncertainty, and constant nominal wind power generation

The controllers are tuned both to give optimal performance in terms of the balance between reference tracking and control effort and to give comparable results. As can be seen from figure 4.5 the $H\infty$ controller has a different behaviour in terms of the BESS handling and SOC compared to the MPC controllers. This is discussed further in section 5.1, but the results reveal that the BESS is actually slowing down the frequency response by charging in stead of discharging.
Another key observation to make is that the controllers DMPC, SMPC FP and SMPC NP has the exact same behaviour. Because there are no parametric or disturbance uncertainty, and hence only one scenario, the SMPC controllers are effectively DMPC controllers. Since their weights are equal, so are their behaviours.

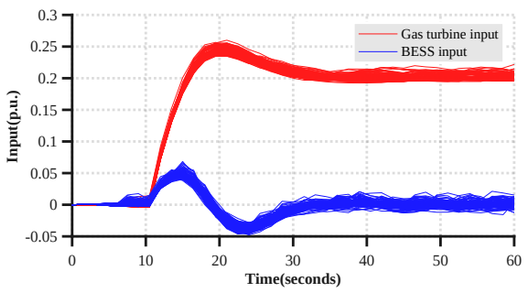
4.2 10% Parametric uncertainty and disturbance uncertainty



(a) Frequency deviation

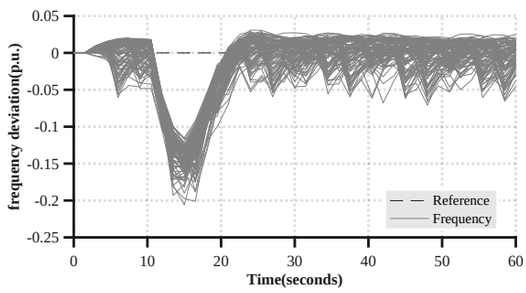


(b) SOC deviation

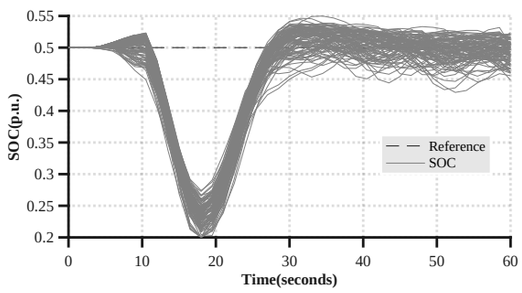


(c) Gas turbine and BESS input

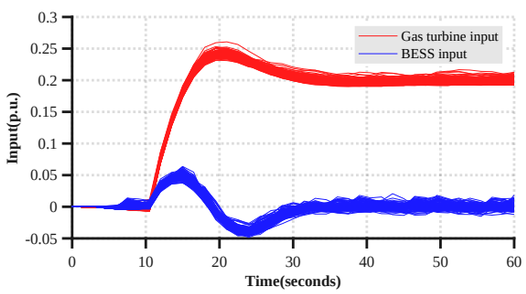
Figure 4.7: 10% parametric uncertainty and uncertain wind power generation behaviour for DMPC



(a) Frequency deviation

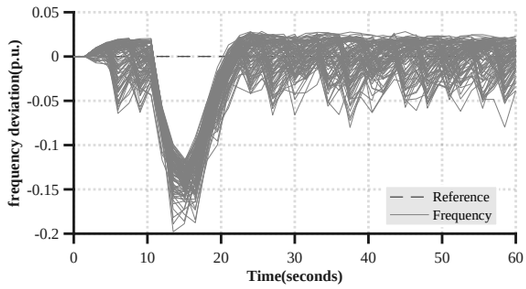


(b) SOC deviation

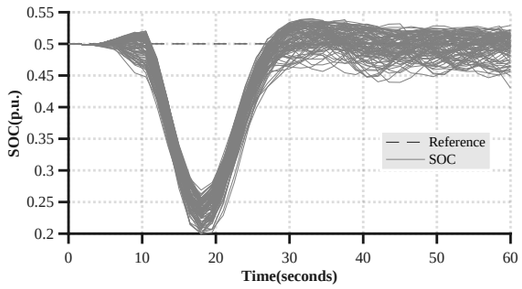


(c) Gas turbine and BESS input

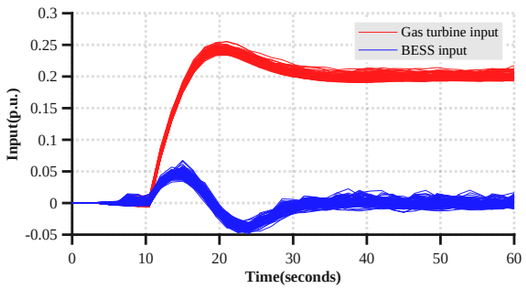
Figure 4.8: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 250 scenarios



(a) Frequency deviation

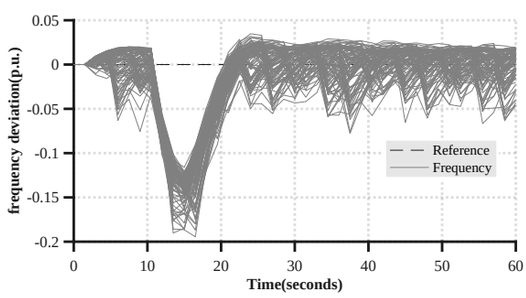


(b) SOC deviation

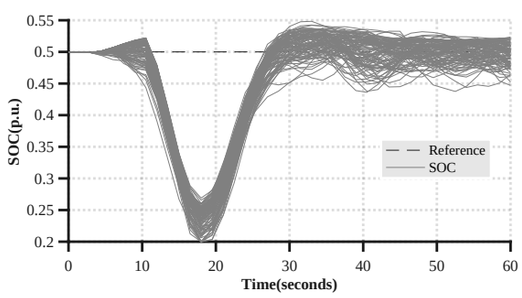


(c) Gas turbine and BESS input

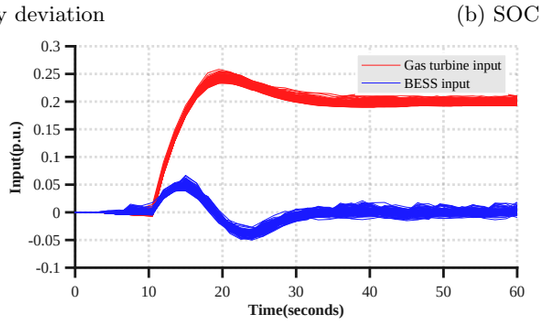
Figure 4.9: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 500 scenarios



(a) Frequency deviation

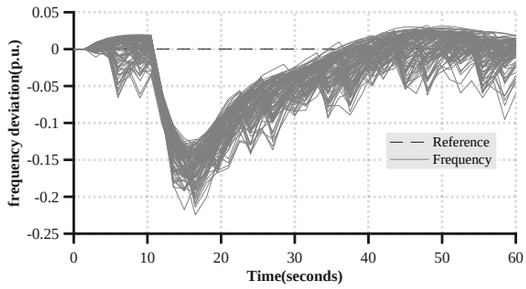


(b) SOC deviation

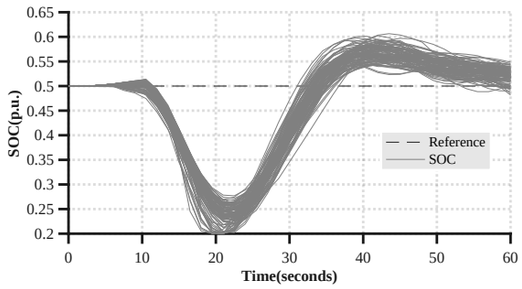


(c) Gas turbine and BESS input

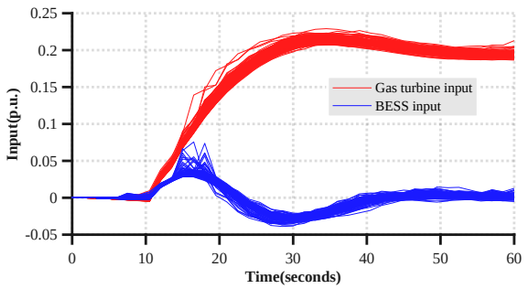
Figure 4.10: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 750 scenarios



(a) Frequency deviation

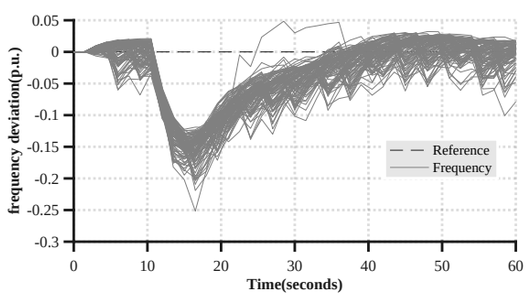


(b) SOC deviation

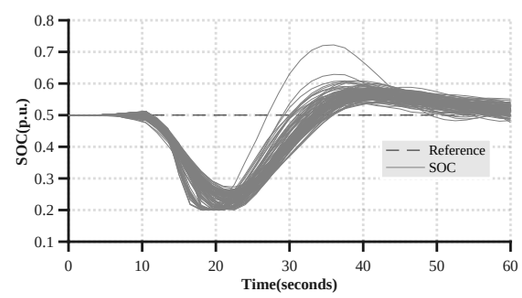


(c) Gas turbine and BESS input

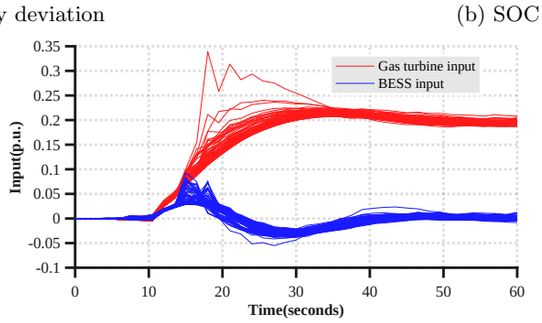
Figure 4.11: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 250 scenarios



(a) Frequency deviation

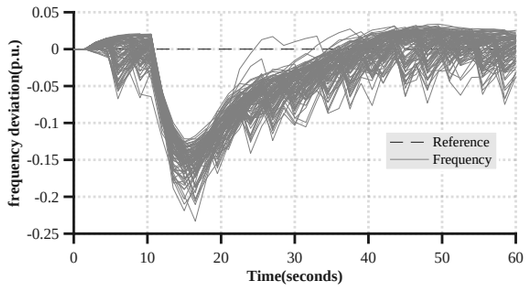


(b) SOC deviation

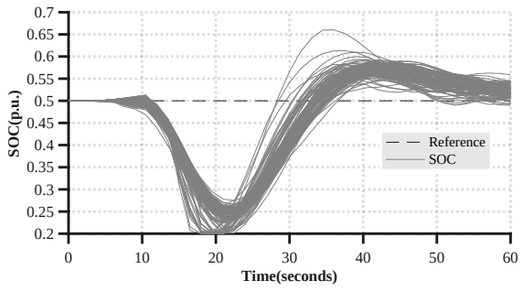


(c) Gas turbine and BESS input

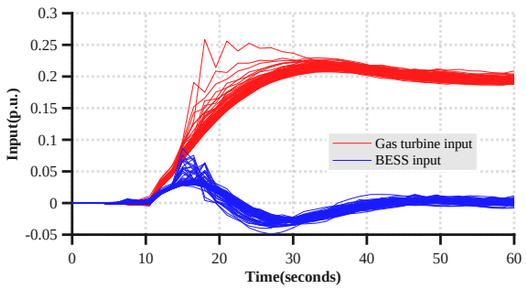
Figure 4.12: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 500 scenarios



(a) Frequency deviation

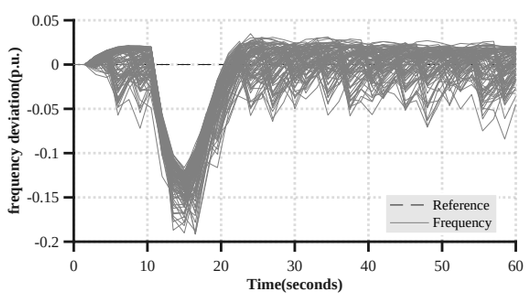


(b) SOC deviation

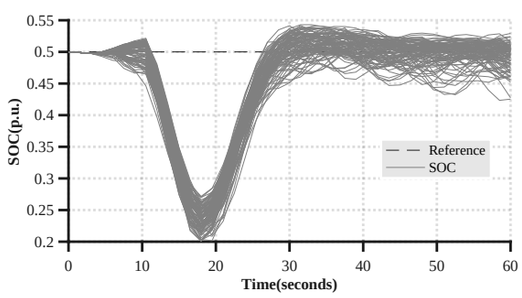


(c) Gas turbine and BESS input

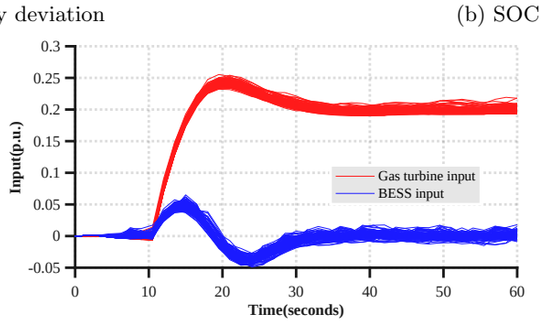
Figure 4.13: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 750 scenarios



(a) Frequency deviation

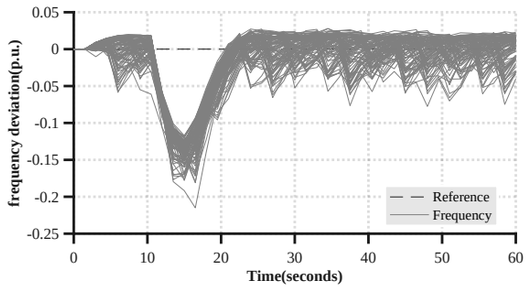


(b) SOC deviation

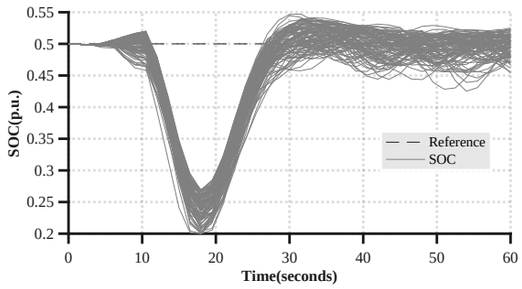


(c) Gas turbine and BESS input

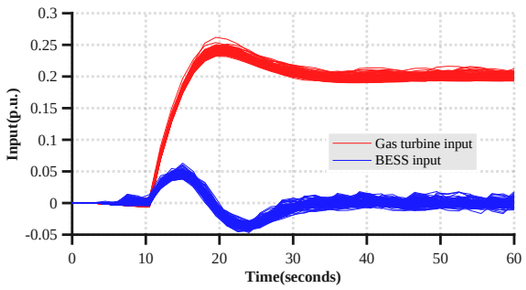
Figure 4.14: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 250 scenarios



(a) Frequency deviation

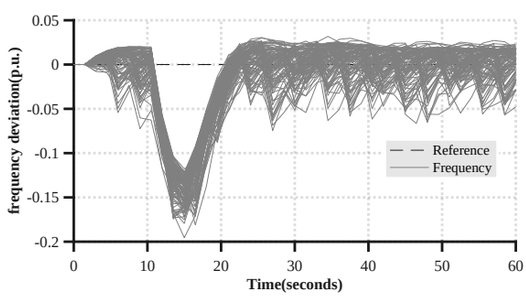


(b) SOC deviation

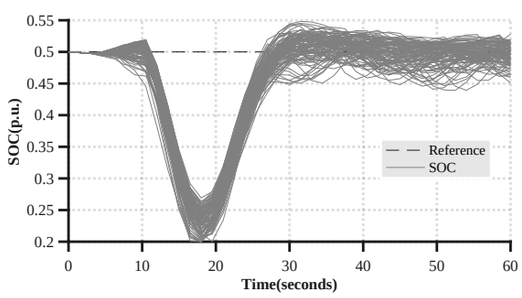


(c) Gas turbine and BESS input

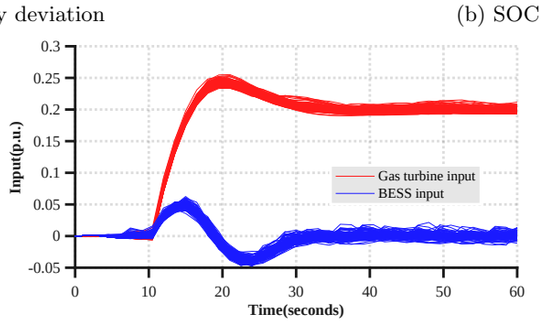
Figure 4.15: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 500 scenarios



(a) Frequency deviation

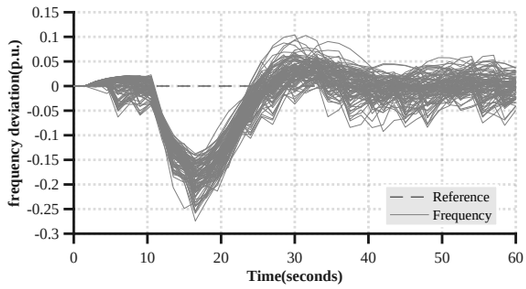


(b) SOC deviation

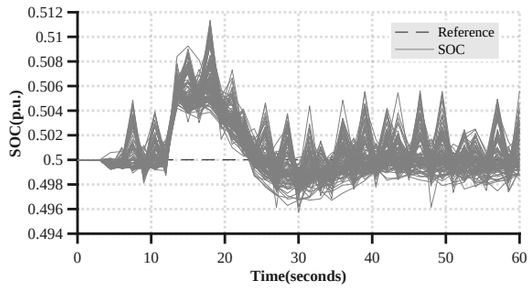


(c) Gas turbine and BESS input

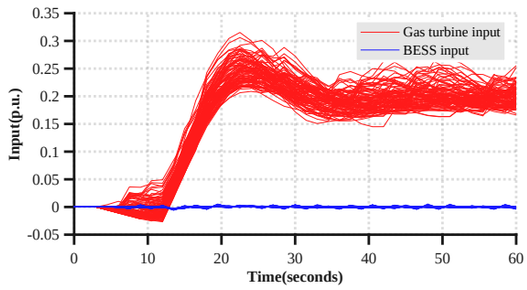
Figure 4.16: 10% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 750 scenarios



(a) Frequency deviation



(b) SOC deviation

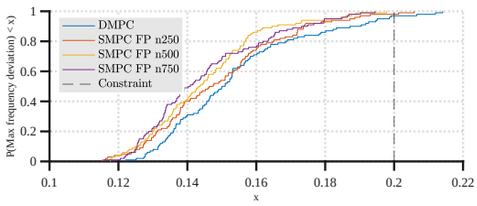


(c) Gas turbine and BESS input

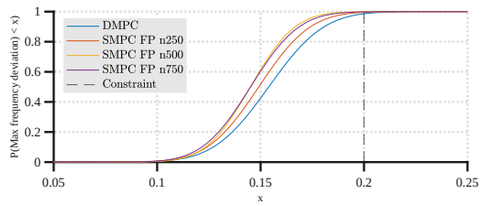
Figure 4.17: 10% parametric uncertainty and uncertain wind power generation behaviour for $H\infty$ controller

4.2.1 Constraint violation

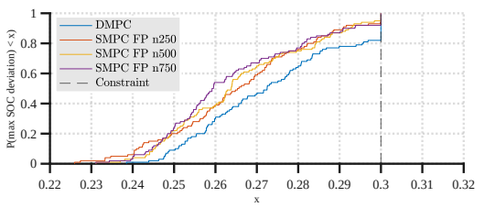
To simulate scenarios where constraint violation is a probable scenario is perhaps the most interesting comparison between SMPC controllers, DMPC controller and the $H\infty$ controller. Because the scenario theory for SMPC presented in section 2.6.1 "guarantees" a maximal theoretical violation probability, it is interesting to compare the theoretical value to the actual violation probability for our case. As discussed earlier, in [24] they found that the actual violation probability was significantly lower than the theoretical value.
For this section, both the empirical and the fitted normal distribution curve of the cumulative probability density plot are included. The empirical is included to provide the actual number of constraint violations and to capture the possible deviation from the normal distribution around the constraints. The fitted curves are included for a smooth representation of the CDFs for illustration purposes. The DMPC controller is used in the plots as a benchmark to compare against and is therefore included in all plots.



