UNIVERSITY OF GRONINGEN

Distributed Model Predictive Control in the DC Power Network

by

M.A.S. Apel

A thesis presented for the degree of Master of Science

Supervisors:

prof. dr. ir. J.M.A. Scherpen

dr. G.K.H. Larsen

dr. M. Cucuzzella

dr. K.C. Kosaraju

Industrial Engineering and Management

Faculty of Science and Engineering

April 2019

"The way you learn anything is that something fails, and you figure out how not to have it fail again."

Robert S. Arrighi

Abstract

DC microgrids are increasingly utilized, because they have favourable characteristics over AC networks. The orientation of this research is to establish a distributed model predictive control based consensus algorithm for current sharing in DC microgrids. In centralized MPC control approaches for the DC microgrids the dynamics are coupled, while currents flow between the nodes. In large-scale networks centralized control is considered infeasible, nonscalable, too costly or too fragile due to the fact that every control action is executed by one controller. Therefore the establishment of a distributed MPC scheme for in the DC microgrid is essential. In this research the dynamics are decoupled by means of dual decomposition and subgradient iterations to reach a distributed formulation. Wherein each local controller, present at each node, solves its own subproblem, solely based on local information. Together they will arrive to the solution of the original problem, but without solving a centralized MPC problem. The proposed controllers also ensure that both the buck converter output voltages as well as the load voltages remain within acceptable bounds. The dual decomposition is performed on the physical system, as the DGUs share currents with each other. In previous research, dual decomposition was usually performed on a simple matrix, ensuring relatively steady convergence properties for the subgradient algorithm. In this research it is performed on a more complex coupling matrix representing the physical properties of the DC microgrid. As a result, it was discovered that the convergence behaviour of the subgradient algorithms was greatly affected by the configuration of the system in the optimization. Considering the effects of these parameters, it has been observed that MPC via dual decomposition and gradient iterations is a suitable design approach for reaching consensus in the DC microgrid in a distributed manner.

Contents

List of Figures

Nomenclature

Alphabetical

$\mathcal{B}$	Incidence matrix
$\mathcal{C}$	Set of edges
$\mathcal{D}$	Domain of a set of points
$\mathcal{G}$	Connected and undirected graph
$\mathcal{K}$	Set of horizon steps
$\mathcal{L}$	Laplacian matrix
$\mathcal{N}_i$	Set of neighbours of i
$\mathcal{Q}$	Weighted identity matrix
$\mathcal{R}$	Weighted identity matrix
$\mathcal{T}$	Set of time steps
$\mathcal{V}$	Set of nodes
A	State matrix
B	Input matrix
C	Output matrix
C_s	Shunt capacitance
d	Dual solution
e	Error vector
f_0	Convex objective function
f_i	Convex inequality constraint functions
g	Dual objective function
h_i	Linear equality functions
I	Line current
I_L	Unknown load demand

I_s Generated current

J Objective function

k Horizon step

L Lagrangian

L_s Filter inductance

m Number of lines

m_{avg} Moving average of the gradient

N Horizon window

n Number of nodes

p Primal solution

R Line resistance

r Reference signal vector

R_s Filter resistance

s Iteration step

S_k Iteration window

T Sampling time

t Time step

u Buck converter output voltage (input vector)

V Load voltage

v Auxiliary variable

W Weighing matrix

x State vector

y Output vector

w_i Weighting value for DGU_i

Greek

β Momentum term

ϵ Error

Γ Coefficient matrix

γ Step size or learning rate

λ Lagrangian multiplier

θ Lagrangian multiplier

Mathematical

$\{x(t)\}_{t=0}^{t=\infty}$ Sequence of $x(0), x(1), x(2), ...$

$\hat{x}$ Estimate of vector x

$\mathbb{1}_n$ Vector space of n-component with value one

$\mathbb{R}_{\geq \mathbb{0}}$ Positive real numbers

$\mathbb{R}$ Real numbers

$\mathbb{R}^n$ Vector space of n-component real vectors

$\mathbb{R}^{n \times m}$ Vector space of $n \times m$ real matrices

$\overline{x}$ Steady state of a vector x

x^T Transpose of a vector x

x^{-1} Inverse of a vector x

$x^\star$ Optimal value of x

x_{max} Maximum of vector x

x_{min} Minimum of vector x

Chapter 1

Introduction

Renewable energy is a practical, affordable solution to the world's electricity needs. The world production of renewable energy is increasing and is expected to grow even more in the short term future. This would lead to an energy transition from non-renewable energy sources to a more efficient, lower-carbon energy mix with energy production and consumption systems relying more on renewable energy sources (RES). It is predicted that renewable energy sources in 2050 will generate 50% of the U.S. electricity market [51]. Due to this wide diffusion of renewable energy sources and the active participation of consumers to the electricity market this would lead to [23], the power network needs to be adjusted to incorporate sustainable energy sources, which are unpredictable in their energy generation.

Power systems that can account for this uncertainty, that characterizes RES, are called microgrids. Microgrids are low-voltage electrical distribution networks consisting of clusters of loads, storage systems and distributed generation units (DGUs) that are interconnected through power lines. Generally, microgrids are either Alternating Current (AC) or Direct Current (DC) networks, where each type requires specific control strategies [50]. Renewable energy sources such as photovoltaic (PV) systems are DC based and need inverters to operate in an AC grid, which means that fewer conversion systems are required once incorporated into the DC grid [26]. Beyond the fact that renewable energy sources are better suited to DC grids, they have some other advantages over AC grids. While the conversion losses are reduced between the DC output sources and DC grids, the power system can be 15% more efficient than AC systems [25]. Furthermore in DC networks, no harmonics are present. Therefore the frequency synchronization as well as the reactive power [33] are not required to be taken into account. Also, DC distribution generation systems are highly reliable microgrids [24]. As a result of all these aforementioned factors, the importance of DC microgrids has increased. Consequently DC microgrids are attracting growing interest and receive much research attention.

Microgrids often require a consensus algorithm to optimize its operations while satisfying some inputs and state constraints. Generally, the two control objectives in DC microgrids are voltage regulation and current sharing. Coordination within the system is needed to ensure these objectives, so that voltages are around desired values and current is proportionally shared among the various sources [50]. In this research a DC microgrid consisting of buck converters is discussed where both the objectives are to be achieved. Various control approaches, which can be found in literature, are used to reach the consensus in the DC power network [10]. The unpredictable

behavior of the RES call for a suitable and robust control strategy. A control scheme that accounts for the occurrence of interactions between plant components is model predictive control (MPC). In this research a MPC strategy will be applied to obtain system-wide current sharing among DGUs and voltage regulation in the DC microgrid. MPC techniques use a prediction of a system model to establish an appropriate control response that minimizes a certain cost function over the prediction horizon in the presence of disturbances and constraints [52], [6]. After the optimization, the first control sample is implemented to the system. Whereafter the optimization is shifted to a further time step.