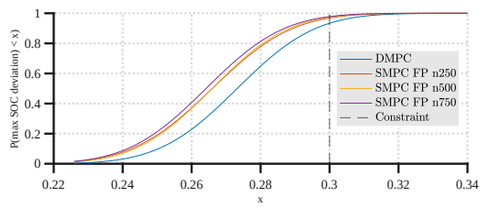
(a) Frequency violation, empirical curve



(b) Frequency violation, fitted normal distributed curve



(c) SOC violation, empirical curve

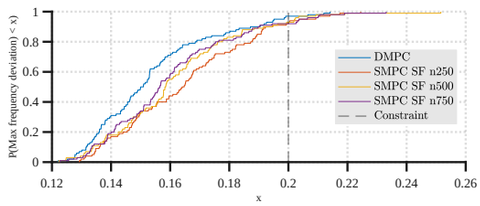


(d) SOC violation, fitted normal distributed curve

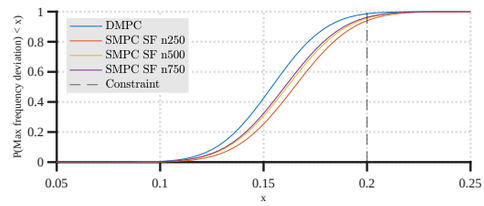
Figure 4.18: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC FP with increasing scenario collection

SMPC full parameterization The first observation one can make from figure 4.18 is that SMPC FP is more left shifted than DMPC. This indicates that SMPC FP is a more conservative controller with better margins in terms of constraint violation than DMPC. Note also that

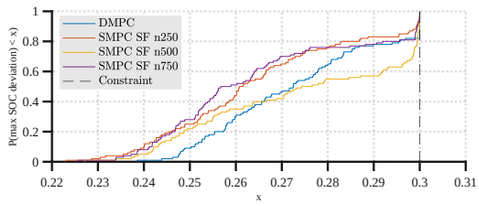
increasing the number of scenarios shifts the curve towards the left and the controller becomes more conservative. This is in line with the theory presented in [24].
Another interesting observation is the fact that there are no SOC violations. Instead, there are many cases where the maximal SOC deviation constraint is active. This indicates that the predictions of the controllers reflect the model accurately. Whether or not this is realistic is discussed later in section 5.2. Since there are more cases where the limit is reached for DMPC, it can be argued that it is more vulnerable to uncertainties not included in this model.



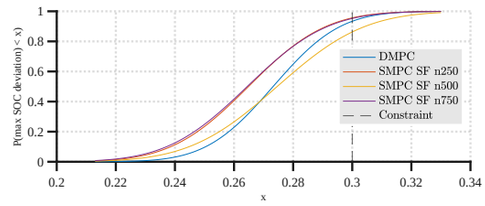
(a) Frequency violation, empirical curve



(b) Frequency violation, fitted normal distributed curve



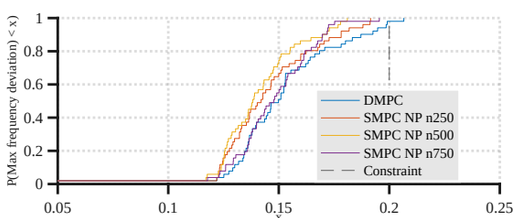
(c) SOC violation, empirical curve



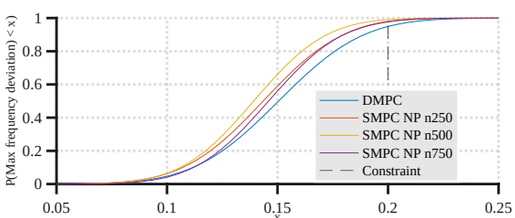
(d) SOC violation, fitted normal distributed curve

Figure 4.19: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC SF with increasing scenario collection

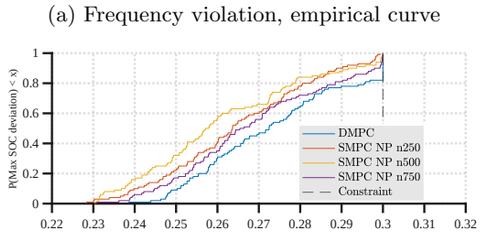
SMPC state feedback parameterization In figure 4.19 it can be seen that the state feedback controllers is actually right shifted compared to DMPC for frequency violation. There occurs 8 frequency violations for SMPC FP with 750 collected scenarios, as can bee seen from table 4.2. This is higher than the theoretical guarantee of 4% and a topic for discussion is section 5.2. For SOC violation no conclusions of right or left shift can be drawn. Another observation is that the number of collected scenarios does not seem to improve the constraint violation probability. Especially in figure 4.19c, the SMPC SF correlation between number of scenarios and left shift seems to be more or less random. This is in contradiction with previous research performed in [24] and [26] and is a topic for discussion later in section 5.2.
Lastly, one can observe that there are no SOC constraint violations for this parameterization either, as was the case for the SMPC FP.



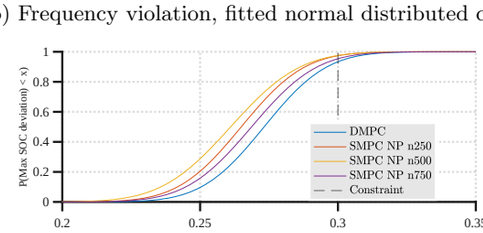
(a) Frequency violation, empirical curve



(b) Frequency violation, fitted normal distributed curve



(c) SOC violation, empirical curve

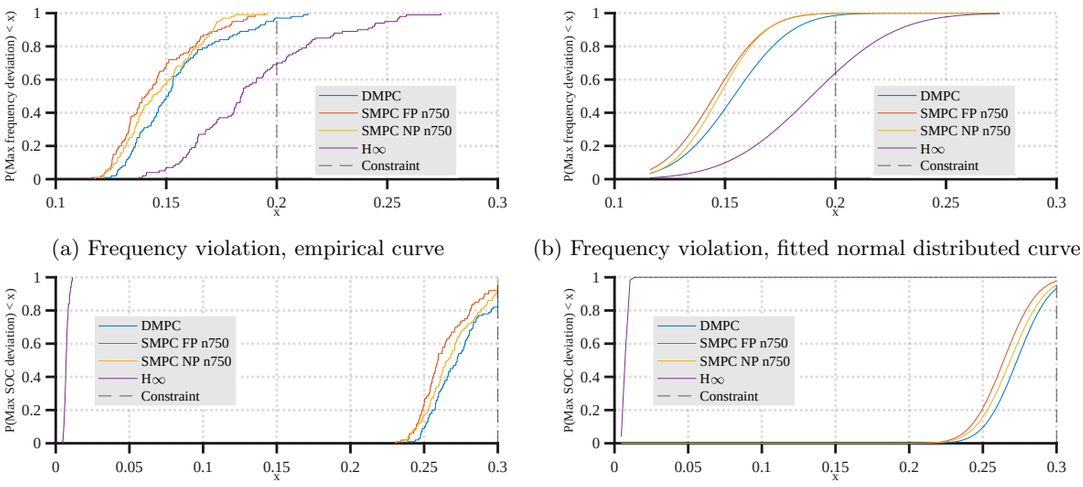


(d) SOC violation, fitted normal distributed curve

Figure 4.20: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC NP with increasing scenario collection

No parameterization From figure 4.20 one can observe that the SMPC NP controller is generally more left shifted than DMPC. This goes for both the frequency and SOC deviation

violation probability. Again, this tells us that the SMPC NP controller is more conservative than the DMPC controller and gives lower violation probability.
However in contradiction to the SMPC FP controller, the correlation between the number of scenarios and left shift of the curve is not clear. It is the curve for 500 collected scenarios that is the most left shifted for both the frequency deviation and SOC deviation plots. This is the same issue as mentioned for the state feedback parameterization and is discussed in section 5.2. As for the SMPC FP and SMPC SF, there are no SOC constraint violations.



(a) Frequency violation, empirical curve

(b) Frequency violation, fitted normal distributed curve

(c) SOC violation, empirical curve

(d) SOC violation, fitted normal distributed curve

Figure 4.21: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC FP N=750 vs SMPC NP N=750 vs $H\infty$

Mixed synthesis $H\infty$ controller Figure 4.21 clearly demonstrates the weakness of the $H\infty$ controller as a MIMO controller with multiple objectives. Since the BESS is barely activated, and actually charges the battery in the event of the load increase, the frequency is driven way below the constraint of 0.2 p.u. Even though $H\infty$ is a robust control strategy, it is not robust to such constraints as the imposed frequency and SOC deviation constraint. As explained in section 2.5.2 it only guarantees that the magnitude of the sensitivity function and complementary sensitivity function is below a certain value, which implies that the magnitude of the amplification of a reference signal or disturbance signal is below a certain value. Please note that tuning the $H\infty$ controller could possibly give better performance for frequency tracking. Great effort was spent on tuning the $H\infty$ controller in this thesis and the results presented here was the best obtained frequency tracking response. The difficulty in tuning the $H\infty$ is also discussed in section 5.1.

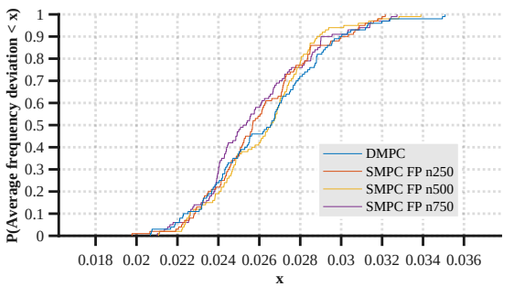
100 samples	Empirical		Theoretical
	Frequency	SOC	
DMPC	3%	0%	-
$H\infty$	31%	0%	-
SMPC FP N = 250	2%	0%	24.90%
SMPC FP N = 500	0%	0%	12.95%
SMPC FP N = 750	0%	0%	8.85%
SMPC SF N = 250	8%	0%	11.30%
SMPC SF N = 500	7%	2%	5.95%
SMPC SF N = 750	8%	0%	4.00%
SMPC NP N = 250	0%	0%	11.30%
SMPC NP N = 500	1%	0%	5.95%
SMPC NP N = 750	0%	0%	4.00%

Table 4.2: Violation probability for maximal frequency and SOC deviation constraints for DMPC vs SMPC FP with increasing scenario collection and 10% parametric uncertainty

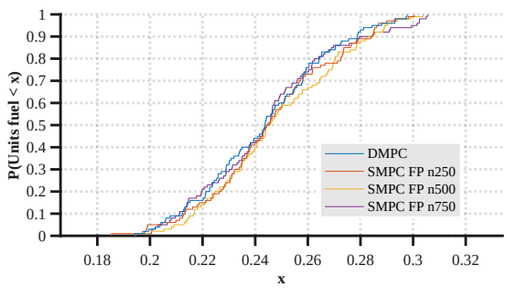
4.2.2 Performance

In this section, metrics which better highlights the average performance of the control strategies is presented. For many applications, the average performance is more important than the constraint violation.
For this section the empirical CDF curves are presented. A general remark to have in mind

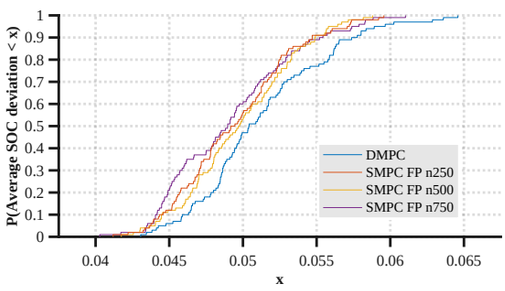
for this section is that the average performance for different metrics is influenced by the tuning of the controllers via the objective function weights of the MPC controllers and the pre and post multiplication weights of the $H\infty$ controller. For this reason the controllers are tuned to obtain the same nominal performance in regards to prioritization of different control objectives as discussed in section 4.1. The reason this is not so prominent for the constraint violation is obviously because the constraints are absolute and not weighted in the same way as the control objectives.



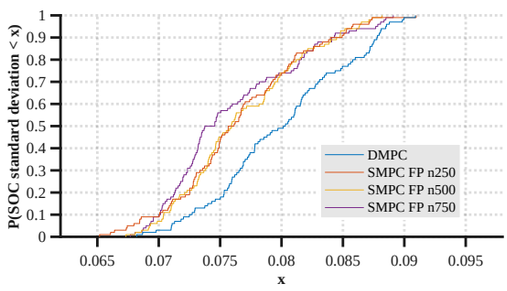
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



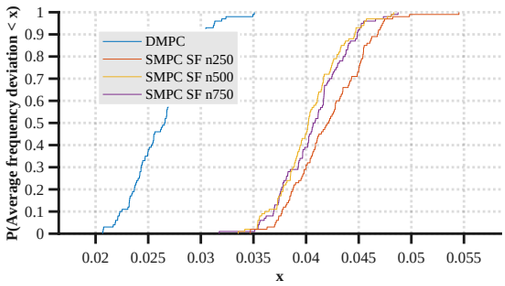
(c) Average SOC deviation, empirical curve



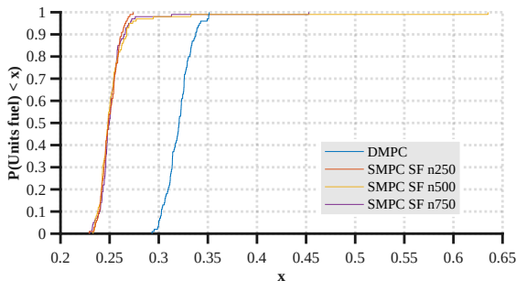
(d) SOC standard deviation, empirical curve

Figure 4.22: CDF for performance metrics, DMPC vs SMPC FP, 10% parametric uncertainty

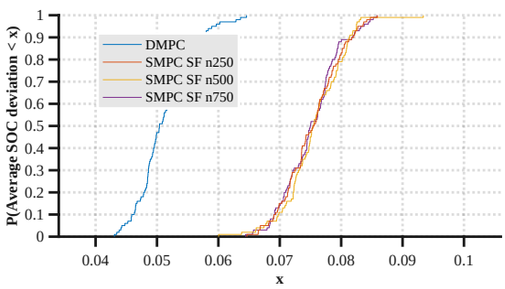
SMPC full parameterization As one can observe from figure 4.22 the average frequency deviation and units of fuel curves are overlapping for SMPC full parameterization and DMPC. No conclusions can be drawn from this plots other than that they perform equal. For the average SOC deviation and SOC standard deviation however, DMPC is significantly right shifted. This tells us that the battery degradation will happen more rapidly with the DMPC controller than for the SMPC FP.
In addition to the observations mentioned above, one can see by a close inspection that the SMPC curves are slightly left shifted with increasing number of scenarios. However, this is so marginal it is not considered significant.



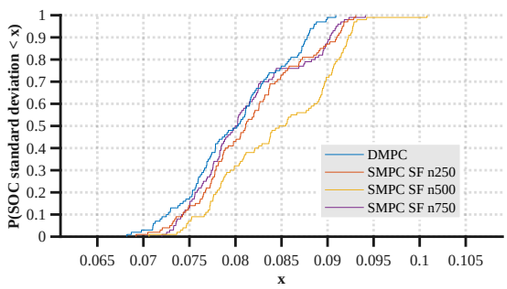
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



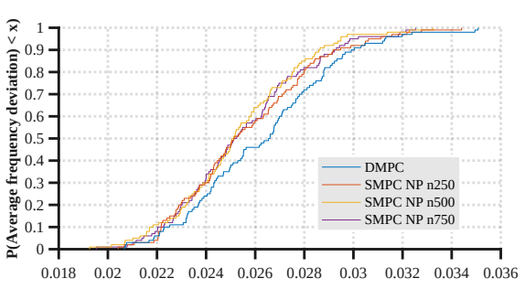
(c) Average SOC deviation, empirical curve



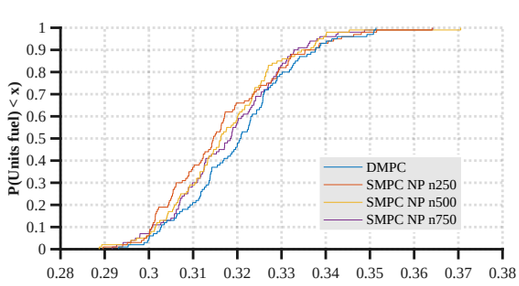
(d) SOC standard deviation, empirical curve

Figure 4.23: CDF for performance metrics, DMPC vs SMPC SF, 10% parametric uncertainty

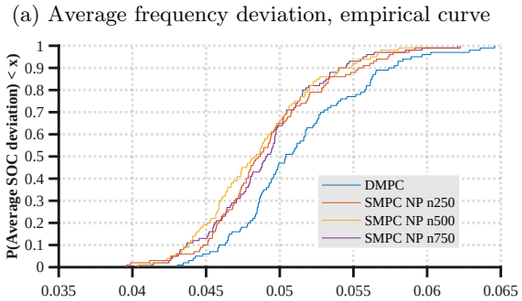
SMPC state feedback parameterization Figure 4.23 shows that the SMPC with state feedback is right shifted on all performance metrics except units of fuel. This means that for all the measured performance metrics the SMPC with state feedback is performing significantly worse compared to DMPC. The frequency is less accurately tracked and the battery is degrading faster. The only advantage of the SMPC SF is the conservative usage of fuel for the gas turbine. The interpretation from this is that the gas turbine is used passively, which results in the BESS trying to compensate for this which it does not accomplish. Perhaps this controller could have been tuned to reduce the cost of $\mathbf{\Delta u_1}$.



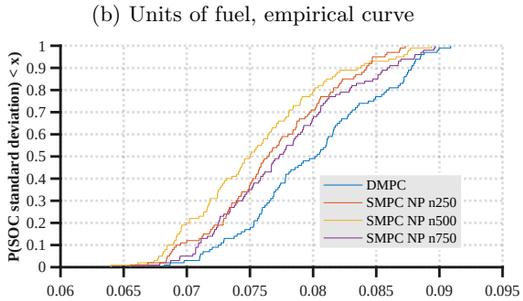
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



(c) Average SOC deviation, empirical curve

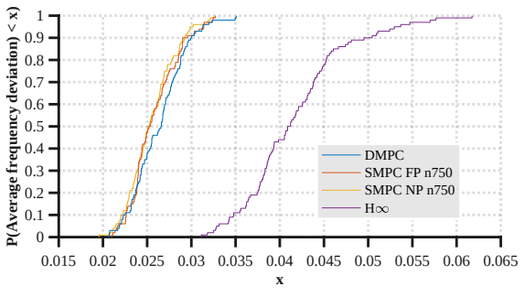


(d) SOC standard deviation, empirical curve

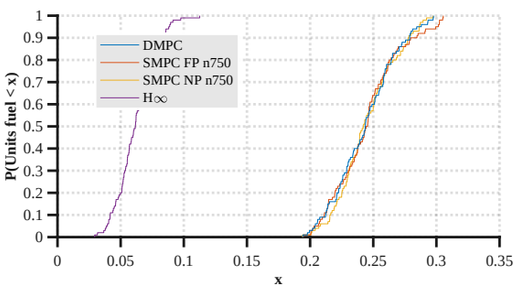
Figure 4.24: CDF for performance metrics, DMPC vs SMPC NP, 10% parametric uncertainty

No parameterization The SMPC controller with no parametrisation on the control action is left shifted for all metrics compared to the DMPC as can be seen from figure 4.24. Hence the SMPC NP is simply a better performing controller according to the metrics of evaluation used here. It achieves better frequency tracking, with less use of fossile fuel and less battery degradation compared to the DMPC.

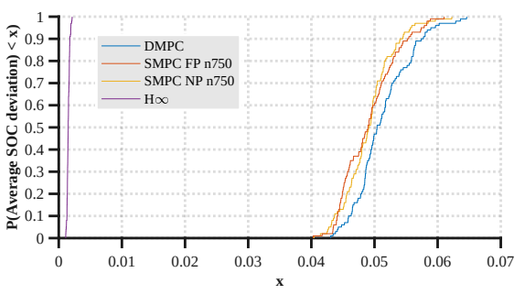
Another observation to be made, is that the number of scenarios collected for the SMPC NP does not seem to affect the average performance, as is also the case for the SMPC with the other control parameterizations.



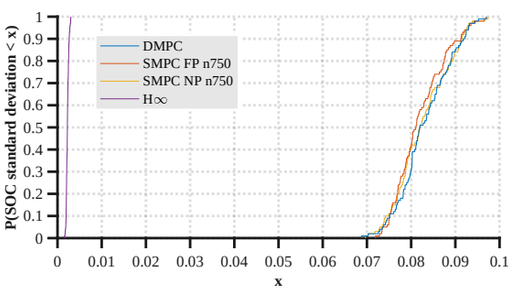
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



(c) Average SOC deviation, empirical curve



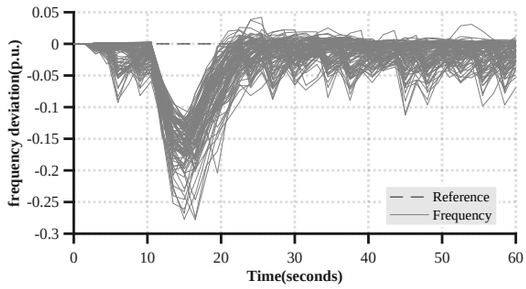
(d) SOC standard deviation, empirical curve

Figure 4.25: CDF for performance metrics, DMPC vs SMPC FP N=750 vs SMPC NP N=750 vs $H\infty$, 10% parametric uncertainty

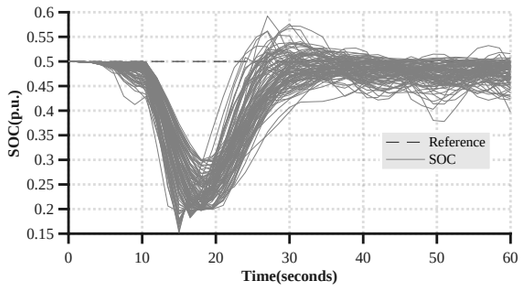
Mixed synthesis $H\infty$ controller what we can observe from figure 4.25 is a continuation of the observations made from figure 4.21. The frequency deviation is significantly larger, compared to all MPC controllers while the control action usage is conservative leading to little fuel consumption and slow battery degradation.

4.3 50% Parametric uncertainty and disturbance uncertainty

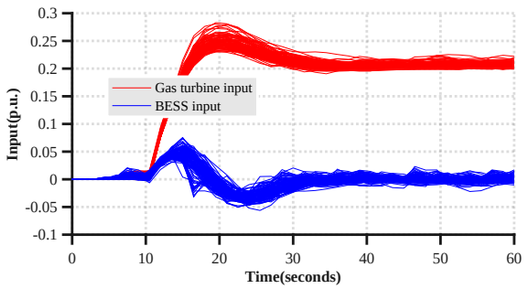
The simulations for 50% parametric uncertainty was conducted because of the suggestions in the literature[46] that this is a realistic scenario. This amount of uncertainty will also test the robustness of the controllers for extreme cases and if good performance is obtained, it will broaden the spectrum of applications for the SMPC controllers.



(a) Frequency deviation

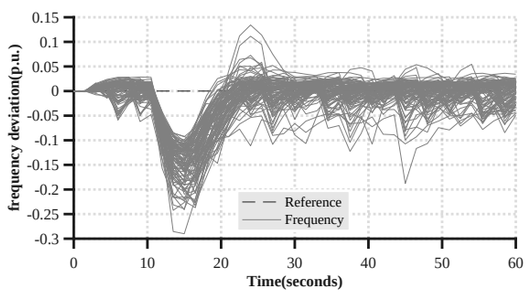


(b) SOC deviation

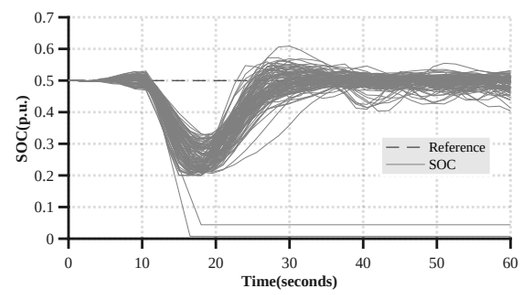


(c) Gas turbine and BESS input

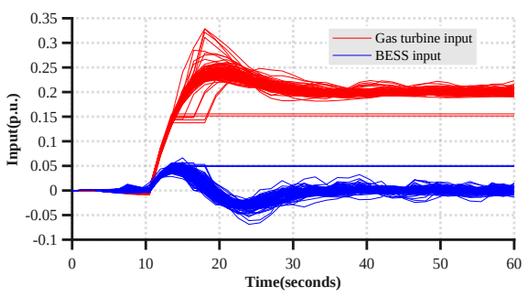
Figure 4.26: 50% parametric uncertainty and uncertain wind power generation behaviour for DMPC



(a) Frequency deviation

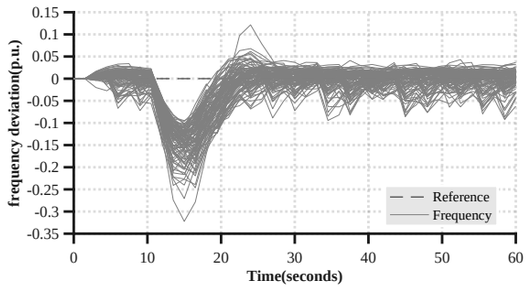


(b) SOC deviation

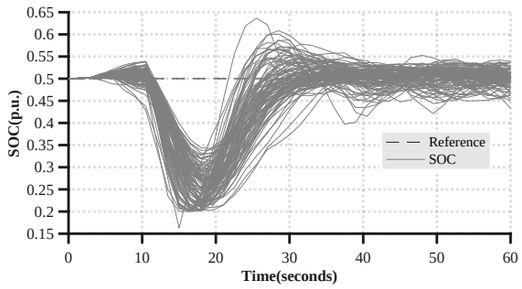


(c) Gas turbine and BESS input

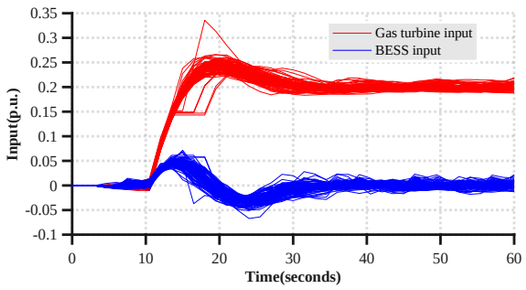
Figure 4.27: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 250 scenarios



(a) Frequency deviation

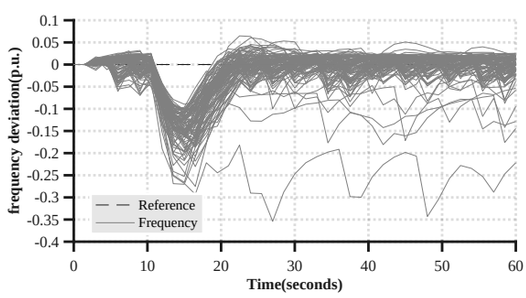


(b) SOC deviation

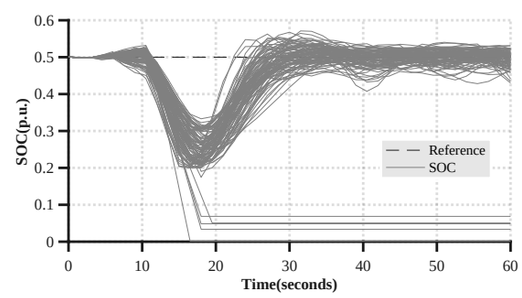


(c) Gas turbine and BESS input

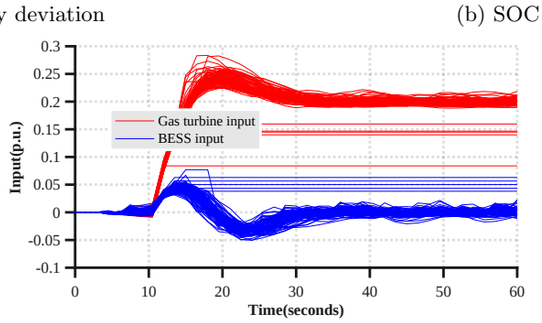
Figure 4.28: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 500 scenarios



(a) Frequency deviation

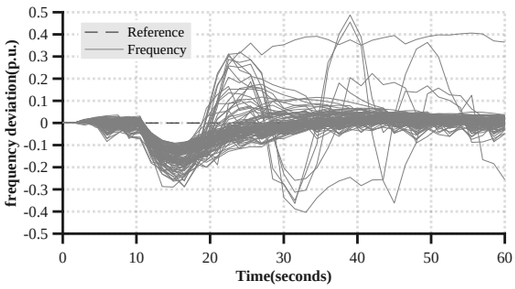


(b) SOC deviation

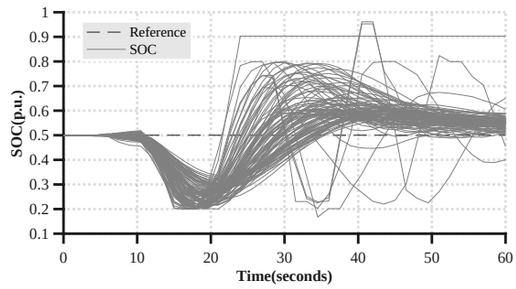


(c) Gas turbine and BESS input

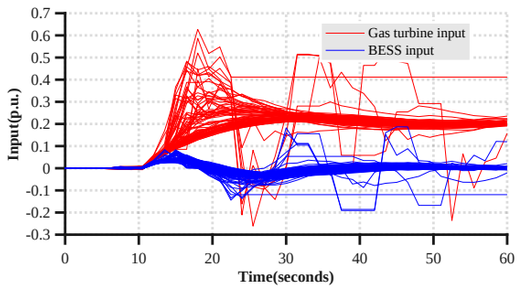
Figure 4.29: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC FP 750 scenarios



(a) Frequency deviation

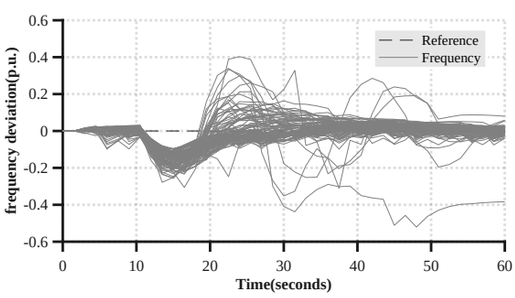


(b) SOC deviation

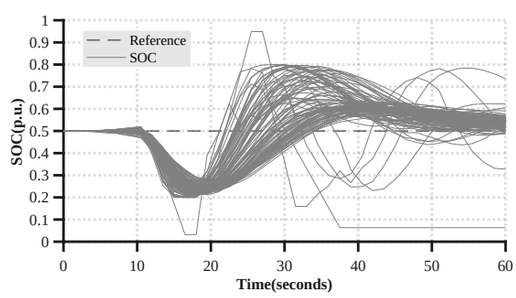


(c) Gas turbine and BESS input

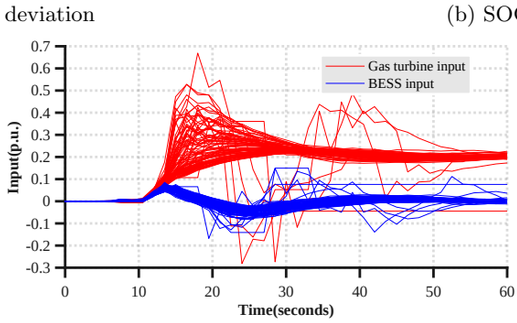
Figure 4.30: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 250 scenarios



(a) Frequency deviation

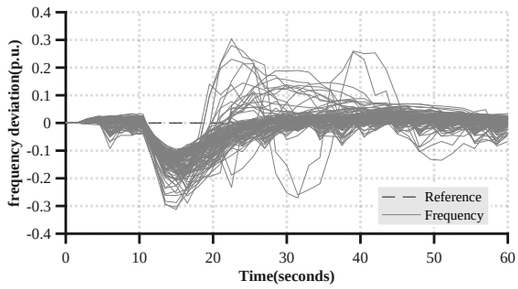


(b) SOC deviation

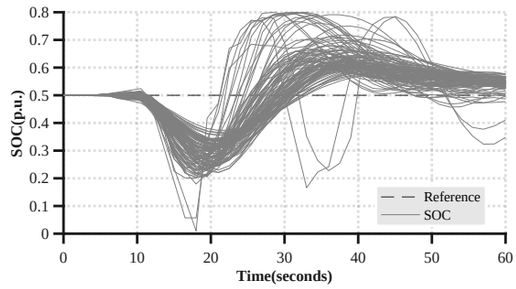


(c) Gas turbine and BESS input

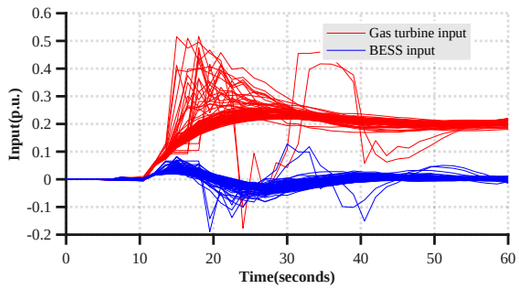
Figure 4.31: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 500 scenarios