MPC and other control schemes can either be centralized or distributed. Centralized control relies on a central control entity which regulates the network through real-time feedback signals [57]. Due to the form of the communication network, as the number of DGUs increases, centralized control becomes impractical. The centralized controller cannot divulge all the information, causing the overall performance and efficiency to decrease [31]. Therefore to address this computational issue; distributed control schemes are often exploited in large-scale networks, in which the optimization is broken into smaller sub-problems and decision-making is performed locally. Allowing the network to consist of a considerable number of DGUs without the demanding communication requirements of centralized control schemes.

In this research the centralized MPC based consensus algorithm achieving current sharing in the DC microgrid proposed in [38] is distributed. Wherein each DGU has its own MPC controller to regulate the buck converter's output voltage (which is controllable) and can work cooperatively with connected nodes to ensure total current sharing in the network. Hereby achieving a degree of coordination among DGUs that are solving MPC problems with locally relevant variables, costs and constraints, but without solving a centralized MPC problem. It is crucial that the MPC controllers account for the interactions between the subsystems to determine the optimal response. As the subsystems communicate with each other, the coupling between them is enforced in each optimization. Furthermore the proposed MPC scheme generates a prediction of future behaviour, which if appropriately integrated will improve the overall system performance [54],[7]. Consequently, due to its ability to generate a prediction of future subsystem behavior, MPC is chosen over other controller paradigms. If the likely influence of interconnected subsystems is known, each local controller can possibly determine suitable feedback action that accounts for these external influences.

Chapter 2

Research Approach

2.1 Problem Definition

DC based networks are increasingly interesting due to their advantageous characteristics with respect to AC networks. The MPC control approach as proposed in [38] is very suitable to ensure both current sharing as well as load voltage regulation in the DC microgrids. However for large scale networks, centralized control would be infeasible. Therefore there is a need for a distributed MPC for the DC microgrid, where each node solves its own sub-problem, based only on local information, and together they arrive at the solution of the original problem. The combination between distributed MPC and the DC microgrid dynamics has not been researched. This rises the following problem:

Problem: A distributed MPC approach has not been used to ensure current sharing and voltage regulation, so it is not clear whether the use distributed MPC will work in a DC based network.

2.2 Goal

The problem is transformed into a research goal which sets the aim to reach in this thesis. The goal is formulated as:

Goal: Development of a distributed MPC algorithm that achieves proportional current sharing and voltage regulation in the DC microgrid and examining the effects of different parameters upon this system.

2.3 Research Questions

From the goal of this thesis, the main research question is created:

Main RQ: How to develop a distributive model predictive control algorithm that reaches consensus in the DC microgrid?

This research question is going to be answered according to the following sub-questions.

1. How to model the DC microgrid?
    - What are the dynamics of the DC microgrid?
    - What are the control objectives in the DC microgrid?
2. How to does optimization according to the MPC approach work?
3. How can MPC be applied as a centralized control algorithm for the DC microgrid?
    - What is centralized control?
    - What are the dynamics of a centralized MPC in the DC microgrid?
4. How can the centralized MPC for DC microgrids be decomposed to a distributed form?
    - What is distributed control?
    - How can dual decomposition decouple a centralized optimization?
    - How can the dual decomposition method be applied to the MPC in DC microgrids?
    - How can current sharing be distributed?
5. How do different variables, parameters and constraints in the optimization affect the distributed MPC?

2.4 Outline

The sub-questions are used as a set up to outline this research. *Chapter 3* introduces the dynamics of the DC microgrid and the to be reached objectives. This dynamical model and set of constraints is to be optimized, which is done by means of a MPC scheme. A basic understanding of MPC is provided in *Chapter 4*. Whereafter in *Chapter 5*, the MPC approach is applied to control the DC microgrid, explained in *Chapter 3*. When the centralized MPC scheme for DC microgrids is known, this has to be decomposed into a distributed formulation, this will be done in, *Chapter 6*. However firstly the mathematical techniques necessary to decompose such a system are explained. Thereafter a general MPC problem is decomposed and at last this theory is applied to the centralized MPC of *Chapter 5*. In *Chapter 7* different scenarios will be formulated to test the effects of different variables, parameters and constraints on the distributed MPC for DC microgrids. These tests provide different results and conclusions from these results will be drawn in *Chapter 8*. In *Chapter 9* recommendations for further research will be given.

Chapter 3

DC Network

Due to the to high efficiency, high reliability, and easy interconnection of renewable sources, DC microgrids have several advantages over AC microgrids. As already stated the DC microgrids are therefore gaining popularity and receive much research attention. The power networks are also progressively utilized for a range of applications. DC grids can operate with an appropriate control strategy disconnected from the grid, ensuring stability and voltage regulation [27]. This, together with the fact that RES are DC operated, make that DC microgrids serve perfectly for electrifying remote villages [21] e.g. on the Hawaiian island Moku o Lo'e (Coconut island) where an islanded DC microgrid has been deployed. The ability to act in isolation from the main grid and their high reliability, results in ships having on-board DC grids [22]. Furthermore DC microgrids are utilized in military bases [49], industrial facilities (Honda distribution center) [61], data centers [45] and universities [12].

Microgrids are defined as low-voltage electrical distribution networks, composed of DGUs, loads and storage systems interconnected through power lines. AC and DC microgrids have their own system dynamics and control strategies, for AC systems see e.g. [55], [8] and [29]. As there is no reactive power and harmonics present, DC microgrids are regarded to be more simple than their AC counterparts.

In this research a (Kron-reduced) buck converter based DC microgrid is considered, which will entail n DGUs that are connected through m resistive power lines (see *Figure 3.1*). The Kron reduced system is equivalent to the network in [60] and usually a Kron reduction of the system results in a simpler set of equations while providing the same physical relations between voltage and current at the DGUs. The energy source of each DGU is represented by a DC voltage source that supplies a local load through a DC-DC buck converter. This local DC load is connected to the Point of Common Coupling (PCC). The i-th node (DGU), according to Kirchhoff's current and voltage laws, has the dynamics given by

$$L_{s_i}\dot{I}_{s_i}(t) = -R_{s_i}I_{s_i}(t) - V_i(t) + u_i(t), \tag{3.1a}$$

$$C_{s_i}\dot{V}_{s_i}(t) = I_{s_i}(t) - I_{L_i} - \sum_{j\in\mathcal{N}_i} I_{ij}(t). \tag{3.1b}$$

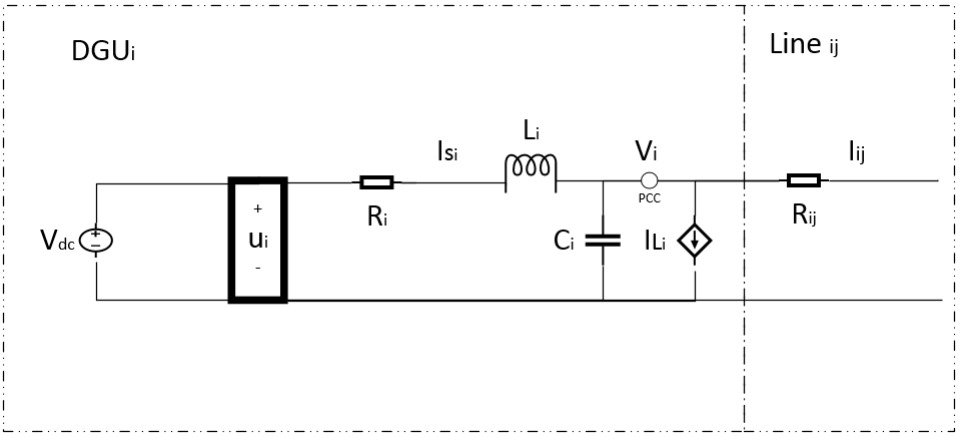

FIGURE 3.1: Electrical scheme of DGU i and line ij

Where $\mathcal{N}_i$ is the set of power lines connected to the i-th DGU, while $u_i(t)$ is the controllable output voltage of the i-th buck converter, which could be provided by a renewable energy source or a battery. Furthermore the current between DGUs j and i is represented by $I_{ij}(t)$ and its approximation, for each $j \in \mathcal{N}_i$, is given by

$$I_{ij}(t) = \frac{1}{R_{ij}}(V_i(t) - V_j(t)). \tag{3.2}$$

Symbol	Description
	Parameters
R_{s_i}	Filter resistance
L_{s_i}	Filter inductance
C_{s_i}	Shunt capacitor
R_{ij}	Line resistance
	State variables
I_{s_i}	Generated current
V_i	Load voltage
I_{ij}	Line current
	Inputs
u_i	Buck converter output voltage
I_{L_i}	Load demand

TABLE 3.1: Description of symbols

All the remaining symbols are outlined in *Table 3.1.* To formulate a compact notation for the DC microgrid dynamics of (3.1), the network topology is to be defined. The network is represented by a connected and undirected graph, $\mathcal{G} = (\mathcal{V}, \mathcal{C})$, where the set of nodes, $\mathcal{V} = \{0, ..., n\}$, and the set of edges, $\mathcal{C} = \{0, ..., m\}$, indicate the DGUs and the power lines connecting the DGUs respectively. The incidence matrix $\mathcal{B} \in \mathbb{R}^{n \times m}$ can describe the network and its entries are given

by

$$\mathcal{B}_{ij} = \begin{cases} +1 & \text{if } i \text{ is the positive end of } j \\ -1 & \text{if } i \text{ is the negative end of } j \\ 0 & \text{otherwise} \end{cases}$$

Combining (3.1) and (3.2) results in the overall microgrid system that can be written as

$$L_s \dot{I}_s(t) = -R_s I_s(t) - V(t) - u(t), \tag{3.3a}$$
$$C_s \dot{V}(t) = I_s(t) - I_L - \mathcal{B} R^{-1} \mathcal{B}^T V(t). \tag{3.3b}$$

Where $I_s, V, u : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$, $I : \mathbb{R}_{\geq 0} \to \mathbb{R}^m$ and $I_L \in \mathbb{R}^n$ represents the constant overall current demand. Moreover $L_s, R_s, L_s \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ are positive definite diagonal matrices.

3.1 Control Objectives

Generally the control objectives in DC microgrids are voltage regulation and (proportional) current sharing. Careful coordination is needed to ensure that voltage levels are within acceptable ranges, to provide a proper functioning of connected loads, and current sharing which prevents the overstressing of any source and helps to elongate the lifetime of the power-generating entities in the microgrid. In this research, another objective is added. Since the buck converter's output can be controlled it is constrained as well, making up the third objective. The three objectives are outlined below.

3.1.1 Current Sharing

At steady state, the second line in (3.3) with steady state solution $(\overline{I}_s, \overline{V}, \overline{I})$, constant inputs $\overline{u}$ and given current demand I_L satisfies

$$I_L = \overline{I}_s - \mathcal{B} R^{-1} \mathcal{B}^T \overline{V}. \tag{3.4}$$

Which suggests that at steady state the total current demand $\mathbb{1}_n^T I_L$ is equal to the total generated current $\mathbb{1}_n^T \overline{I}_s$[1], where $\mathbb{1}_n \in \mathbb{R}^n$ represents a vector of ones. The goal is to achieve a form of current sharing in the microgrid, which means that the total generated current is shared among the various DGUs. To improve the generation efficiency, the current is shared proportionally to the generation capacity of their respective energy source [10]. This desire can be mathematically formulated by

$$w_i I_{s_i} = w_j I_{s_j}, \quad \forall\ i, j \in \mathcal{V} \tag{3.5}$$

where $w_i \in \mathcal{R}_{\geq 0}$ is the weighting value for the i-th DGU, corresponding to the generation capacity of its converter. For example, a relatively large value of w_i resembles a small generation capacity [50]. To avoid overstressing of a source, it is desirable to distribute the total load in the network fairly among the DGUs, leading to the first objective concerning the desired steady state value of the generated currents $\overline{I}_s$.

$$\lim_{t \to \infty} I_s(t) = \overline{I}_s := W^{-1} \mathbb{1}_n i_s^\star, \tag{3.6}$$

[1] A property of the incidence matrix $\mathcal{B}$ is that it satisfies, $\mathbb{1}_n^T \mathcal{B} = \mathbb{0}$

where W is the positive diagonal desired weighing matrix and $i_s^\star \in \mathbb{R}$. The steady state requirement of $\mathbb{1}_n^T I_L = \mathbb{1}_n^T \overline{I}_s$, automatically requires $i_s^\star = \mathbb{1}_n^T I_L / (\mathbb{1}_n^T W^{-1} \mathbb{1}_n)$. This implies that $\mathbb{1}_n^T \bar{I}_s = \mathbb{1}_n^T W^{-1} \mathbb{1}_n i_s^\star = \mathbb{1}_n^T I_L$ is satisfied. Thereby the desired generated current depends on the overall load current, as

$$\overline{I}_s = W^{-1} \mathbb{1} \big(\mathbb{1}_n^T I_L / (\mathbb{1}_n^T W^{-1} \mathbb{1}_n) \big). \tag{3.7}$$