(a) Frequency deviation

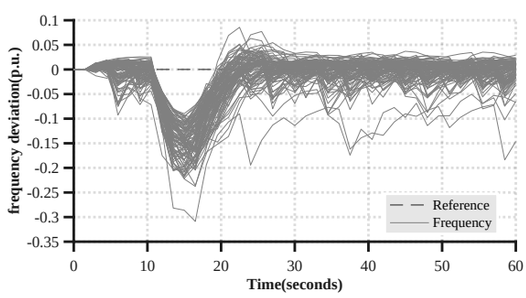


(b) SOC deviation

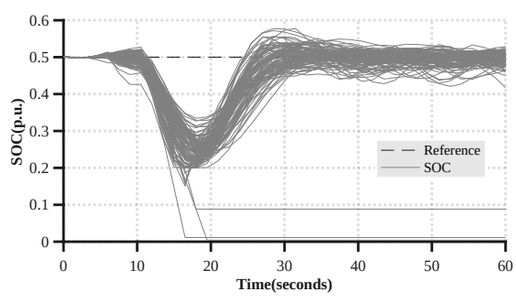


(c) Gas turbine and BESS input

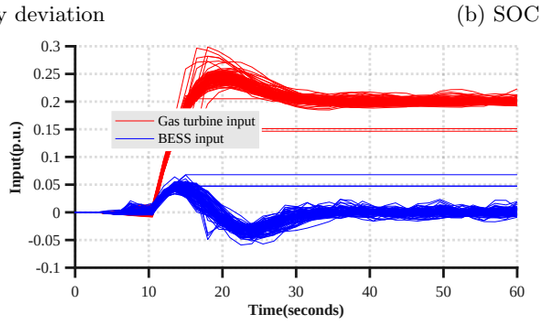
Figure 4.32: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC SF 750 scenarios



(a) Frequency deviation

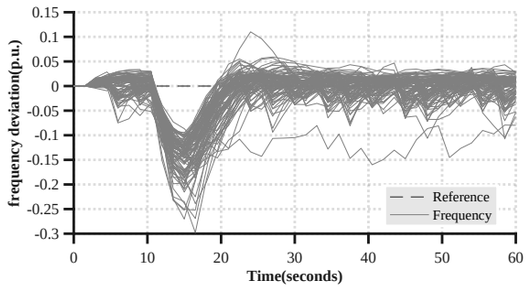


(b) SOC deviation

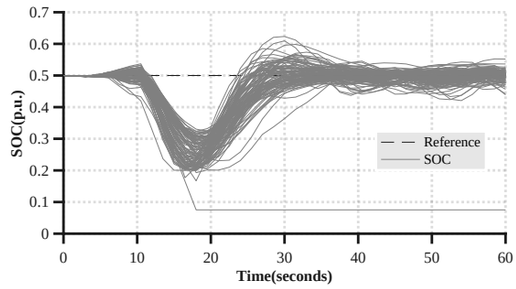


(c) Gas turbine and BESS input

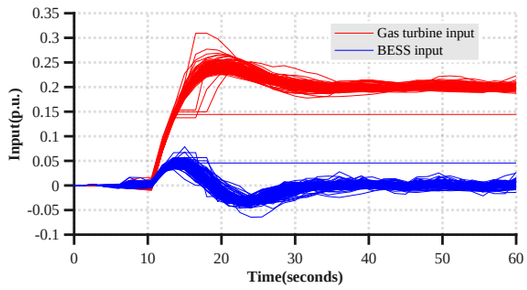
Figure 4.33: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 250 scenarios



(a) Frequency deviation

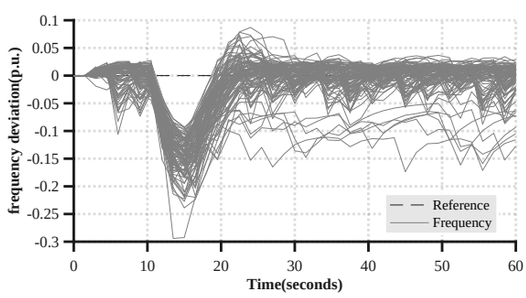


(b) SOC deviation

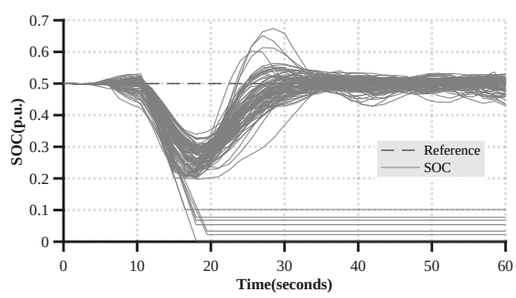


(c) Gas turbine and BESS input

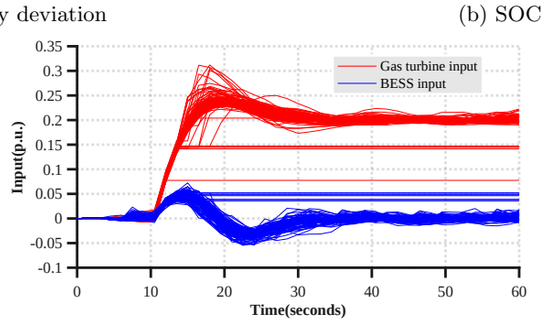
Figure 4.34: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 500 scenarios



(a) Frequency deviation

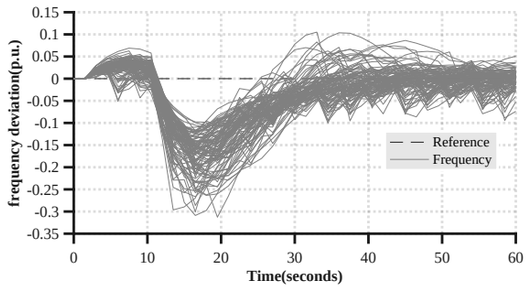


(b) SOC deviation

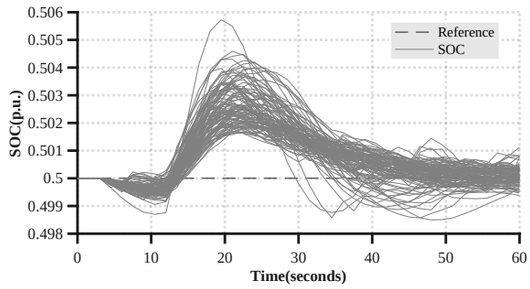


(c) Gas turbine and BESS input

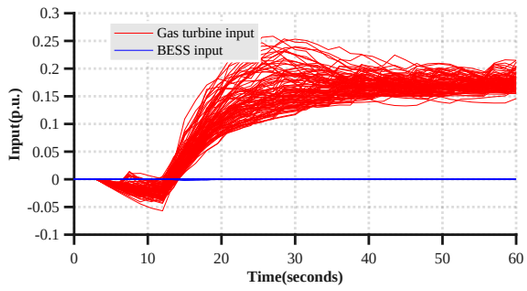
Figure 4.35: 50% parametric uncertainty and uncertain wind power generation behaviour for SMPC NP 750 scenarios



(a) Frequency deviation



(b) SOC deviation



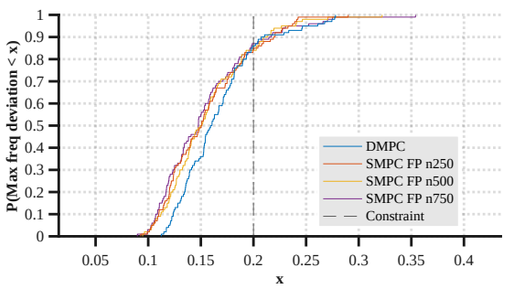
(c) Gas turbine and BESS input

Figure 4.36: 50% parametric uncertainty and uncertain wind power generation behaviour for $H\infty$ controller

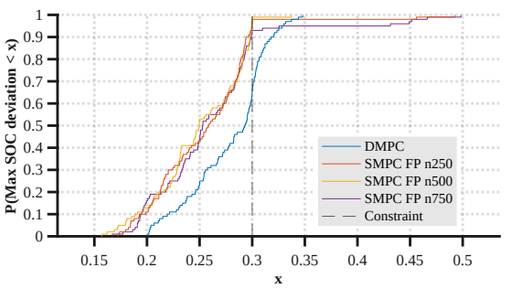
4.3.1 Constraint violation

For this section, only the empirical plots are presented. This is because of the large deviation between the fitted normal distribution curves and the empirical curves.

SMPC full parameterization A key observation to make from the raw data graphs in figures 4.27 to 4.29 and which affects the results for the constraint violation plots in figure 4.37 is that the SMPC FP reaches an infeasible state for some scenarios. As we can see, increasing the number of scenarios results in an increase in the infeasibility occurrences. Another interesting observation is that the DMPC controller has a high SOC constraint violation probability. This is vastly different from the simulations with 10% parametric uncertainty.



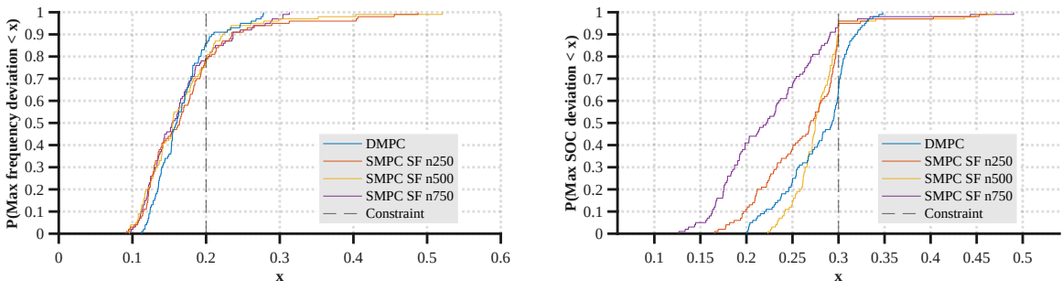
(a) Frequency violation, empirical curve



(b) SOC violation, empirical curve

Figure 4.37: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC FP with increasing scenario collection and 50% parametric uncertainty

SMPC state feedback For the state feedback parameterization, the correlation between the number of scenarios and the number of infeasibility occurrences is not so clear. Note also that the number of scenarios is not correlated with the left shift of the curve either. This is the same observation that was made for 10% parametric uncertainty.

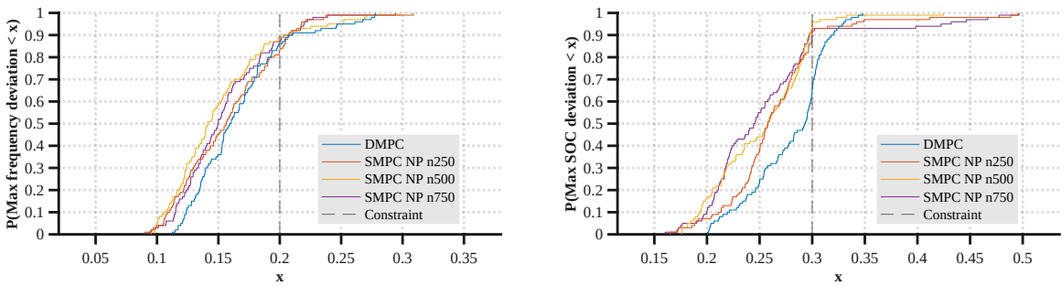


(a) Frequency violation, empirical curve

(b) SOC violation, empirical curve

Figure 4.38: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC SF with increasing scenario collection and 50% parametric uncertainty

SMPC no parameterization As for the SMPC FP, the SMPC NP has an increased number of infeasibility occurrences for higher scenario collections.

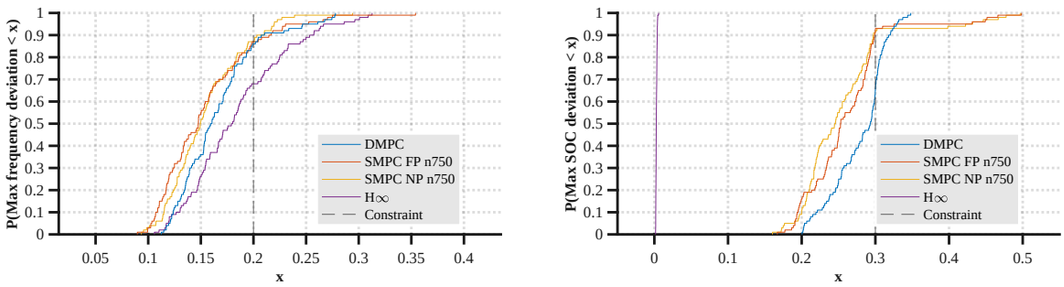


(a) Frequency violation, empirical curve

(b) SOC violation, empirical curve

Figure 4.39: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC NP with increasing scenario collection and 50% parametric uncertainty

Mixed synthesis $H\infty$ controller If one compares figure 4.21a to figure 4.40a, an interesting observation is that the performance of the $H\infty$ controller in terms of frequency violation probability is quite similar, while it has worsened drastically for the MPC controllers. However, the constraint violation probability is still higher for the $H\infty$ controller.



(a) Frequency violation, empirical curve

(b) SOC violation, empirical curve

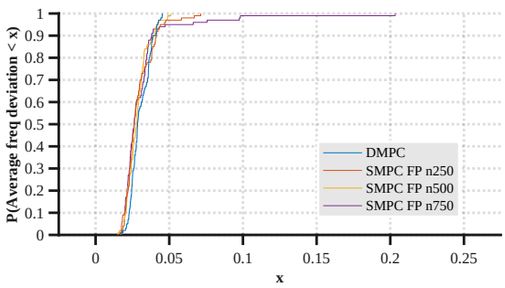
Figure 4.40: CDF for maximal frequency deviation and maximal SOC deviation, DMPC vs SMPC FP N=750 vs SMPC NP N=750 vs $H\infty$, 50% parametric uncertainty

100 samples	Empirical		Theoretical
	Frequency	SOC	
DMPC	14%	35%	-
$H\infty$	32%	0%	-
SMPC FP N = 250	15%	3%	24.90%
SMPC FP N = 500	16%	1%	12.95%
SMPC FP N = 750	13%	7%	8.85%
SMPC SF N = 250	12%	6%	11.30%
SMPC SF N = 500	23%	5%	5.95%
SMPC SF N = 750	12%	4%	4.00%
SMPC NP N = 250	17%	7%	11.30%
SMPC NP N = 500	11%	4%	5.95%
SMPC NP N = 750	13%	8%	4.00%

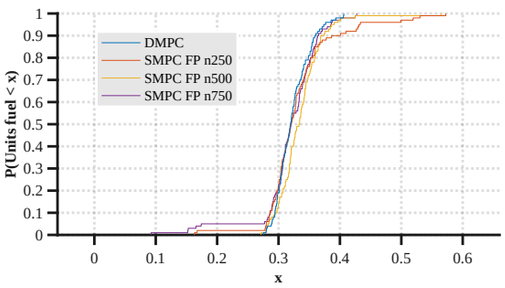
Table 4.3: Violation probability for maximal frequency and SOC deviation constraints for DMPC vs SMPC FP with increasing scenario collection and 50% parametric uncertainty

4.3.2 Performance

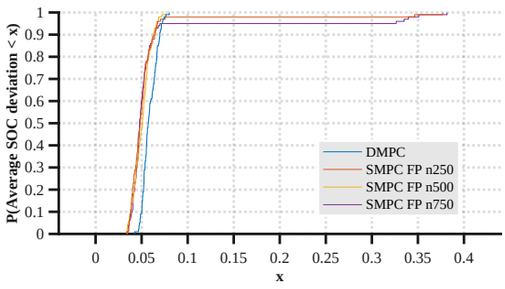
SMPC full parameterization Comparing figure 4.22 to figure 4.41, the only striking difference, a part from a slight general left shift of the curves, is how the infeasibility occurrences results in a worst case average performance which is much worse for 50% uncertainty. This is not surprising because, infeasibility results in a non-optimal control action.



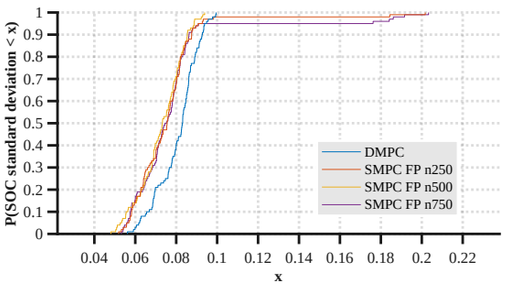
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



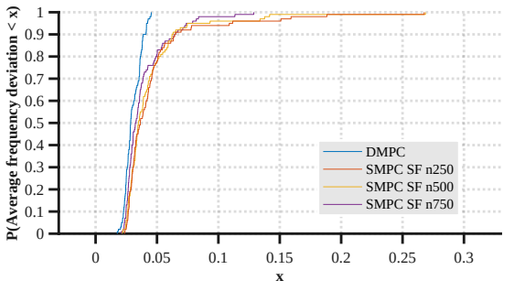
(c) Average SOC deviation, empirical curve



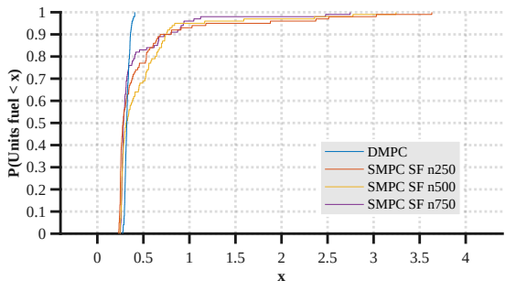
(d) SOC standard deviation, empirical curve

Figure 4.41: CDF for performance metrics, DMPC vs SMPC FP, 50% parametric uncertainty

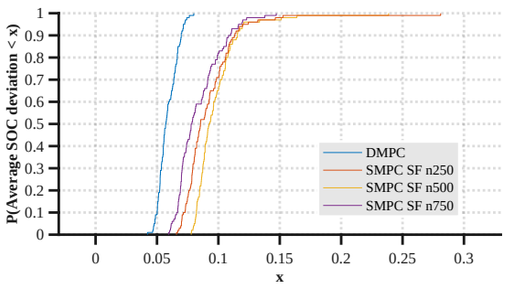
SMPC state feedback A comparison of the 10% and 50% parametric uncertainty graphs for the SMPC SF showed that compared to the DMPC, SMPC SF is not performing a lot better. While it was right shifted for every metric except SOC standard deviation for 10% uncertainty, it is now significantly right shifted only for the average SOC deviation. Hence, the SMPC SF handles 50% parametric uncertainty better than DMPC.