3.1.2 Voltage Regulation

The requirement for current sharing generally does not permit the steady state load voltage to be equal to a desired reference value. Because currents in the network are tightly related to voltages it is not possible to freely adjust the voltages while still expecting a proper allocation of the generated currents [50]. Therefore to regulate the voltages, two alternatives are often considered; average voltage regulation [10], [50] and a bounded voltage regulation as proposed in [38]. Firstly the control objective proposed in [38] is portrayed, where the load voltages for each DGU are constrained.

$$V_{min} \leq V(t) \leq V_{max}, \tag{3.8}$$

where $\underline{\mathrm{V}}$ and $\overline{V}$ denote the minimum and maximum allowable load voltages. The load voltages stay within these bounds for constant load demand. However, a substantially large load demand variation can cause the load voltages to reach values outside the bounds. Alternatively, in [10] the controller has to guarantee voltage balancing, where the weighted average value of the desired reference voltages, $V^\star$, is kept equal to the weighted average value of the voltages at steady state, $\bar{V}$.

$$\lim_{t \to \infty} \mathbb{1}_n^T W^{-1} V(t) = \mathbb{1}_n^T W^{-1} \bar{V} = \mathbb{1}_n^T W^{-1} V^\star, \tag{3.9}$$

where W is the positive diagonal desired weighing matrix as defined in *Section 3.1.1*, such that at the converters with a relatively large generation capacity, there is a relatively small voltage deviation.

3.1.3 Input Feasibility

To achieve Objective 1 and 2, the control input u can be adjusted. This input is, as well as the load voltage, constrained by bounds

$$u_{min} \leq u(t) \leq u_{max}, \tag{3.10}$$

where $\underline{\mathrm{u}}$ and $\overline{u}$ denote the minimum and maximum allowable buck converter output voltages.

Chapter 4

Model Predictive Control

A MPC based algorithm is used to control the DC microgrid and reach consensus for current sharing, while keeping the buck converter output voltages and load voltages within the bounds. An optimal control problem can be solved using a MPC scheme, in this section the general idea behind MPC will be explained.

MPC was developed around 1970 [52] and since the evolution of fast processors, that can handle complex problems in a split second, the control methodology has gained more and more popularity. MPC applications can be found in chemical plants [28], oil refineries [58] and more recently in power systems. Even though MPC requires high computational efforts, it can easily handle multivariable, nonlinear systems and has excellent constraint-handling capabilities [53].

MPC relies on the dynamic model of a process. The future behaviour of a system is predicted using that dynamic model, together with given measurements or estimates of the current state of the system and a hypothetical future input trajectory or feedback control policy. This predicted behavior is used to optimize some cost function subjected to constraints. The optimization yields an optimal control sequence of which only the first control input is implemented in the system. Thereafter the process is repeated for the next time instant using newly available information on the system state. This repetition is instrumental in reducing the gap between the predicted and the actual system response [6]. The main advantage of MPC is the fact that it allows the current time instance to be optimized, while keeping future time instance into account. This is achieved by optimizing over a finite time-horizon, but only implementing the current time instance and then optimizing again, repeatedly.

A MPC optimization will be computed at each time step, $t \in \mathcal{T}$, over a time horizon with step, $k \in \mathcal{K}$, where

$$\mathcal{K} = (0, 1, ..., N-2, N-1), \qquad \mathcal{T} = (0, 1, ..., t_{\max}-1, t_{\max}). \tag{4.1}$$

So the basis for MPC control are predictions of the future as it uses known characteristics of the dynamic behavior of processes to feed forwardly drive the processes to the desired conditions. This in contrast to PID control, where the control actions are based on the past. Nevertheless, it must be stated that MPC is not a specific control strategy, but belongs to a family of wide range

control methods developed around common ideas. More specifically, all MPC based control methods are designed according to these five principles [52]:

1. Process and disturbance models
2. Performance index
3. Constraints
4. Optimization
5. Redecing horizon Principle

Which will be discussed, according to [6] and [52], in the next sections.

4.1 Prediction model

The model is discrete and predicts the process signals for future time instants over a specified horizon. Different prediction models can be utilized to capture the dynamic and static interactions between input, output and disturbance variables, however in this section a state space description will be used.

The model is the cornerstone of MPC and the success of MPC depends on the accuracy of the process model. Inaccurate predictions can make matters worse, instead of better [47]. The prediction model will be used to make an estimate of the future behaviour of the system, over the horizon $\mathcal{K}$. A generalized model can be portrayed by

$$\hat{x}(t|t) = x(t), \tag{4.2a}$$
$$\hat{y}(t|t) = y(t), \tag{4.2b}$$
$$\hat{x}(t+k|t) = \hat{A}\hat{x}(t+k-1|t) + \hat{B}\hat{u}(t+k-1|t), \qquad \forall\ k \in \mathcal{K}\backslash\{0\} \tag{4.2c}$$
$$\hat{y}(t+k-1|t) = \hat{C}\hat{x}(t+k-1|t). \qquad \forall\ k \in \mathcal{K}\backslash\{0\} \tag{4.2d}$$

Where the first two lines denote the current state of the model or plant and serve as the initial conditions, $\hat{x}(t+k|t)$ is the predicted value of $x(t+k)$ at time instance t. Moreover $\hat{u}$ and $\hat{y}$ are the estimated system input and output respectively.

4.2 Performance index

The MPC model, tries to optimize a performance index or cost-criterion over the future horizon. The performance index ensures that the future output $\hat{y}(t+k|t)$, which is the predicted value of $y(t+k)$ at time t, follows a determined reference signal $r(t+k)$ over the chosen horizon. The error, given by

$$e(t+k|t) = \hat{y}(t+k|t) - r(t+k), \tag{4.3}$$

should be optimized for all the time steps till N. The performance index can be optimized by adjusting the controllable input u, of which only the first step from the optimal solution $u(t)$ is

applied and sent to the process. A standard 2-norm performance index, is portrayed by

$$J(u,t) = \sum_{k\in\mathcal{K}} e(t+k|t)^T \Gamma(k) e(t+k|t). \tag{4.4}$$

Where $\Gamma(k)$ is a diagonal selection matrix that can define the degree of importance through weights of the error at each horizon step k. Furthermore, it is also possible to penalize the controllable input in the performance index.

4.3 Constraints

In general, all processes are subject to constraints. This can be due to safety or environmental regulations, consumer specifications, physical restrictions etc. MPC is a suitable method to handle problems with hard constraints. These constraints can be equality or inequality constraints and cannot be violated under any circumstance. Inequality constraints can be bounds on control, state or output signals.

$$u_{min} \leq \hat{u}(t+k|t) \leq u_{max}, \tag{4.5a}$$

$$x_{min} \leq \hat{x}(t+k|t) \leq x_{max}, \tag{4.5b}$$

$$y_{min} \leq \hat{y}(t+k|t) \leq y_{max}. \tag{4.5c}$$

Equality constraints, usually originate from the respective model itself. An example to ensure more system robustness is given in [52] by

$$\Delta\hat{u}(t+k|t) = 0, \quad \forall k \geq N_c \tag{4.6}$$

Where N_c is is a control horizon that is $0 \leq N_c \leq N$. (4.6) forces the control signal to become constant over time and reach some form of steady state.

4.4 Optimization

The optimization of (4.4) yields an optimal control sequence. In order to obtain the sequence of optimal future control signals $\hat{u}(t+k|t)$, a sequence of predicted outputs, $\hat{y}(t+k|t)$, is calculated as a function of past inputs and outputs through the chosen model that minimizes the performance index subject to given constraints.

4.5 Redecing horizon principle

After the optimization of the optimal control sequence, only the first control sample will be implemented to the system. Thereafter the horizon is shifted with one time step and optimization is started again. At time t the future control sequence $\{\hat{u}(t|t), ..., \hat{u}(t+N-1|t)\}$ is determined so that the performance index is minimized while keeping the constraints in eight. The first input $\hat{u}(t|t)$ of the sequence is then applied to the real process. Then the horizon is shifted to time $t+1$ and the optimization is performed over that time instance.

4.6 Example

To understand the concept of MPC, one can consider a well-known analogy of a driver steering a car [44]. Within that analogy

- **Prediction Model**: Describes how the vehicle is expected to move on the map.
- **Disturbances**: The driver's inattention and other reasons for uncontrolled deviation from the desired trajectory.
- **Performance Index**: May be the goal of minimum time, minimum distance etc. to reach the desired location.
- **Constraints**: The set of rules to drive on roads, respect one ways, don't exceed mechanical capabilities of the vehicle.
- **Receding Horizon**: Would re-plan the route of the car and the corresponding driver actions periodically in time, find the overall set of actions over a time horizon, apply the first and then re-plan for the next-step.

Chapter 5

Centralized MPC for DC Microgrids

As microgrids require a consensus algorithm in order to optimize its operations, in this research, an MPC scheme, as introduced in *Section 4*, is applied to solve the optimal control problem of the DC microgrid from *Section 3*. MPC is particularly interesting for power system balancing because of the systematic way to include input and state constraints as well as its robustness to external disturbances. However, before a distributed MPC scheme for DC microgrids can be developed, firstly the definition of a centralized control is explained. Wherafter the centralized MPC approach for achieving Objectives *1*, *2* and *3* is studied. In this section a centralized MPC scheme for DC microgrids established as in [38], will be discussed.

5.1 What is centralized control?

In a centralized control model all of the control aspects are concentrated in a single entity. There is one controller that handles the execution of a set of procedures associated with the process and all control operations are reported to this controller (see *Figure 5.1*). Because of the central architecture and all control actions are wired back, control efforts require much computational power. Therefore centralized MPC often becomes impractical for the control of large-scale systems.

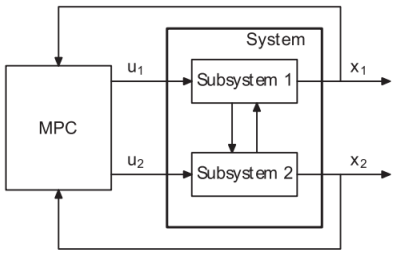

FIGURE 5.1: Centralized MPC Architecture

5.2 Centralized MPC for DC microgrids

In [38] a classical MPC approach is proposed to solve the optimal control problem of the DC microgrid. Optimization of a MPC problem takes place at each time step, whereafter only the first control input value from the optimal control sequence is used and the horizon is shifted towards the next time step and the procedure is repeated. Again there is a distinction between current time step $t \in \mathcal{T}$ and horizon step $k \in \mathcal{K}$, where

$$\mathcal{K} = (0, 1, ..., N-2, N-1), \qquad \mathcal{T} = (0, 1, ..., t_{\max} - 1, t_{\max}). \tag{5.1}$$

MPC optimizes the controllable buck converter output over a future horizon such that the control objectives are met, whilst considering grid constraints. A model is used to define the relationships between the generated current, load voltage, buck converter output voltage etc. Such a continuous system model has been formulated in (3.3), where the physical representation of the DC microgrid is depicted. To design the MPC scheme these dynamics have to be discretized. Discretization of (3.3) by Euler approximation method [37] gives

$$I_s(k+1) = I_s(k) + TL_s^{-1}\big(- R_s I_s(k) - V(k) + u(k)\big), \tag{5.2a}$$
$$V(k+1) = V(k) + TC_s^{-1}\big(I_s(k) - I_L - \mathcal{B}R^{-1}\mathcal{B}^T V(k)\big), \tag{5.2b}$$

where T is the sampling period.

Now the optimal control problem in the form of a MPC scheme can be stated. As can be seen from Objective 1, the total desired generated current $\overline{I}_s$ depends on the overall load current I_L, which is known. If the load demands were not to be known, an observer could be designed to estimate the load demand [37]. In [38] the performance index is set so that it achieves proportional current sharing by minimizing the generated current with its steady state.

$$obj(t) = \sum_{k \in \mathcal{K}} \big(\hat{I}_s^T(t+k|t) - \overline{I}_s\big)^2. \quad \forall\, k \in \mathcal{K} \tag{5.3}$$

However in this research the quadratic cost function proposed in [23] is adopted to achieve proportional current sharing, in which the controllable input minimizes,

$$obj(t) = \sum_{k \in \mathcal{K}} \hat{I}_s^T(t+k|t) W\mathcal{L}W\hat{I}_s(t+k|t) + \hat{u}^T(t+k|t)\mathcal{R}\hat{u}(t+k|t). \quad \forall\, k \in \mathcal{K} \tag{5.4}$$

Where $\hat{I}(t+k|t)$ represents the predicted value of $I(t+k)$ at time step t and $\mathcal{L} \in \mathbb{R}^{n \times n}$ denotes the weighted Laplacian matrix associated with the physical network, $\mathcal{L} = \mathcal{B}\mathcal{Q}\mathcal{B}^T$. Furthermore $\mathcal{R}, \mathcal{Q} \in \mathbb{R}^{n \times n}$, are weighted identity matrices. Proportional current sharing is achieved if $\mathcal{L}W\hat{I}_s(t+k|t) = \mathbb{0}$ holds. The term $\hat{I}_s^T W\mathcal{L}W\hat{I}_s$ is equal to zero, if and only if $\hat{I}_s$ is equal to the null space of matrix $\mathcal{L}W$ and can be denoted as $\mathcal{N}(\mathcal{L}W) = \{\alpha \mid \mathcal{L}W\alpha = \mathbb{0}\}$ with $\alpha \in \mathbb{R}^n$. This means that at steady state the aforementioned term is zero if and only

$$\overline{I}_s = \alpha, \tag{5.5}$$

notice that, $\overline{I}_s \in \mathcal{N}(\mathcal{L}W)$, and to that effect

$$\mathbb{1}^T\alpha = \mathbb{1}^T I_L. \tag{5.6}$$

Then by (3.7), proportional current sharing is achieved [23]. The second term in the cost function, captures the cost of the controllable input so that the system will go to its steady state. The objective is subjected to the constraints represented by the discrete time system dynamics and inequality constraints on states and input variables, displayed in the next section.