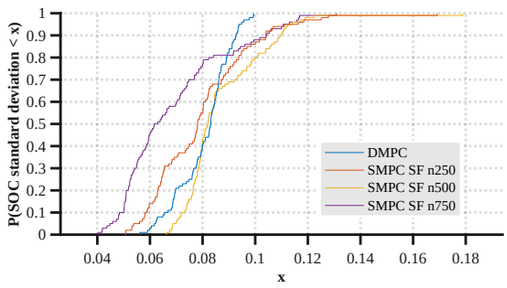
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



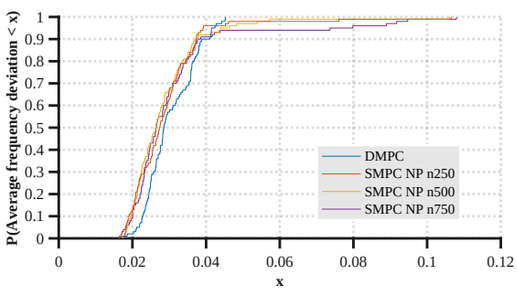
(c) Average SOC deviation, empirical curve



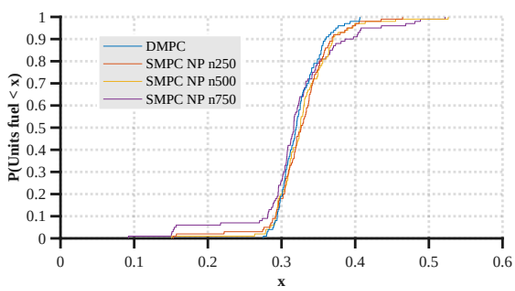
(d) SOC standard deviation, empirical curve

Figure 4.42: CDF for performance metrics, DMPC vs SMPC SF, 50% parametric uncertainty

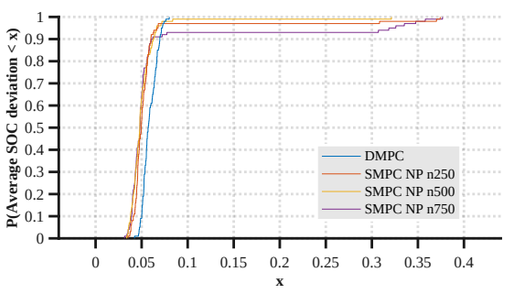
SMPC no parameterization The performance pattern of the SMPC NP for 50% parametric uncertainty is quite similar as for SMPC FP. The curves are generally left shifted, but the infeasibility cases affects the worst case performance of the SMPC NP.



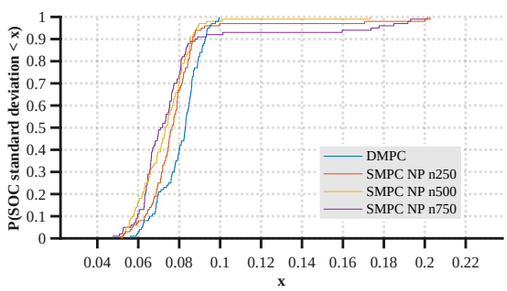
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



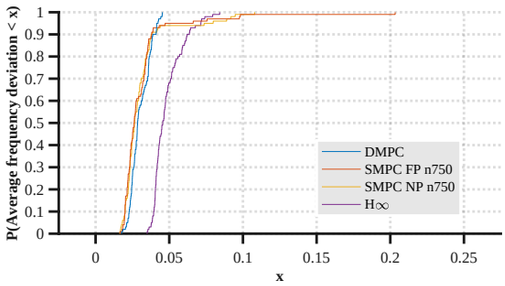
(c) Average SOC deviation, empirical curve



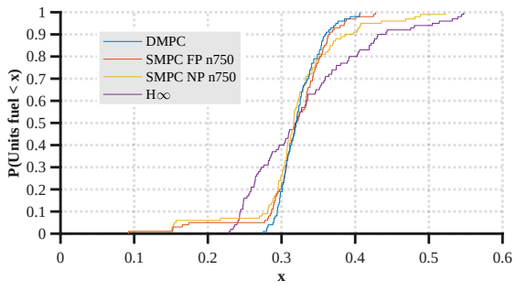
(d) SOC standard deviation, empirical curve

Figure 4.43: CDF for performance metrics, DMPC vs SMPC NP, 50% parametric uncertainty

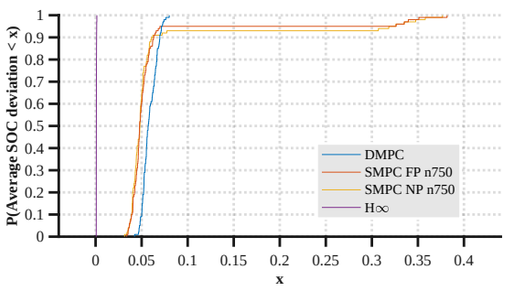
Mixed synthesis $H\infty$ controller An interesting observation from figure 4.44 is how the units of fuel metric has left shifted compared to the 10% parametric uncertainty plot in figure 4.25. This is presumably caused by the fact that the bandwidth of the $H\infty$ controller had to be further reduced for 50% parametric uncertainty. This resulted in a slight overshoot of the gas turbine input, which is reflected in the units of fuel metric.



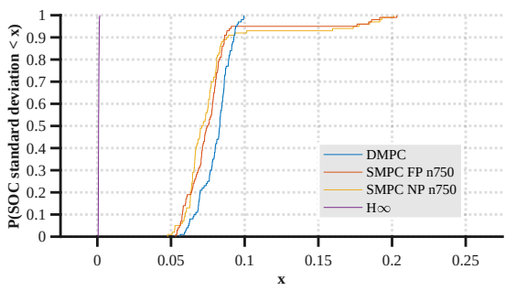
(a) Average frequency deviation, empirical curve



(b) Units of fuel, empirical curve



(c) Average SOC deviation, empirical curve

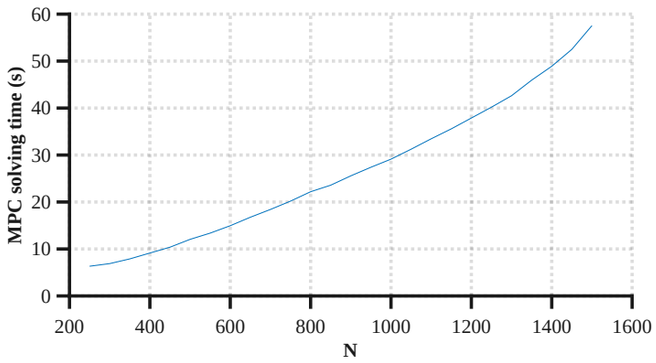


(d) SOC standard deviation, empirical curve

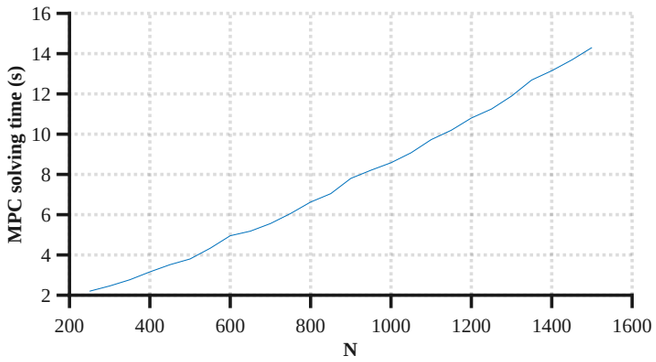
Figure 4.44: CDF for performance metrics, DMPC vs SMPC FP N=750 vs SMPC NP N=750 vs $H\infty$, 50% parametric uncertainty

4.4 MPC calculation time

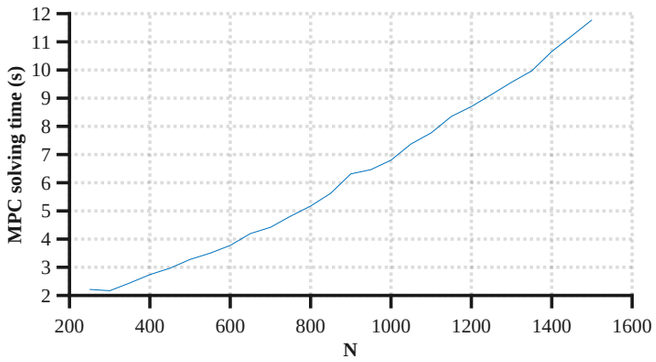
Another performance metric which so far has not been mentioned is the calculation time of the different MPC control strategies. This metric is decoupled from the above performance metrics, but might be one of the most critical ones. For the controller to even be feasible for the application, the calculation time, which is the time it takes to calculate the optimal solution to the optimization problem and output the control action, must be smaller than the time step. The graphs shown in figure 4.45 correlates the number of collected scenarios to the average calculation time over the course of one simulation.



(a) MPC solving time SMPC FP



(b) MPC solving time SMPC SF



(c) MPC solving time SMPC NP

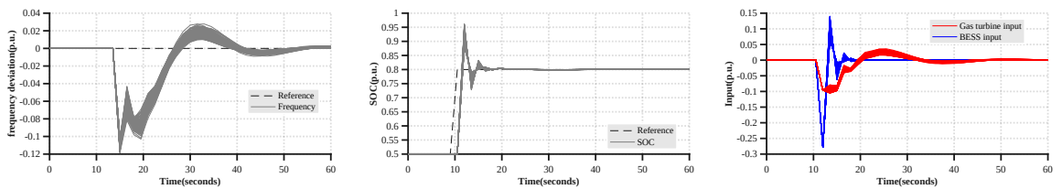
Figure 4.45: MPC calculation time (with overhead) for increasing scenario collection

The shapes of the curves are quite similar. By inspection, one can see that the curve resembles a linear curve in the plotted region. There is a small tendency, especially in figure 4.45a that the relation is actually non-linear for higher number of scenarios, but this is not considered relevant for our purposes.
Figure 4.45a also reveals that SMPC FP has the by far largest computational cost. With 250 scenarios, the average calculation time is approximately 5 seconds and with 1500 scenarios the calculation time is over 50 seconds. This is a huge problem for our application because the upper limit of the calculation time is the time step which is 1.5 seconds.
The calculation time for SMPC SF and SMPC NP is more equal, however the state feedback parameterization is some seconds slower. This implicates that the number of variables is vital for the calculation time. Where the SMPC FP has 32 variables for our case, the SMPC SF and SMPC NP only have 8 variables. Note that all controllers has too high calculation time in the current implementation, since they are all above the time step Δt. Whether or not the SMPC controllers are feasible for our application is discussed in detail in section 5.4.

5 Discussion of results

5.1 Nominal behaviour

The nominal results with no uncertainty in the parameters or disturbance, gives a good understanding of the basic functioning of the controllers. As mentioned in section 4.1, we can observe that the DMPC, SMPC FP, and SMPC NP are in fact behaving equally for the nominal case because of the equal tuning. However, SMPC SF is tuned with different weights because of the feedback gain K which affects the behaviour. The SMPC SF has a slower frequency response, which is caused by the conservative usage of the control action as can be seen in figure 4.3c. Notice that the SOC is already overshooting and therefore the usage of control action should not be increased as this would lead to an even higher overshoot.
The most striking result from the nominal behaviour is the poor BESS management of the $H\infty$ controller. At first glance it may look like the usage of the BESS is simply restricted too much, but if one looks closely, the SOC is actually rising instead of decreasing after the load step increase. This means that the battery is slowing the frequency response and hence it is necessary for the $H\infty$ to restrict the BESS control action. This response was first considered to be unreasonable and for this reason, a SOC reference step was simulated to check the dynamics of the model and controller, and evaluate the response. This gave the following result



(a) Frequency deviation (b) SOC (c) Gas turbine and BESS input

Figure 5.1: $H\infty$ controller behaviour for SOC reference step, 10% parametric uncertainty

As we can see from figure 5.1, the $H\infty$ controller manages to follow the reference step with a precise control action. This tells us that the dynamics are modeled properly in the transfer function of the plant, and that the $H\infty$ controller has the right information on what control action leads to what SOC output. We can also observe that the frequency drops as the BESS charges, which makes perfect sense as it draws power from the grid. We therefore conclude that this reveals a weakness for the MIMO $H\infty$ controller in tracking multiple references with multiple control actions.

5.2 Empirical validation study for 10% parametric uncertainty and variable wind generation

The first observation we can make simply by inspection of the raw data graphs in figures 4.7 to 4.17, is that the controller with the most outliers is the SMPC SF. Where the other controllers seem to have an evenly distributed grey scenario spectrum, the SMPC SF has some extreme outliers for all the different numbers of collected scenarios. This is unfortunate when the goal is to achieve robust results with low violation probability and, as one can see from table 4.2, the empirical violation probability is higher than the theoretical value for the simulated scenarios.

By inspecting the cumulative probability density function (CDF) plots of the maximal frequency and SOC deviation in figures 4.18 to 4.21, we can draw some conclusions with regards to the constraint violation probabilities of the different controllers. We can observe that the two controllers that are left shifted in comparison with the DMPC, are the SMPC FP and the SMPC NP. Hence, both the $H\infty$ controller and the SMPC SF have a higher constraint violation probability compared to the DMPC. However, what separates the SMPC FP and SMPC NP is that the SMPC FP is the only controller who manages to show a consistent left shift in the CDF plots when increasing the number of collected scenarios. This is considered to be the effect of the disturbance feedback in the SMPC FP, which adjusts the optimal control action based on the realizations of the disturbance. Hence, we can observe the effect of the control parameterization, which was discussed in section 2 paragraph **Control parameterization**.
When looking at the CDF plots for the maximal SOC deviation, we observed in section 4.2.1 that the SOC constraint was never violated but there were many cases where the constraint was active. This was a surprising result as the frequency deviation has another behaviour in terms of constraint violation and they both are influenced by parametric uncertainty. However, a key difference is the fact that the variable wind generation affects the frequency directly while only affecting the SOC indirectly through the control action. This is a possible reason for the

different behaviour.
When we turn our attention to the average performance metrics in figure 4.22 to 4.25, the conclusion is that the DMPC generally outperforms SMPC FP and SMPC NP in terms of average performance. This is as expected because the SMPC controllers are more conservative and robust controllers than the DMPC because of the scenario collection. We can also observe that the number of scenarios does not seem to have a big impact on the average performance. This is an interesting observation which strengthens the power of scenario collection. When investigating the performance of the SMPC SF and $H\infty$ controller, the difference in nominal behaviour is also prominent for the uncertainty results. SMPC SF and $H\infty$ generally fails to hit the balance between control action usage and reference tracking.

5.3 Empirical validation study for 50% parametric uncertainty and variable wind generation

The most prominent difference between the 10% parametric uncertainty simulations and the 50% parametric uncertainty simulations was the introduction of infeasibility occurrences. This happens because the additive uncertainty in some cases leads to inaccurate predictions from the MPC controllers and the reference deviation increases. This again leads to infeasibility because the SMPC controller cannot manage to find a control action that satisfies the constraint for all the collected scenarios. As we observed from the constraint violation probability analysis for the SMPC FP and SMPC NP, the chance of infeasibility increased with the number of collected scenarios. This is logical because there are more constraints to satisfy. This also demonstrates the danger of implementing arguably soft constraints as hard constraints, as was done for the SOC and frequency desired maximal deviations.
An important remark is that the infeasibility actually results in a constraint violation that is higher than the theoretical "guaranteed" violation probability. This demonstrates a weakness in the scenario theory which is also discussed in the literature, see for example [24] page 9.
Also note that even though the SMPC reaches infeasibility, the constraint violation probability is still higher for DMPC and $H\infty$.
As mentioned in the results observations, the bandwidth of the $H\infty$ controller was increased for 50% parametric uncertainty to obtain a stable controller. This resulted in a higher gas turbine fuel consumption. However, for real-world applications this would not have been necessary because the time step would have been lower. As mentioned earlier, it was considered necessary to have the same time step for all controllers to obtain a fair comparison. This remains a topic for discussion.

5.4 MPC solving time

The solving time plot in figure 4.45 of the SMPC controllers reveal a potential problem. The correlation between the number of variables and the MPC solving time tells us that the feedback information, which gives good constraint violation performance, comes at the cost of linearly increased solving time. With the implementation of the SMPC controllers in this thesis, they are infeasible for an application with a time step of $\Delta t = 1.5s$. However, there can be taken several measures to reduce the computational time.
The first one is to optimize the code. Code optimization, like removing FOR loops and other time consuming operations, can drastically reduce computational time. This has been done iterative for the SMPC in this thesis and has reduced the computational time by large fractions of up to 90% for early iterations. There are reasons to believe that it is still some room for additional gains.
Secondly, the overhead computations can be done offline. This means setting up the optimization problem for the solver and performing any sort of operations independent on the state of the system. The computational time in this thesis includes all calculations and is therefore not a satisfactory indication of the maximal performance. The actual CPU time, which is the time spent by the CPU on solving the optimization problem, can give a better prediction for the feasibility of the SMPC. This may be provided by the solver, however for CVX with SDPT3, the CPU time was not given as a data struct but only as printed information and was therefore not analyzed. By inspection of some random samples, the actual CPU time is estimated to be approximately 25% of the calculation time given in figure 4.45.
Lastly, faster hardware can reduce the CPU time. The simulations were performed on a 2010 MacBook Pro with 2,4 GHz Intel Core 2 Duo processor. With new technology showing the potential for 10 GHz processors [43], there is potential for 4 times faster CPU calculation time.

5.5 Concluding remarks

In this section, we will try to summarize and draw a couple of conclusions from the work and results presented in this thesis.
Firstly, we can conclude that the MPC controller has an ability to handle a MIMO system in a way that $H\infty$ methods cannot match. The results in the project leading to this thesis[44] were similar for the comparison between a multiple loop PID controller and MPC. The ability to coordinate resources by minimizing the tailor made objective function is both powerful and intuitive.
Another conclusion is that in terms of average performance, the SMPC does not give a performance benefit compared to the DMPC. This is neither controversial nor surprising as the purpose of the SMPC is to achieve a robust and optimal handling of uncertainty in regards to constraint handling.
In terms of constraint violation, the SMPC FP is the best performing controller in minimizing the maximal reference deviation for both objectives, and achieve a consistent correlation between the constraint violation probability and the number of scenarios. However, this comes at the cost of higher computational time.
An important lesson from the simulations with 50% parametric uncertainty was how implementing constraints as hard constraints can lead to infeasibility for additive uncertainty. This gives unwanted behaviour for the cases of constraint violation, which might undermine the purpose of stochastic control that allows constraint violation on the premise of acceptable behaviour and consequences when violation occurs.
The feasibility of SMPC is a topic for further research and cannot be answered with high confidence based on the results gathered in this thesis. SMPC with fewer optimization variables is however much more likely to be feasible, while the full parameterization is less likely.