5.2.1 Total centralized MPC scheme

The optimal control problem for a DC microgrid consisting of buck converters established in *Section 3* can now be represented as a MPC optimization problem. Where proportional current sharing is achieved and also buck converter output as well as load voltage regulation is ensured. The total MPC scheme for DC microgrids is given by,

Performance index

$$obj(t) = \sum_{k\in\mathcal{K}} I_s^T(t+k|t)W\mathcal{L}WI_s(t+k|t) + u(t+k|t)^T\mathcal{R}u(t+k|t), \quad \forall\ k \in \mathcal{K} \tag{5.7}$$

Initial conditions

$$\hat{I}_s(t|t) = I_s(t), \tag{5.8a}$$
$$\hat{V}(t|t) = V(t), \tag{5.8b}$$

Process model

$$\hat{I}_s(t+k|t) = \hat{I}_s(t+k-1|t) + TL_s^{-1}\Big(-R_s\hat{I}_s(t+k-1|t) - \hat{V}(t+k-1|t) + \hat{u}(t+k-1|t)\Big), \quad \forall\ k \in \mathcal{K}\backslash\{0\} \tag{5.9a}$$
$$\hat{V}(t+k|t) = \hat{V}(t+k-1|t) + TC_s^{-1}\Big(\hat{I}_s(t+k-1|t) - I_L - \mathcal{B}R^{-1}\mathcal{B}^T\hat{V}(t+k-1|t)\Big), \quad \forall k \in \mathcal{K}\backslash\{0\} \tag{5.9b}$$

Constraints

$$\hat{V}_{min} \leq V(t+k|t) \leq \hat{V}_{max}, \quad \forall\ k \in \mathcal{K} \tag{5.10a}$$
$$\hat{u}_{min} \leq u(t+k|t) \leq \hat{u}_{max}. \quad \forall\ k \in \mathcal{K} \tag{5.10b}$$

Equations (5.8a - 5.8b) denote the initial conditions and thus the current states of the system at time t. The microgrid dynamics are portrayed by (5.9a - 5.9b) and the voltage regulation and input feasibility constraints by (5.10a - 5.10b) respectively. Noticeably, Objective *2* and *3* are incorporated in the constraints and Objective *1* in the performance index.

Chapter 6

Distributed MPC for DC Microgrids

In this section the distributed control strategy for MPC in the DC microgrids is outlined. Firstly the general idea of distributed control is explained. Whereafter the basic mathematical concepts of convexity, Lagrange duality and the subgradient method are discussed. These ideas are necessary to decompose a centralized control design to a form of *communicative* distributed control. Then the general approach for the decomposition of centralized MPC problems is outlined, after that this technique is applied to centralized MPC for DC microgrids from *Section 5*.

6.1 What is distributed control?

The number of nodes in the network causes a centralized MPC scheme to be unfeasible due to the computational burden on the controller (communication bandwidth limitations), as all the control inputs are computed in one optimization problem. Therefore other distributed control structures are more suitable for multi-agent systems comparable to the electricity grid. In distributed control networks the total system consists of several subsystems that have their own local controllers, which allow information to travel between them (see *Figure 6.1*). The local control inputs are computed using local measurements with local dynamics, resulting in multiple local optimizations. Distributed control is not to be confused with decentralized control, which is also based on local regulators, but does not allow for an exchange of information among the various local controllers. Distributed controllers communicate, so that they have knowledge of the behavior of the others [46]. When the local controllers are designed with MPC, the information transmitted typically consists of the future predicted control or state variables computed locally, so that any local regulator can predict the interaction effects over the considered prediction horizon.

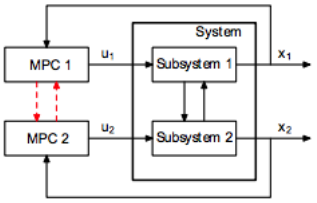

FIGURE 6.1: Distributed MPC Architecture

Generally, in distributed MPC there are two approaches for the information exchange between the controllers. There are so called *noniterative algoritms*, that transmit and receive information from other controllers once within each sampling time. Contrariwise, there exist *iterative algoritms*, that allow for information to be exchanged multiple times within the sampling period, [16]. It is obvious that within the iterative method, the information available is higher and thus system objectives can be reached faster and even within one time interval.

The objective is to achieve some degree of coordination in a multi-agent system, where each controller solves a local MPC problem. The type of coordination depends on the knowledge of each others cost functions. A distinction can be made between *communication based strategies* and *cooperation based strategies*. Pure communication based strategies optimize their own local performance index and have no knowledge of each others cost functions. Whereas with cooperation based distributed control, the subsystems cooperate, rather than compete, with each other to reach system wide objectives (global cost function [7], [54]). Iterative and cooperative control strategies can replicate ideal centralized strategies. However the design phase of such controllers is more complex relative to centralized controllers.

Note that distributed solutions are particularly attractive in large-scale networks where a centralized solution is infeasible, nonscalable, too costly, or too fragile. Therefore the MPC optimization for DC microgrids (*Section 5*) has to be decomposed and modified into a distributed form. This is done according to theories established in [43], which is based on the mathematical principles of convex optimization, discussed in the next section.

6.2 Basic Concepts of Convex Optimization

For the decomposition of centralized control problems, some mathematical background on convexity, Lagrange duality and subgradient methods is needed. A review of those concepts is given in this section.