5.6 Future work

As mentioned earlier, this is an active field of research and there are many interesting directions of further developing this study. Some of the most prominent extensions are listed in this section.
Firstly, including more control parameterizations is a natural action to take because the results showed how the SMPF FP gives better constraint violation performance and steadily increased performance with the number of scenarios, while the MPC calculations time was in the absolute upper region of what is feasible for this application. As presented in [24], there are several suggestions for parameterizations with a number of variables that is in between the full parameterization and no parameterization.
Secondly, introducing scenario reduction algorithms could be a possible extension. The effectiveness of such algorithms is demonstrated in [24] and others. However, the challenge of the short time step for our application may leave such online algorithms infeasible. Investigating how the effect of such algorithms is in theory would still be interesting, and testing the naive scenario reduction algorithm proposed in section 2 paragraph **Scenario reduction algorithms** could perhaps lead to improved performance.
Thirdly, collecting real-world model parameters for the gas turbine generator, wind turbine generator, BESS, and load condition will also give more realistic simulations and, more importantly, it can give good estimates for the bounds of the uncertainty. The difference between 10% and 50% parametric uncertainty is significant in terms of which controller is optimal for the given application.
Another interesting extension could be to improve the battery degradation modeling and penalizing. In this thesis, this was done simply by penalizing the SOC deviation and the SOC standard deviation and this gave descent results. However, more accurate cycle counting algorithms exist like the Rainflow algorithm. As discussed in section 3.1.4, this method is non-convex but developing a regression model, preferably linear, between the SOC profile and the cycle count is a possible solution. This is a complex idea and could probably be the focus of a separate thesis or paper.

References

[1] P. M. Ashton, C. S. Saunders, G. A. Taylor, A. M. Carter, M. E. Bradley.
Inertia Estimation of the GB Power System Using Synchrophasor Measurements
IEEE Transactions on Power Systems 30(2):701-709 march 2015.

[2] Seaseung Oh, Suyong Chae, Jason Neely, Jongbok Baek, Marvin Cook.
Efficient Model Predictive Control Strategies for Resource Management in an Islanded Microgrid
Energies 2017, 10, 1008.

[3] Ziba Rostami, Sajad Najafi Ravadanegh, Navid Taghizadegan Kalantari.
Dynamic Modeling of Multiple Microgrid Clusters Using Regional Demand Response Programs
Energies 2020, 13, 4050.

[4] Simone Garatti, Marco C. Campi.
Modulating Robustness in Control Design
IEEE CONTROL SYSTEMS MAGAZINE p.36-51, April 2013.

[5] Giuseppe Calafiore, M.C. Campi.
Robust Convex Programs: Randomized Solutions and Application in Control
Proceedings of the 42nd IEEE, Conference on Decision and Control, Maui, Hawaii USA, December 2003.

[6] Kutaiba S. El-Bidairi, Hung Duc Nguyen, Thair S. Mahmoud, S.D.G. Jayasinghe, Josep M. Guerrero. *Optimal sizing of Battery Energy Storage Systems for dynamic frequency control in an islanded microgrid: A case study of Flinders Island, Australia*
Energy 195 (2020) 117059.

[7] Philipp Fortenbacher, Johanna L. Mathieu, Göran Andersson.
Modeling and Optimal Operation of Distributed Battery Storage in Low Voltage Grids
IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 32, NO. 6, NOVEMBER 2017.

[8] Antonio Pepiciello, Alfredo Vaccaro, Domenico Villacci, Federico Milano.
A Method to Evaluate the Inertial Response of Frequency Controlled Converter-Interfaced Generation
IEEE, 2020.

[9] Wen Tan.
Unified Tuning of PID Load Frequency Controller for Power Systems via IMC
IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 25, NO. 1, FEBRUARY 2010.

[10] Ting Yang, Yajian Zhang, Zhaoxia Wang, Haibo Pen.
Secondary Frequency Stochastic Optimal Control in Independent Microgrids with Virtual Synchronous Generator-Controlled Energy Storage Systems
Energies 2018, 11, 2388.

[11] Philipp Fortenbacher, Johanna L. Mathieu, Göran Andersson.
Modeling, Identification, and Optimal Control of Batteries for Power System Applications
IEEE xplore.

[12] Ekaterina Gavenas, Knut Einar Rosendahl, Terje Skjerpen.
CO_2-emissions form Norwegian oil and gas extraction
Discussion Papers, Statistics Norway, Research department, No. 806 April 2015.

[13] Arno J. Brand, Joachim Peinke, Jakob Mann.
Turbulence and Wind Turbines
Journal of Physics Conference Series 318(7), December 2011.

[14] Ahmadreza Abazari, Hassan Monsef, Bin Wu.
Coordination strategies of distributed energy resources including FESS, DEG, FC and WTG in load frequency control (LFC) scheme of hybrid isolated microgrid
Int. J. Electrical Power & Energy Systems, 2019, 109, 535–547.

[15] Álvaro Ortega, Federico Milano.
Generalized Model of VSC-Based Energy Storage Systems for Transient Stability Analysis
IEEE Transactions on Power Systems, V.31, I.5, September 2016

[16] Liuping Wang.
Model Predictive Control System Design and Implementation Using MATLAB®
Springer, Advances in Industrial Control, 2009

[17] ViJay P. Singh, Soumya R. Mohanty, Nand Kishor, Prakash K. Ray.
Robust H-infinity load frequency control in hybrid distributed generation system
Electrical Power and Energy Systems 46 (2013) 294-305

[18] Vaishali Sohoni, S.C. Gupta, R.K. Nema.
A Critical Review on Wind Turbine Power Curve Modelling Techniques and Their Applications in Wind Based Energy Systems
Journal of Energy, 2016, Article ID 8519785

[19] James F. Manwell, Jon G. McGowan.
LEAD ACID BATTERY STORAGE MODEL FOR HYBRID ENERGY SYSTEMS
Solar Energy, Volume 50, Issue 5, May 1993, p. 399-405

[20] C. Gavriluta, S. Spataru, I. Mosincat, C. Citro, I. Candela, P. Rodriquez.
Complete methodology on generating realistic wind speed profiles based on measurements
presented at ICREPQ'12, march 2012 by EA4EPQ

[21] Pedro Andrè Carvalho Rosas.
Dynamic Influences of Wind Power on The Power System
Technical University of Denmark. Denmark. Forskningscenter Risoe. Risoe-R, No. 1408(EN), 2004

[22] Sigurd Skogestad, Ian Postlethwaite.
MULTIVARIABLE FEEDBACK CONTROL Analysis and design
Second Edition, This version: August 29, 2001, JOHN WILEY & SONS

[23] D.-W. Gu, P. Hr. Petkov and M. M. Konstantinov.
Robust Control Design with MATLAB
Springer-Verlag London Limited 2005

[24] Marco C. Campi, Simone Garatti and Maria Prandini.
Scenario optimization for MPC
In: Raković S., Levine W. (eds) Handbook of Model Predictive Control. Springer International Publishing AG, part of Springer Nature 2019

[25] Edwin González, Javier Sanchis, Sergio García-Nietoand José Salcedo.
A Comparative Study of Stochastic Model Predictive Controllers
Electronics, December 6, 2020

[26] M. Cannon, B. Kouvaritakis, and X. Wu.
Model predicitve control for systems with stochastic multiplicative uncertainty and probabilistic constraints
Automatica, 45:167–172, 2009

[27] P.J. Goulart, E.C. Kerrigan, and J.M. Maciejowski.
Optimization over state feedback policies for robust control with constraints
Automatica, 42(4):523–533, April 2006

[28] Yuanyuan Shi, Bolun Xu, Yushi Tan, and Baosen Zhang.
A Convex Cycle-based Degradation Model for Battery Energy Storage Planning and Operation
2018 Annual American Control Conference (ACC)

[29] Elias Hartvigsson, Erik O. Ahlgren *Comparison of load profiles in a mini-grid: Assessment of performance metrics using measured and interview-based data* Energy for Sustainable Development 43 (2018) 186–195

[30] Mathworks.com
`<https://se.mathworks.com/help/mpc/ug/choosing-sample-time-and-horizons.html>`
Accessed April 20th 2021.

[31] CVX user guide.
`<http://cvxr.com/cvx/doc/>`
Last edited in 2012, accessed 2th March 2021

[32] SDTP3 home page.
`<https://www.math.cmu.edu/~reha/sdpt3.html>`
Last edited August 2001, Acessed 1th April 2021

[33] Boyd and Vandenberghe.
Convex Optimization
Cambridge University Press, 2004

[34] Yu Zhang, Nikolaos Gatsis, Georgios B. Giannakis.
Robust Energy Management for Microgrids With High-Penetration Renewables
IEEE TRANSACTIONS ON SUSTAINABLE ENERGY, VOL. 4, NO. 4, OCTOBER 2013

[35] Panagiotis Patrinos, Sergio Trimboli and Alberto Bemporad.
Stochastic MPC for real-time market-based optimal power dispatch
2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)Orlando, FL, USA, December 12-15, 2011

[36] Stefano Di Cairano, Daniele Bernardini, Alberto Bemporad, and Ilya V. Kolmanovsky.
Stochastic MPC With Learning forDriver-Predictive Vehicle Control and its Application to HEV Energy Management
1018IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 22, NO. 3, MAY 2014

[37] Georg Schildbach, Lorenzo Fagiano, Christoph Frei, Manfred Morari.
The scenario approach for Stochastic Model Predictive Control with bounds on closed-loop constraint violations
Automatica 50 (2014) 3009–3018

[38] Wikipedia.org
<https://en.wikipedia.org/wiki/Fuel_cell#Research_and_development>
last edited on 23 May 2021, accessed 27 May 2021

[39] Wikipedia.org
<https://en.wikipedia.org/wiki/Superconducting_magnetic_energy_storage>
last edited on 13 May 2021, accessed 27 May 2021

[40] Wikipedia.org
<https://en.wikipedia.org/wiki/Supercapacitor#Energy_recovery>
last edited on 11 May 2021, accessed 27 May 2021

[41] Office of Energy Efficiency & Renewable Energy
<https://www.energy.gov/eere/solar/solar-integration-inverters-and-grid-services-basics>
Accessed 27 May 2021

[42] Heijungs, R.
On the number of Monte Carlo runs in comparative probabilistic LCA
Int J Life Cycle Assess 25, 394–402 (2020)

[43] sciencedaily.com
<https://www.sciencedaily.com/releases/2021/01/210107112418.htm>
Last edited January 7 2021, accessed 28th May 2021

[44] Andreas Faanes.
Model Predictive Controller Design for Isolated Wind Powered Microgrids
Project in TTK 4550 17th December 2020

[45] National Grid ESO
Electricity Safety, Quality and Continuity Regulations (ESQCR)
2002 (Amended 2006)

[46] Thongchart Kerdphol, Fathin Saifur Rahman, Yasunori Mitani, Masayuki Watanabe, Sinan Küfeoğlu.
Robust Virtual Inertia Control of an Islanded Microgrid Considering High Penetration of Renewable Energy
IEEE Access, vol. 6, pp. 625-636, 2018

[47] The Renewable Energy Hub UK.
<https://www.renewableenergyhub.co.uk/main/wind-turbines/wind-turbine-electronics/#jump_50>
Last edited January 21 2018, accessed 4th June 2021

[48] National Wind Watch.
<https://www.wind-watch.org/documents/how-turbulence-can-impact-power-performance/> Posted September 9th 2012, accessed 4th June 2021

[49] Statkraft.
<https://www.statkraft.com/what-we-do/hydropower/> Last edited 15th January 2021, accessed 4th June 2021

[50] Yuejun Jiang, Henry Kautz, and Bart Selman.
Solving Problems with Hard and Soft Constraints Using a Stochastic Algorithm for MAX-SAT
1st International Joint Workshop on Artificial Intelligence and Operations Research,Timberline,Oregon,1995