6.2.1 Convex optimization and Lagrange duality

A general optimization problem, has the form of

$$\begin{aligned} \min_{x} \quad & f_0(x), \\ \text{subject to} \quad & f_i(x) \leq 0, \quad i = 1, ..., m \\ & h_i(x) = 0, \quad i = 1, ..., p \end{aligned} \tag{6.1}$$

where $x = (x_1, ..., x_n) \in \mathbb{R}^n$ is the optimization variable, the function $f_0 : \mathbb{R}^n \to \mathbb{R}$ is the cost or objective function, $(f_1, ..., f_m) : \mathbb{R}^n \to \mathbb{R}$ are the m inequality constraint functions and $(h_1, ..., h_p) : \mathbb{R}^n \to \mathbb{R}$ are the p equality functions [4]. If the objective and inequality are convex and the equality constraint functions are linear, the problem is considered to be a convex optimization problem [39]. The domain is the set of points for which the objective and all constraints functions are defined,

$$\mathcal{D} = \bigcap_{i=0}^{m} \text{dom } f_i \cap \bigcap_{i=0}^{p} \text{dom } h_i.$$

A point $x \in \mathcal{D}$ is feasible if it satisfies the constraints in (6.1). The problem is said to be feasible if there exist at least one feasible point and infeasible otherwise. Along those lines the optimal value $p^\star$ of the problem (6.1) is achieved at $x^\star$, thus $p^\star = f_0(x^\star)$. Convexity of the problem makes the optimization easier, because a local optimum of convex optimization is also globally optimal.

Solving the convex problem (6.1) implies that the hard constraints need to be taken into account, to find the optimal $x^\star$. A method to solve the problem, without explicitly solving the hard constraints, is with the use of Lagrange multipliers. The idea of Lagrangian relaxation is to take the constraints into account by augmenting the objective function with a weighted sum of the constraint functions [30]. Define the *Lagrangian* $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as

$$L(x, \lambda, \theta) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \theta_i h_i(x), \tag{6.2}$$

where $\lambda \in \mathbb{R}^m_{\geq 0}$ and $\theta \in \mathbb{R}^p$ are the Lagrangian multipliers associated with the inequality and equality constraints in (6.1). These vectors λ and θ are also named as the dual variables, whereas x is called the primal variable. So the Lagrangian can be thought of as a modified version of the objective function to the original convex optimization problem which accounts for each of the constraints and the Lagrange multipliers λ_i and θ_i can be seen as "costs" associated with violating different constraints [13].

Primal and dual problem

The problem depicted in (6.1) is called the primal problem and the $f_0(x)$ is referred to as the primal objective. To show the relationship between the Lagrangian and the primal problem, the primal problem can be written in terms of the Lagrangian

$$\begin{aligned} \sup_{\lambda \geq 0, \theta} L(x, \lambda, \theta) &= \sup_{\lambda \geq 0, \theta} \Big(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \theta_i h_i(x) \Big), \\ &= \begin{cases} f_0(x) & f_i(x) \leq 0 \text{ and } h_i(x) = 0, \quad \forall i \\ \infty & \text{otherwise.} \end{cases} \end{aligned} \tag{6.3}$$

If x is feasible and thus constraints $f_i(x) \leq 0, i = 1, ..., m$ and $h_i(x) = 0, i = 1, ..., p$ are satisfied, the supremum over Lagrangian with the dual variables is equal to the primal objective. The optimal value of the primal problem in terms of the Lagrangian is

$$p^\star = \inf_{x \in \mathcal{D}} \sup_{\lambda \geq 0, \theta} L(x, \lambda, \theta). \tag{6.4}$$

On the other hand the dual objective function can be defined as the minimum value of the

Lagrangian over x: for λ, θ,

$$g(\lambda,\theta) = \inf_{x\in\mathcal{D}} L(x,\lambda,\theta) = \inf_{x\in\mathcal{D}} \Big(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \theta_i h_i(x)\Big). \tag{6.5}$$

Notice that $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$, is a concave function of λ and θ. The dual variables are dual feasible if $\lambda \geq 0$. Taking the minimum of the Lagrangian, for a fixed λ and θ over $x \in \mathcal{D}$ gives a lower bound for optimal value $p^\star$ [30]. A proof is given below

$$\begin{aligned} g(\lambda,\theta) &= \inf_{x\in\mathcal{D}} L(x,\lambda,\theta), \\ &\leq L(x^\star,\lambda,\theta), \\ &= f_0(x^\star) + \sum_{i=1}^{m} \lambda_i f_i(x^\star) + \sum_{i=1}^{p} \theta_i h_i(x^\star), \\ &\leq f_0(x^\star) = p^\star. \end{aligned} \tag{6.6}$$

The first and third step of (6.6) follow from (6.5) and (6.2) respectively. The second step follows from the fact that the infimum over the Lagrangian of a possible value of x is always less or equal to the Lagrangian of $x^\star$. The last step comes from the fact that $x^\star$ is primal feasible and λ and v are dual feasible, thus the sums of the weighted constraints are non-positive [13]. Finding the best lower bound of (6.5) is called the dual problem and gives the optimal dual objective,

$$d^\star = \sup_{\lambda\geq 0,\theta} \inf_{x\in\mathcal{D}} L(x,\lambda,\theta). \tag{6.7}$$

By switching the order of the infimum and supremum above, we obtain an entirely different optimization problem [4]. The dual problem can be depicted as

$$\begin{aligned} \max_{\lambda,\theta} \quad & g(\lambda,\theta), \\ \text{subject to} \quad & \lambda \geq 0, \end{aligned} \tag{6.8}$$

where $\lambda^\star$ and $v^\star$ are optimal dual variables if they are optimal for the dual problem. Thus $d^\star = g(\lambda^\star, v^\star)$.

Weak and strong duality

The optimal value of the dual problem is by definition the best lower bound of the primal problem's optimal value. In mathematical notation this is

$$d^\star \leq p^\star. \tag{6.9}$$

This is true for any optimization problem, not just for convex primal problems. Basically the equation says that the greatest lower bound is always less or equal than the least upper bound. A proof for this equation is given below. Take $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ where $W \in \mathbb{R}^n$ and $Z \in \mathbb{R}^m$, then for any $w_0 \in W$ and $z_0 \in Z$ the following holds,

$$\begin{aligned} \inf_{w\in W} f(w,z_0) \leq f(w_0,z_0) \leq \sup_{z\in Z} f(w_0,z), \\ \sup_{z_0\in Z} \inf_{w\in W} f(w,z_0) \leq \inf_{w_0\in W} \sup_{z\in Z} f(w_0,z), \\ \sup_{z_\in Z} \inf_{w\in W} f(w,z) \leq \inf_{w\in W} \sup_{z\in Z} f(w,z). \end{aligned} \tag{6.10}$$

The first equation is trivial and states that lower bound is less than or equal the upper bound of a function. Since this is true for all $w_0 \in W$ and $z_0 \in Z$, the second and third equation also hold [4] .