A Wind speed generator documentation

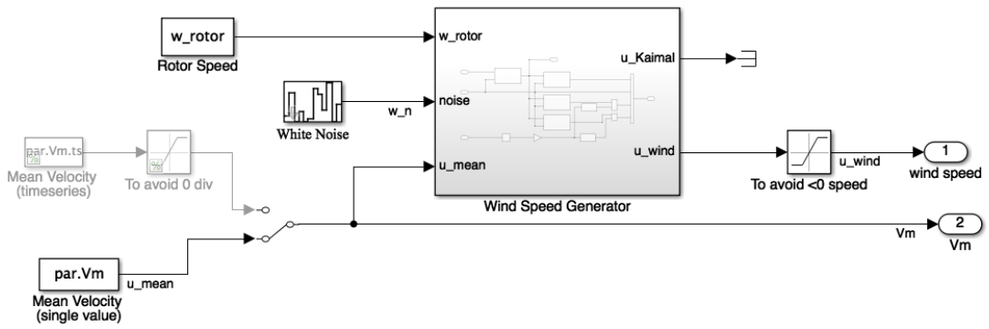


Figure A.1: Wind speed generator overall structure

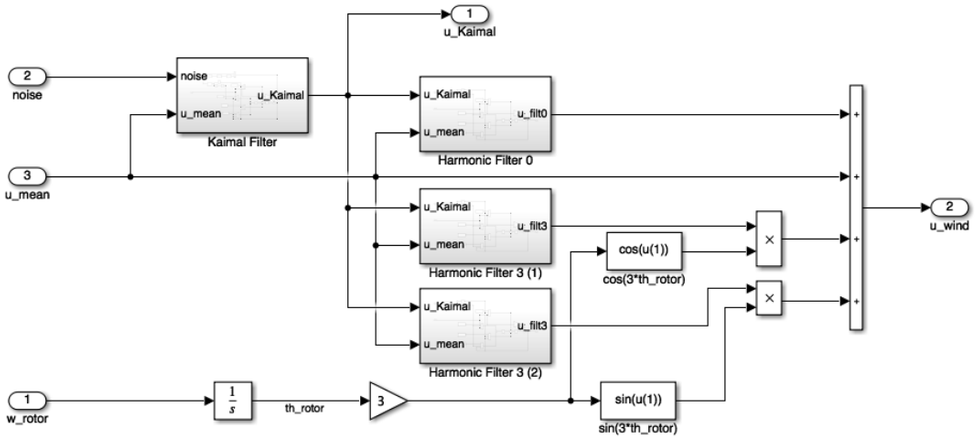


Figure A.2: Wind speed generator subsystem

B Controllers implementation code

B.1 Initialization script

```
%% initialization script
tic
clear logsout SMPCqpSFPCVX SMPCqpSSFCVX DMPCqpCVX batModel SMPCqpSNPCVX;
% global vars;
%% uncertainty parameter
% percentage = 0; % ONLY FOR INITIAL SIMULATION!   REMEMBERTOCHANGE
percentage = vars.mod.percentage;
%% simulation parameters
simTime = 60;
vars.simTime = simTime;
dt = 1.5;
vars.sim.dt = dt; % timestep
Np = 10;
vars.sim.Np = Np; % prediction horizon
Nc = 4;
vars.sim.Nc = Nc; % control horizon
vars.sim.loadstep = 0.2;
%% SCMPC parameters (comment out for monte carlo simulation)
% N = 10; % ONLY FOR INITIAL SIMULATION!   REMEMBERTOCHANGE
N = vars.SCMPC.N;
%% batery constants
vars.bat.SOC_r = 0.5; % state of charge reference value
cw0 = 0.93; % charge well factor
vars.bat.cw0 = cw0;
cr0 = 2.24*10^-5; % recovery factor
vars.bat.cr0 = cr0;

vars.bat.n = 1; % charge/discharge efficiency
% vars.bat.n_gen = 0.97; % discharge efficiency
Cbat0 = 1; % energy capacity(per unit)
```

```
vars.bat.Cbat0 = Cbat0;
vars.bat.Xinit = vars.bat.SOC_r/2; %initial states of battery
vars.bat.R = 1.5; % mOhm, internal resistance
vars.bat.Qbat = 31.25; % Ah, charge capacity
vars.bat.k1 = 1.5;
vars.bat.k2 = 4;
%% model constants
% nominal
T_gov0 = 0.05;
T_tur0 = 0.5;
T_wtg0 = 0.04;
T_bess0 = 0.1;
D0 = 1;
M0 = 3;
vars.plant.T_gov0 = T_gov0;
vars.plant.T_tur0 = T_tur0;
vars.plant.T_wtg0 = T_wtg0;
vars.plant.T_bess0 = T_bess0;
vars.plant.D0 = D0;
vars.plant.M0 = M0;
%% transfer functions
% nominal transfer functions
Hgov0 = tf([1],[T_gov0 1]);
Htur0 = tf([1],[T_tur0 1]);
Hwtg0 = tf([1],[T_wtg0 1]);
Hbess0 = tf([1],[T_bess0 1]);
Hmg0 = tf([1],[M0 D0]);
%% plant matricies
% nominal matrices
[vars.plant.A0, vars.plant.B0, vars.plant.H0] = ...
    tf2ss(Hgov0,Htur0,Hwtg0,Hbess0,Hmg0,vars.bat.cr0,...
vars.bat.cw0,vars.bat.n,vars.bat.Cbat0);
vars.plant.C = [0 0 0 1 0 0;
                0 0 0 0 1 1];
vars.plant.D = 0;
%% dimensions
[Ax,Ay] = size(vars.plant.A0);
[Bx,By] = size(vars.plant.B0);
[Hx,Hy] = size(vars.plant.H0);
Abx = 2;
Bbx = 2;
Bby = 1;
vars.dims.Ax = Ax;
vars.dims.Ay = Ay;
vars.dims.Bx = Bx;
vars.dims.By = By;
vars.dims.Hx = Hx;
vars.dims.Hy = Hy;
vars.dims.Abx = Abx;
%% recursive elimination matrices
A_e = zeros(Ax*Np*N,Ay);
B_0 = zeros(Bx*Np*N,By);
B_e = zeros(Bx*Np,By*Nc*N);
H_e = zeros(Hx*Np,Hy*Np*N);

SFA_e = zeros(Ax*Np*N,Ay);
SFB_0 = zeros(Bx*Np*N,By);
SFB_e = zeros(Bx*Np,By*Nc*N);
SFH_e = zeros(Hx*Np,Hy*Np*N);

A_e_bat = zeros(Abx*Np*N,Abx);
B_0_bat = zeros(Bbx*Np*N,Bby);
B_e_bat = zeros(Bbx*Np,Bby*Nc*N);

% LQR kontroller gain
% weights for state feedbac parametrisation
% control input Δ u
r = [100 0;0 50];
R = kron(eye(Nc),r);
R_chol = chol(R);
vars.w.SMPCSSF.R_chol = R_chol;

% frequency deviation penalty
qy = 100;
q = diag(vars.plant.C(1,:)*qy);
Q = diag(qy*ones(1,Np));
% Q = kron(eye(Np),q);
Q_chol = chol(Q);
vars.w.SMPCSSF.Q_chol = Q_chol;
k = dlqr(vars.plant.A0,vars.plant.B0,q,r);
for i = 1:N
    % sampling N number of instances of model parameters
    T_gov_i = (2*rand-1)*(percentage/100)*T_gov0 + T_gov0;
    T_tur_i = (2*rand-1)*(percentage/100)*T_tur0 + T_tur0;
    T_wtg_i = (2*rand-1)*(percentage/100)*T_wtg0 + T_wtg0;
```

```
    T_bess_i = (2*rand-1)*(percentage/100)*T_bess0 + T_bess0;
    D_i = (2*rand-1)*(percentage/100)*D0 + D0;
    M_i = (2*rand-1)*(percentage/100)*M0 + M0;
    Cbat = (2*rand-1)*(percentage/100)*Cbat0 + Cbat0;
    cw = (2*rand-1)*(percentage/100)*cw0 + cw0;
    cr = (2*rand-1)*(percentage/100)*cr0 + cr0;
    Hgov_i = tf([1],[T_gov_i 1]);
    Htur_i = tf([1],[T_tur_i 1]);
    Hwtg_i = tf([1],[T_wtg_i 1]);
    Hbess_i = tf([1],[T_bess_i 1]);
    Hmg_i = tf([1],[M_i D_i]);
    [A_i, B_i, H_i] = ...
        tf2ss(Hgov_i,Htur_i,Hwtg_i,Hbess_i,Hmg_i,cr,cw,vars.bat.n,Cbat);
    %% discretized model
    sys = ss(A_i,[B_i H_i],vars.plant.C,vars.plant.D);
    [dsys] = c2d(sys,dt);

    Ad_i = dsys.A;
    Bd_i = dsys.B(:,1:By);
    Hd_i = dsys.B(:,By+1:end);

    Ad_bat_i = Ad_i(end-1:end,end-1:end);
    Bd_bat_i = Bd_i(end-1:end,end);

    ABK_i = Ad_i-Bd_i*k;

    [A_e_bat_i, B_0_bat_i, B_e_bat_i,¬] = RecelDeltaU(Ad_bat_i,Bd_bat_i,[]);
    A_e_bat(Abx*Np*(i-1)+1:Abx*Np*i,:) = A_e_bat_i;
    B_0_bat(Bbx*Np*(i-1)+1:Bbx*Np*i,:) = B_0_bat_i;
    B_e_bat(:,Bby*Nc*(i-1)+1:Bby*Nc*i) = B_e_bat_i;

    [A_e_i, B_0_i, B_e_i, H_e_i] = RecelDeltaU(Ad_i,Bd_i,Hd_i);
    A_e(Ax*Np*(i-1)+1:Ax*Np*i,:) = A_e_i;
    B_0(Bx*Np*(i-1)+1:Bx*Np*i,:) = B_0_i;
    B_e(:,By*Nc*(i-1)+1:By*Nc*i) = B_e_i;
    H_e(:,Hy*Np*(i-1)+1:Hy*Np*i) = H_e_i;

    [SFA_e_i, SFB_0_i, SFB_e_i, SFH_e_i] = RecelDeltaU(ABK_i,Bd_i,Hd_i);
    SFA_e(Ax*Np*(i-1)+1:Ax*Np*i,:) = SFA_e_i;
    SFB_0(Bx*Np*(i-1)+1:Bx*Np*i,:) = SFB_0_i;
    SFB_e(:,By*Nc*(i-1)+1:By*Nc*i) = SFB_e_i;
    SFH_e(:,Hy*Np*(i-1)+1:Hy*Np*i) = SFH_e_i;
end
C_e_bat = kron(eye(Np),ones(1,Bbx));
C_e = kron(eye(Np),dsys.C);

vars.plant.A_e = A_e;
vars.plant.B_0 = B_0;
vars.plant.B_e = B_e;
vars.plant.C_e = C_e;
vars.plant.H_e = H_e;

vars.plant.SFA_e = SFA_e;
vars.plant.SFB_0 = SFB_0;
vars.plant.SFB_e = SFB_e;
vars.plant.SFH_e = SFH_e;

vars.plant.A_e_bat = A_e_bat;
vars.plant.B_0_bat = B_0_bat;
vars.plant.B_e_bat = B_e_bat;
vars.plant.C_e_bat = C_e_bat;
vars.LQR.k = k;
%% constraints
vars.cstr.batCap = 1;
vars.cstr.max_u1 = 1;
vars.cstr.min_u1 = -1;
vars.cstr.max_u2 = 1;
vars.cstr.min_u2 = -1;
vars.cstr.max_du1 = 0.1;
vars.cstr.min_du1 = -0.1;
vars.cstr.max_du2 = 0.2;
vars.cstr.min_du2 = -0.2;
vars.cstr.max_f = 0.2;
vars.cstr.min_f = -0.2;
vars.cstr.max_SOC = 0.2;
%% generate wind samples
%run Wind speed generator initialization script
% ConstantWindSpeed;

T_wtg0 = vars.plant.T_wtg0;
T_wtg = (2*rand-1)*(percentage/100)*T_wtg0 + T_wtg0;
Hwtg = tf([1],[T_wtg 1]);
vars.Tf.d.Hdwtg = c2d(Hwtg,dt);

[W,Wmu] = generateWindSample(vars.SCMPC.N,vars.sim.Np,vars.sim.dt);
```

```
vars.SCMPC.W = W;
vars.wind.Wmu = Wmu;
out = sim('WindGen19a',simTime);
out.windpower.data(2:end) = out.windpower.data(2:end) - ...
    vars.wind.Wmu*ones(length(out.windpower.data(2:end)),1);
%% calculate Hinfinity controller
Time = toc;
fprintf('InitSim took %d seconds to run\n',Time);
```

B.2 DMPC

```
function [controllerOut] = DMPCqpCVX(input)
tic
vars = evalin('base', 'vars');
ref = input(1);
SOC_r = input(2);
Load = input(3);
x0 = input(4:end);
dt = vars.sim.dt;
Np = vars.sim.Np;
Nc = vars.sim.Nc;
N = vars.SCMPC.N;
W = zeros(Np,1);
%% state space model
% plant matricies
A = vars.plant.A0;
B = vars.plant.B0;
H = vars.plant.H0;
C = vars.plant.C;
D = vars.plant.D;
sys = ss(A,[B H],C,D);
[dsys] = c2d(sys,dt);
Ad = dsys.A;
Bd = dsys.B(:,1:vars.dims.By);
Hd = dsys.B(:,vars.dims.By+1:end);
[A_e, B_0, B_e, H_e] = RecelDeltaU(Ad,Bd,Hd);
C_e = vars.plant.C_e;

Ad_bat = Ad(end-1:end,end-1:end);
Bd_bat = Bd(end-1:end,end);
[A_e_bat, B_0_bat, B_e_bat,¬] = RecelDeltaU(Ad_bat,Bd_bat,[]);
C_e_bat = vars.plant.C_e_bat;
%% initialization of previous states
Abx = vars.dims.Abx;
Bx = vars.dims.Bx;
By = vars.dims.By;
Hy = vars.dims.Hy;
persistent prev_u prev_x;
if isempty(prev_u)
    prev_u = zeros(By,1);
end
if isempty(prev_x)
   prev_x = zeros(nStates,1);
   prev_x(end-1) = vars.bat.Xinit;
   prev_x(end) = vars.bat.Xinit;
end
ΔX = x0 - prev_x;
prev_x = x0;

%% solver cvx
% weights
% control input Δ u
r = [200 0;0 50];
R = kron(eye(Nc),r);
R_chol = chol(R);

% frequency deviation penalty
q = [100 100 100 100 100 100 100 100 100 100];
Q = diag(q);
% Q = kron(eye(Np),q);
Q_chol = chol(Q);

% SOC deviation penalty
L = 2;
L = kron(eye(Np),L);
L_chol = chol(L);

% SOC fluctuation penalty(standard deviation)
M = 1;
% saturation limits on u and battery capacity
batCap = vars.cstr.batCap;

```

```
max_u1 = vars.cstr.max_u1;
min_u1 = vars.cstr.min_u1;
max_u2 = vars.cstr.max_u2;
min_u2 = vars.cstr.min_u2;

max_du1 = vars.cstr.max_du1;
min_du1 = vars.cstr.min_du1;
max_du2 = vars.cstr.max_du2;
min_du2 = vars.cstr.min_du2;
max_f = vars.cstr.max_f;
min_f = vars.cstr.min_f;
max_SOC = vars.cstr.max_SOC;

x0_bat = [x0(end-1);x0(end)];
u0_bat = prev_u(2);

cvx_begin quiet
    variable U(Nc*By,1);
    expression J;
    U1 = U(1:2:end,:);
    U2 = U(2:2:end,:);
    LOAD = -Load*ones(10,1);
    Wlarge = reshape([W';LOAD'],size(W,1)+size(LOAD,1),[]);
    x = A_e*x0 + B_0*prev_u + B_e*U + H_e*Wlarge;

    % cost of Δ u
    termi = sum_square(R_chol*U);
    J = J + termi;
    % cost of frequency deviation
    freqDev = C_e(1:2:end,:)*x - ref*ones(Np,1);
    termi = sum_square(Q_chol*freqDev);
    J = J + termi;
    % cost of SOC deviation
     xbat = A_e_bat*x0_bat + B_0_bat*u0_bat + B_e_bat*U2;
     SOC = C_e_bat*xbat;
     SOCdev = SOC - SOC_r*ones(Np,1);
     termi = sum_square(L_chol*SOCdev);
     J = J + termi;

     % battery degradation: standard deviation
    SOCstd = std(SOC);
    J = J + M*sum(SOCstd);
    minimize(J)
    subject to
        % du saturation limits -0.5 < du < 0.5
        min_du1*ones(Nc*By,1) ≤ U ≤ max_du1*ones(Nc*By,1);
%         % u saturation limits -1 < u < 1
        for j = 1:Nc
            if mod(j,2) == 1
                min_u1 ≤ prev_u(1) + sum(U(1:2:j)) ≤ max_u1;
            elseif mod(j,2) == 0
                min_u2 ≤ prev_u(2) + sum(U(2:2:j)) ≤ max_u2;
            end
        end
        % battery saturation limits 0.2 < SOC < 0.8
        SOC = C_e_bat*xbat;
        max_SOC*ones(Np,1) ≤ SOC ≤ (1-max_SOC)*ones(Np,1);

        % frequency deviation limits  -0.2 < freqDev < 0.2
        min_f*ones(Np,1) ≤ freqDev ≤ max_f*ones(Np,1);

cvx_end

ΔU = [U(1);U(2)];
prev_u = prev_u + ΔU;
time = toc;
controllerOut = [prev_u;cvx_optval;time];
end
```

B.3 SMPC FP

```
function [controllerOut] = SMPCqpSFPCVX(input)
tic
vars = evalin('base', 'vars');
ref = input(1);
SOC_r = input(2);
Load = input(3);
x0 = input(4:end);
dt = vars.sim.dt;
Np = vars.sim.Np;
Nc = vars.sim.Nc;
N = vars.SCMPC.N;
```

```
W = vars.SCMPC.W;

%% state space model
% plant matricies
A_e = vars.plant.A_e;
B_0 = vars.plant.B_0;
B_e = vars.plant.B_e;
H_e = vars.plant.H_e;
C_e = vars.plant.C_e;

%% initialization of previous states
Abx = vars.dims.Abx;
Bx = vars.dims.Bx;
By = vars.dims.By;
Hy = vars.dims.Hy;
persistent prev_u prev_x;
if isempty(prev_u)
    prev_u = zeros(By,1);
end
if isempty(prev_x)
   prev_x = zeros(Bx,1);
   prev_x(end-1) = vars.bat.Xinit;
   prev_x(end) = vars.bat.Xinit;
end
ΔX = x0 - prev_x;
prev_x = x0;

%% solver cvx
% weights
% control input Δ u
r = [200 0;0 50];
R = kron(eye(Nc),r);
R_chol = chol(R);

% frequency deviation penalty
q = [100 100 100 100 100 100 100 100 100 100];
Q = diag(q);
% Q = kron(eye(Np),q);
Q_chol = chol(Q);

% SOC deviation penalty
L = 2;
L = kron(eye(Np),L);
L_chol = chol(L);

% SOC fluctuation penalty(standard deviation)
M = 1;

% saturation limits on u and battery capacity
batCap = vars.cstr.batCap;

max_u1 = vars.cstr.max_u1;
min_u1 = vars.cstr.min_u1;
max_u2 = vars.cstr.max_u2;
min_u2 = vars.cstr.min_u2;

max_du1 = vars.cstr.max_du1;
min_du1 = vars.cstr.min_du1;
max_du2 = vars.cstr.max_du2;
min_du2 = vars.cstr.min_du2;
max_f = vars.cstr.max_f;
min_f = vars.cstr.min_f;
max_SOC = vars.cstr.max_SOC;



W_I = mat2cell(W,size(W,1),ones(1,size(W,2)));
W_I = blkdiag(W_I{:});
cvx_begin quiet
    variable Gamma(Nc*By,1);
    variable Theta(Nc*By,Np);
    expression J;
    U = kron(ones(1,N),Gamma);
    U = U + Theta*W;
    U_I = kron(eye(N),Gamma);
    temp = kron(eye(N),Theta);
    temp = temp*W_I;
    U_I = U_I + temp;
    x = A_e*x0 + B_0*prev_u;
    X = reshape(x,Bx*Np,N);


    LOAD = -Load*ones(Np,N);
    % merge W and LOAD every second row
    Wlarge = [W;LOAD];
```

```
97      oddI = 1:2:size(Wlarge,1);
98      evenI = 2:2:size(Wlarge,1);
99      [¬,reorderI] = sort([oddI,evenI]);
100     Wlarge = Wlarge(reorderI,:);
101
102     % make diagonal matrices with columns on diagonal
103     Wlarge_I = mat2cell(Wlarge,size(Wlarge,1),ones(1,size(Wlarge,2)));
104     Wlarge_I = blkdiag(Wlarge_I{:});
105
106     X = X + B_e*U_I + H_e*Wlarge_I;
107
108     % cost of Δ u
109     termi = sum(sum_square(R_chol*U));
110     J = J + termi/N;
111     % cost of frequency deviation
112     freqDev = C_e(1:2:end,:)*X - ref*ones(Np,N);
113     termi = sum(sum_square(Q_chol*freqDev));
114     J = J + termi/N;
115     % cost of SOC deviation
116      SOC = C_e(2:2:end,:)*X;
117      SOCdev = SOC - SOC_r*ones(Np,N);
118      termi = sum(sum_square(L_chol*SOCdev));
119      J = J + termi/N;
120
121      % battery degradation: standard deviation
122     SOCstd = std(SOC);
123     J = J + M*sum(SOCstd)/N;
124     minimize(J)
125
126     subject to
127         % theta parametrisation constraints
128         for i = 1:Nc
129             for j = 1:Np
130                 if j ≥ i
131                     Theta(By*(i-1)+1:By*i,j) == zeros(By,1);
132                 end
133             end
134         end
135         % du saturation limits -0.5 < du < 0.5
136         min_du1*ones(Nc*By,N) ≤ U ≤ max_du1*ones(Nc*By,N);
137 %         % u saturation limits -1 < u < 1
138         for j = 1:Nc
139             if mod(j,2) == 1
140                 min_u1 ≤ prev_u(1) + sum(U(1:2:j)) ≤ max_u1;
141             elseif mod(j,2) == 0
142                 min_u2 ≤ prev_u(2) + sum(U(2:2:j)) ≤ max_u2;
143             end
144         end
145         % battery saturation limits 0.2 < SOC < 0.8
146         max_SOC*ones(Np,N) ≤ SOC ≤ (1-max_SOC)*ones(Np,N);
147         % frequency deviation limits  -0.15 < freqDev < 0.15
148         min_f*ones(Np,N) ≤ freqDev ≤ max_f*ones(Np,N);
149 cvx_end
150 if strcmp(cvx_status,'Infeasible')
151     ΔU = zeros(By,1);
152     cvx_optval = 100;
153     disp('Infeasibility reached, output is 0')
154 else
155     ΔU = [Gamma(1);Gamma(2)];
156 end
157 prev_u = prev_u + ΔU;
158 time = toc;
159 controllerOut = [prev_u;cvx_optval;time];
160 end
```

B.4 SMPC SF

```
1  function [controllerOut] = SMPCqpSSFCVX(input)
2  tic
3  vars = evalin('base', 'vars');
4  ref = input(1);
5  SOC_r = input(2);
6  Load = input(3);
7  x0 = input(4:end);
8  dt = vars.sim.dt;
9  Np = vars.sim.Np;
10 Nc = vars.sim.Nc;
11 N = vars.SCMPC.N;
12 W = vars.SCMPC.W;
13 k = vars.LQR.k;
14 %% state space model
15 % plant matricies
```

```
A_e = vars.plant.SFA_e;
B_0 = vars.plant.SFB_0;
B_e = vars.plant.SFB_e;
H_e = vars.plant.SFH_e;
C_e = vars.plant.C_e;

%% initialization of previous states
Bx = vars.dims.Bx;
By = vars.dims.By;
Hy = vars.dims.Hy;
Abx = vars.dims.Abx;
persistent prev_u prev_x;
if isempty(prev_u)
    prev_u = zeros(By,1);
end
if isempty(prev_x)
   prev_x = zeros(Bx,1);
   prev_x(end-1) = vars.bat.Xinit;
   prev_x(end) = vars.bat.Xinit;
end
dx = x0 - prev_x;
prev_x = x0;

%% solver cvx
% weights
R_chol = vars.w.SMPCSSF.R_chol;
Q_chol = vars.w.SMPCSSF.Q_chol;

% SOC deviation penalty
L = 4;
L = kron(eye(Np),L);
L_chol = chol(L);

% SOC fluctuation penalty(standard deviation)
M = 1;
% saturation limits on u and battery capacity
batCap = vars.cstr.batCap;

max_u1 = vars.cstr.max_u1;
min_u1 = vars.cstr.min_u1;
max_u2 = vars.cstr.max_u2;
min_u2 = vars.cstr.min_u2;

max_du1 = vars.cstr.max_du1;
min_du1 = vars.cstr.min_du1;
max_du2 = vars.cstr.max_du2;
min_du2 = vars.cstr.min_du2;
max_f = vars.cstr.max_f;
min_f = vars.cstr.min_f;
max_SOC = vars.cstr.max_SOC;


x0_bat = [x0(end-1);x0(end)];
u0_bat = prev_u(2);

cvx_begin quiet
    variable Gamma(Nc*By,1);

    expression J;
    x = A_e*x0 + B_0*prev_u;
    X = reshape(x,Bx*Np,N);

    Gamma_I = kron(eye(N),Gamma);
    LOAD = -Load*ones(Np,N);
    Wlarge = [W;LOAD];
    oddI = 1:2:size(Wlarge,1);
    evenI = 2:2:size(Wlarge,1);
    [¬,reorderI] = sort([oddI,evenI]);
    Wlarge = Wlarge(reorderI,:);

    % make diagonal matrices with columns on diagonal
    Wlarge_I = mat2cell(Wlarge,size(Wlarge,1),ones(1,size(Wlarge,2)));
    Wlarge_I = blkdiag(Wlarge_I{:});

    X = X + B_e*Gamma_I + H_e*Wlarge_I;

    U = kron(ones(1,N),Gamma);
    K = kron(eye(Nc),k);
    U = U - K*X(1:Nc*Bx,:);

    % cost of Δ u
    termi = sum(sum_square(R_chol*U));
    J = J + termi/N;
    % cost of frequency deviation
    freqDev = C_e(1:2:end,:)*X - ref*ones(Np,N);
```

```
    termi = sum(sum_square(Q_chol*freqDev));
    J = J + termi/N;
    % cost of SOC deviation
     SOC = C_e(2:2:end,:)*X;
     SOCdev = SOC - SOC_r*ones(Np,N);
     termi = sum(sum_square(L_chol*SOCdev));
     J = J + termi/N;

     % battery degradation: standard deviation
    SOCstd = std(SOC);
    J = J + M*sum(SOCstd)/N;

    minimize(J)
    subject to
        % du saturation limits -0.1 < du < 0.1
        min_du1*ones(Nc*By,N) ≤ U ≤ max_du1*ones(Nc*By,N);
        % u saturation limits -1 < u < 1
        for j = 1:Nc
            if mod(j,2) == 1
                min_u1 ≤ prev_u(1) + sum(U(1:2:j)) ≤ max_u1;
            elseif mod(j,2) == 0
                min_u2 ≤ prev_u(2) + sum(U(2:2:j)) ≤ max_u2;
            end
        end
        % battery saturation limits 0.2 < SOC < 0.8
        max_SOC*ones(Np,N) ≤ SOC ≤ (1-max_SOC)*ones(Np,N);

        % frequency deviation limits  -0.2 < freqDev < 0.2
        min_f*ones(Np,N) ≤ freqDev ≤ max_f*ones(Np,N);
cvx_end
if strcmp(cvx_status,'Infeasible')
    ΔU = zeros(By,1);
    cvx_optval = 100;
    disp('Infeasibility reached, output is 0')
else
    ΔU = [Gamma(1);Gamma(2)] - k*x0;
end
prev_u = prev_u + ΔU;
time = toc;
controllerOut = [prev_u;cvx_optval;time];
end
```

B.5 SMPC NP

```
function [controllerOut] = SMPCqpSNPCVX(input)
tic
vars = evalin('base', 'vars');
ref = input(1);
SOC_r = input(2);
Load = input(3);
x0 = input(4:end);
dt = vars.sim.dt;
Np = vars.sim.Np;
Nc = vars.sim.Nc;
N = vars.SCMPC.N;
W = vars.SCMPC.W;
%% state space model
% plant matricies
A_e = vars.plant.A_e;
B_0 = vars.plant.B_0;
B_e = vars.plant.B_e;
H_e = vars.plant.H_e;
C_e = vars.plant.C_e;
%% initialization of previous states
Abx = vars.dims.Abx;
Bx = vars.dims.Bx;
By = vars.dims.By;
Hy = vars.dims.Hy;
persistent prev_u prev_x;
if isempty(prev_u)
    prev_u = zeros(By,1);
end
if isempty(prev_x)
   prev_x = zeros(Bx,1);
   prev_x(end-1) = vars.bat.Xinit;
   prev_x(end) = vars.bat.Xinit;
end
ΔX = x0 - prev_x;
prev_x = x0;

%% solver cvx
% weights
```

```
% control input Δ u
r = [200 0;0 50];
R = kron(eye(Nc),r);
R_chol = chol(R);

% frequency deviation penalty
q = [100 100 100 100 100 100 100 100 100 100];
Q = diag(q);
% Q = kron(eye(Np),q);
Q_chol = chol(Q);

% SOC deviation penalty
L = 2;
L = kron(eye(Np),L);
L_chol = chol(L);

% SOC fluctuation penalty(standard deviation)
M = 1;

% saturation limits on u and battery capacity
batCap = vars.cstr.batCap;

max_u1 = vars.cstr.max_u1;
min_u1 = vars.cstr.min_u1;
max_u2 = vars.cstr.max_u2;
min_u2 = vars.cstr.min_u2;

max_du1 = vars.cstr.max_du1;
min_du1 = vars.cstr.min_du1;
max_du2 = vars.cstr.max_du2;
min_du2 = vars.cstr.min_du2;
max_f = vars.cstr.max_f;
min_f = vars.cstr.min_f;
max_SOC = vars.cstr.max_SOC;


W_I = mat2cell(W,size(W,1),ones(1,size(W,2)));
W_I = blkdiag(W_I{:});
cvx_begin quiet
    variable Gamma(Nc*By,1);
    expression J;
    U = kron(ones(1,N),Gamma);
    U_I = kron(eye(N),Gamma);
    x = A_e*x0 + B_0*prev_u;
    X = reshape(x,Bx*Np,N);
    LOAD = -Load*ones(Np,N);
    % merge W and LOAD every second row
    Wlarge = [W;LOAD];
    oddI = 1:2:size(Wlarge,1);
    evenI = 2:2:size(Wlarge,1);
    [¬,reorderI] = sort([oddI,evenI]);
    Wlarge = Wlarge(reorderI,:);

    % make diagonal matrices with columns on diagonal
    Wlarge_I = mat2cell(Wlarge,size(Wlarge,1),ones(1,size(Wlarge,2)));
    Wlarge_I = blkdiag(Wlarge_I{:});

    X = X + B_e*U_I + H_e*Wlarge_I;

    % cost of Δ u
    termi = sum(sum_square(R_chol*U));
    J = J + termi/N;
    % cost of frequency deviation
    freqDev = C_e(1:2:end,:)*X - ref*ones(Np,N);
    termi = sum(sum_square(Q_chol*freqDev));
    J = J + termi/N;
    % cost of SOC deviation
     SOC = C_e(2:2:end,:)*X;
     SOCdev = SOC - SOC_r*ones(Np,N);
     termi = sum(sum_square(L_chol*SOCdev));
     J = J + termi/N;

     % battery degradation: standard deviation
    SOCstd = std(SOC);
    J = J + M*sum(SOCstd)/N;
    minimize(J)

    subject to
        % du saturation limits -0.5 < du < 0.5
        min_du1*ones(Nc*By,N) ≤ U ≤ max_du1*ones(Nc*By,N);
%         % u saturation limits -1 < u < 1
        for j = 1:Nc
            if mod(j,2) == 1
                min_u1 ≤ prev_u(1) + sum(U(1:2:j)) ≤ max_u1;
            elseif mod(j,2) == 0
```

```
                min_u2 ≤ prev_u(2) + sum(U(2:2:j)) ≤ max_u2;
            end
        end
        % battery saturation limits 0.2 < SOC < 0.8
        max_SOC*ones(Np,N) ≤ SOC ≤ (1-max_SOC)*ones(Np,N);
        % frequency deviation limits  -0.15 < freqDev < 0.15
        min_f*ones(Np,N) ≤ freqDev ≤ max_f*ones(Np,N);
cvx_end
if strcmp(cvx_status,'Infeasible')
    ΔU = zeros(By,1);
    cvx_optval = 100;
    disp('Infeasibility reached, output is 0')
else
    ΔU = [Gamma(1);Gamma(2)];
end
prev_u = prev_u + ΔU;
time = toc;
controllerOut = [prev_u;cvx_optval;time];
end
```

B.6 Mixed synthesis $H\infty$

B.6.1 10% parametric uncertainty

```
% infinity loop shaping
clear logsout SMPCqpSFPCVX SMPCqpSSFCVX DMPCqpCVX batModel SMPCqpSNPCVX ...
    Hinfinity2;
close all
A = vars.plant.A0;
B = vars.plant.B0;
H = vars.plant.H0;
C = vars.plant.C;

D = vars.plant.D;

sys = ss(A,B,C,D);
Gnom = tf(sys);
sys2 = ss(A,H,C,D);
Gd = tf(sys2);

s = tf('s');
Wp11inv = makeweight(3.16*10^(-5),[10^(-2.0) 0.1],1.5);
Wp12inv = makeweight(0.00001,[0.05 0.1],1.5);
Wp21inv = makeweight(3.16*10^(-5),[10^(-2.0) 0.1],1.5);
Wp22inv = makeweight(0.0001,[0.05 1],3.16);
Wp11 = 1/Wp11inv;
Wp12 = 1/Wp12inv;
Wp21 = 1/Wp21inv;
Wp22 = 1/Wp22inv;
Wp = [Wp11 Wp12;Wp21 Wp22];

Wu = [1 0.5;1 1];


Wd11 = makeweight(0.1,[1 0.178],0.316);
Wd12 = makeweight(10^(-10),[0.001 10^(-5)],1);
Wd21 = makeweight(0.1,[1 0.316],10);
Wd22 = makeweight(0.1,[10 0.316],1);

Wd = [Wd11 Wd12;Wd21 Wd22];

[K1,CL,gamma] = mixsyn(Gnom,Wp,[],Wd);

Kd1 = c2d(K1,vars.sim.dt);
vars.Hinf.A = Kd1.A;
vars.Hinf.B = Kd1.B;
vars.Hinf.C = Kd1.C;
vars.Hinf.D = Kd1.D;
```

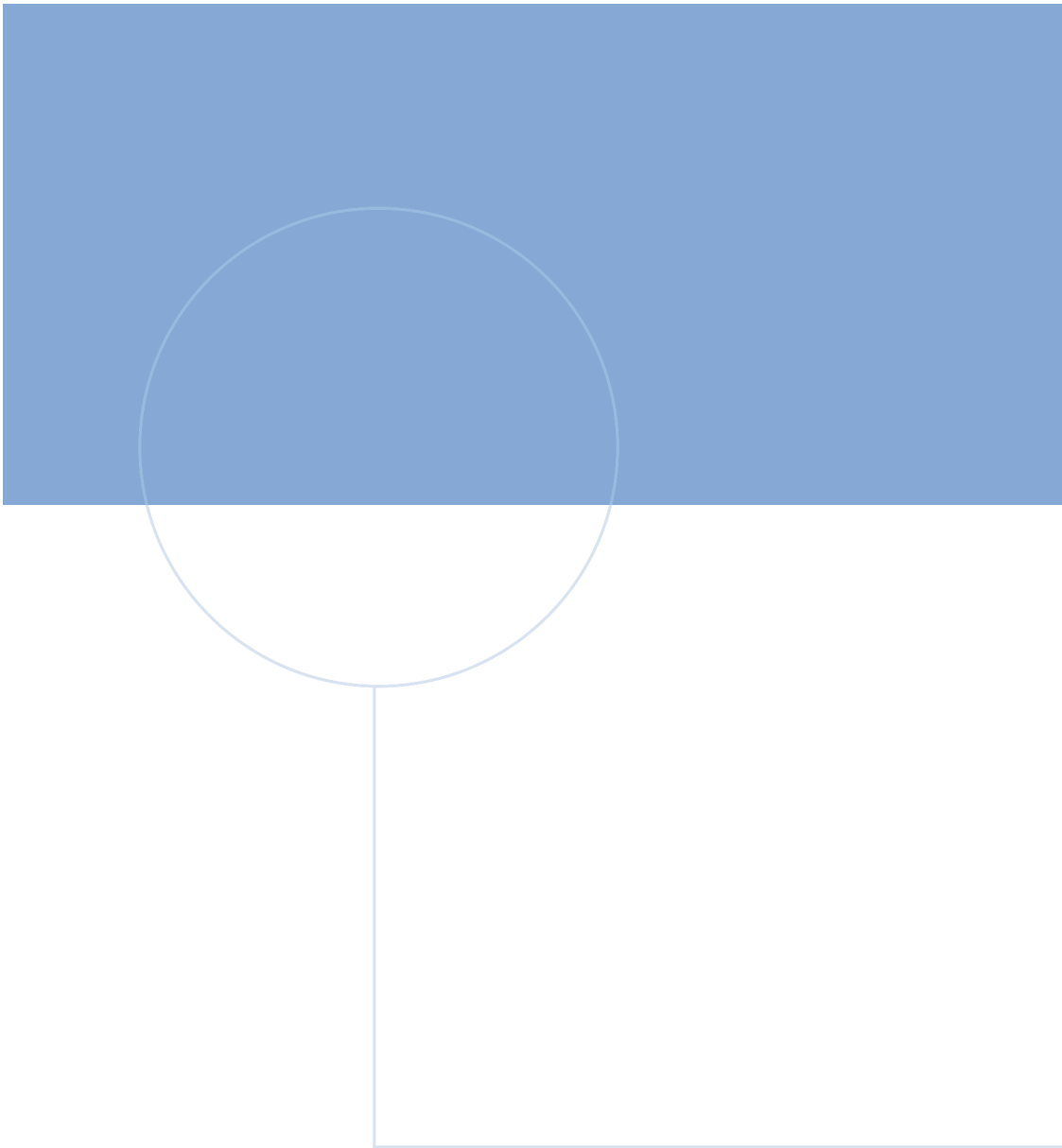
B.6.2 50% parametric uncertainty

```
% infinity loop shaping
clear logsout SMPCqpSFPCVX SMPCqpSSFCVX DMPCqpCVX batModel SMPCqpSNPCVX ...
    Hinfinity2;
close all
A = vars.plant.A0;
B = vars.plant.B0;
H = vars.plant.H0;
C = vars.plant.C;
```

```
8
9  D = vars.plant.D;
10
11 sys = ss(A,B,C,D);
12 Gnom = tf(sys);
13 sys2 = ss(A,H,C,D);
14 Gd = tf(sys2);
15
16 s = tf('s');
17 Wp11inv = makeweight(3.16*10^(-5),[10^(-2.25) 0.1],1.5);
18 Wp12inv = makeweight(0.00001,[0.008 0.1],1.5);
19 Wp21inv = makeweight(3.16*10^(-5),[10^(-2.25) 0.1],1.5);
20 Wp22inv = makeweight(0.0001,[0.008 1],3.16);
21 Wp11 = 1/Wp11inv;
22 Wp12 = 1/Wp12inv;
23 Wp21 = 1/Wp21inv;
24 Wp22 = 1/Wp22inv;
25 Wp = [Wp11 Wp12;Wp21 Wp22];
26
27 Wu = [1 0.5;1 1];
28
29
30 Wd11 = makeweight(0.1,[5 0.178],0.316);
31 Wd12 = makeweight(10^(-10),[0.0005 10^(-5)],1);
32 Wd21 = makeweight(0.1,[5 0.316],10);
33 Wd22 = makeweight(0.1,[5 0.316],1);
34
35 Wd = [Wd11 Wd12;Wd21 Wd22];
36
37 [K1,CL,gamma] = mixsyn(Gnom,Wp,[],Wd);
38
39 Kd1 = c2d(K1,vars.sim.dt);
40 vars.Hinf.A = Kd1.A;
41 vars.Hinf.B = Kd1.B;
42 vars.Hinf.C = Kd1.C;
43 vars.Hinf.D = Kd1.D;
```

C BESS implementation code

```
1  function [batOut] = batModel(Pbat)
2  vars = evalin('base', 'vars');
3  persistent prev_x;
4  if isempty(prev_x)
5      prev_x = ones(2,1)*vars.bat.Xinit;
6  end
7  dt = vars.sim.dt; % timestep
8  %%
9  A = vars.bat.ss.A;
10 B = vars.bat.ss.B;
11 C = vars.bat.ss.C;
12 D = 0;
13 sys = ss(A,B,C,D);
14 dsys = c2d(sys, dt);
15 Ad = dsys.A;
16 Bd = dsys.B;
17 x = Ad*prev_x + Bd*Pbat;
18 %% SOC constraint
19 SOC = x(1) + x(2);
20 if (SOC ≤ 1 && SOC ≥ 0)
21     prev_x = x;
22     inSOC = 1;
23 else
24     x = prev_x;
25     inSOC = 0;
26 end
27 batOut = [inSOC;x];
28 end
```



NTNU

Norwegian University of
Science and Technology