The difference between $p^\star - d^\star$ is called the duality gap. If this gap is greater than zero, this implies weak duality, which is a consequence of (6.10). However for most (not all) convex problems,

$$p^\star = d^\star, \tag{6.11}$$

holds and the duality gap is zero, implying strong duality. Where the best bound from the dual function is equal to the optimal value of the primal (original) problem. By means of this principle, the primal problem (6.1) can be equivalently solved by solving the dual problem (6.8). There exist conditions that ensure strong duality holds, of which the most well-known is called Slater's condition. When the inequality constraint $f_i(x) \leq 0$ are replaced by strict inequality constraints $f_i < 0$. If there exists some feasible primal solution x, for which the strictly inequality constraints are satisfied, Slater's condition holds. And if this condition holds, strong duality also holds [4]. To display the interesting characteristics of strong duality, three important interpretations of Lagrange duality are described below.

Complementary slackness
If strong duality holds, an interesting relationship arises called complementary slackness. Let

$$\begin{aligned} f_0(x^\star) &= g(\lambda^\star, \theta^\star), \\ &= \inf_{x \in \mathcal{D}} \Big(f_0(x) + \sum_{i=1}^{m} \lambda_i^\star f_i(x) + \sum_{i=1}^{p} \theta_i^\star h_i(x) \Big), \\ &\leq f_0(x^\star) + \sum_{i=1}^{m} \lambda_i^\star f_i(x^\star) + \sum_{i=1}^{p} \theta_i^\star h_i(x^\star), \\ &\leq f_0(x^\star). \end{aligned} \tag{6.12}$$

Because the first and last equation are identical, the whole chain is equal. Which indicates that

$$\lambda_i^\star f_i(x) = 0, \quad i = 1, ..., m. \tag{6.13}$$

Because

$$\begin{aligned} \lambda_i^\star > 0 &\Longrightarrow f_i(x^\star) = 0, \\ f_i(x^\star) < 0 &\Longrightarrow \lambda_i^\star = 0. \end{aligned} \tag{6.14}$$

And it means that i^{th} optimal Lagrange multiplier is zero unless the constraint is active at the optimum.

KKT conditions

Furthermore if strong duality holds, the primal and dual optimal points must satisfy Karush-Kuhn-Tucker (KKT) conditions.

$$\begin{aligned} f_i(x^\star) &\leq 0, \quad i = 1, ..., m \\ h_i(x^\star) &= 0, \quad i = 1, ..., p \\ \lambda_i^\star &\geq 0, \quad i = 1, ..., m \\ \lambda_i^\star(x^\star) &\leq 0, \quad i = 1, ..., m \\ \nabla f_0(x^\star) + \sum_{i=1}^{m} \lambda_i^\star \nabla f_i(x^\star) + \sum_{i=1}^{p} \theta_i^\star \nabla h_i(x^\star) &= 0. \end{aligned} \tag{6.15}$$

These conditions are necessary and sufficient for primal-dual optimality [13].

Price or tax interpretation

Lagrange duality can be interpreted in an interesting economic manner. Suppose x is the operational state of an enterprise and $f_0(x)$ indicates the cost of operating at x, where $-f_0(x)$ portrays the profit. The constraints $f_i(x) \leq 0$ represent a limit, for example on the amount of warehouse space. The x at which the profit is maximized can be found through

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0 \quad i = 1, ..., m. \end{aligned} \tag{6.16}$$

The optimal profit is $-p^\star$. The company can pay a price λ_i per unit of the material constraint. For example $f_1(x) \leq 0$, represents a limit on warehouse space. The company can use extra warehouse space at a cost of λ_1 per square meter. Now the total cost for state x and constraint prices λ_i is the Lagrangian of (6.16). The dual function $g(\lambda)$ represents the cost to the firm, while the company desires to minimize the Lagrangian, as a function of price vector λ. The optimal dual value, $d^\star$, is the optimal cost to the enterprise under the least favorable set of prices. Strong duality in this context, means that there is no advantage for the firm to violate the constraints. Weak duality implies that, the cost for the firm in case constraint violations can be bought and sold, is less than or equal to the cost in the original situation. The duality gap is then the minimum possible advantage to the enterprise of being allowed to pay for constraint violations and receive payments for non-tight constraints. In a power network with multiple decision makers, it is useful to work with price like concepts of Lagrange multipliers when dealing with allocation problems [4],[30].

6.2.2 Subgradient-based optimization

Another convex functions' important property is that it the minimum can be found using a search method. Let $f_0 : \mathbb{R}^n \to \mathbb{R}$ be a convex function and the goal is to solve

$$\min_{x \in \mathbb{R}^n} f_0(x). \tag{6.17}$$

As mentioned in the previous section, such a function has a global minimum $x^\star$. Which means that for any point x_0, there is a connected path between x_0 and $x^\star$ such that for every point x_i along that path, $f_0(x_i) \leq f_0(x_0)$ holds. This characteristic will allow an iterative search method to always reduce $f_0(x)$ so that it eventually reaches $f_0(x^\star)$ [39].

The goal of such a method is to generate a sequence of feasible points $\{x(t)\}_{t=0}^{t=\infty}$, where $f_0(x(t+1)) < f_0(x(t))$ and the state vector converges to the optimal $x^\star$ [30]. With the gradient method, as the name suggests, the gradient (first derivative) of a function is calculated at each iteration t and a step in the opposite direction is then taken. So that the method iteratively generates a sequence of feasible points for (6.17) as

$$x(t+1) = x(t) - \gamma(t)w(x(t)). \tag{6.18}$$

Where $\gamma(t)$ is the step size or learning rate and $w(x(t))$ is the gradient of f_0 at point $x(t)$ if f_0 is differentiable. If not $w(x(t))$ denotes the subgradient of f_0 at point $x(t)$ [39]. Convergence of the subgradient method depends on the chosen step size. There are various theories on step size options, e.g. constant step size ($\gamma = \gamma(t)$) or diminishing step size ($\gamma(t) = (1+m)/(t+m)$ with $m \geq 0$).

Section 6.2 already mentioned that some convex optimization problems are hard due to the need to explicitly solve the constraints and can be more easily calculated through solving the dual problem (if strong duality holds). In this case the subgradient method can be applied to the dual problem. If the dual problem of (6.8) is considered, the optimal dual variables λ and θ can be found through the subgradient method for a given $x(t)$. Since $g(\lambda, \theta)$ is concave in (λ, θ), even if the primal problem is not convex. In accordance with [30], let

$$g_t(\lambda(t), \theta(t)) = \inf_{x(t)} L(x(t), \lambda(t), \theta(t)), \tag{6.19}$$

be the dual problem for iteration t. The sequence of feasible points for the Lagrange multipliers is given by

$$\lambda(t+1) = \max(0, \lambda(t) + \gamma(t)f(x(t))), \tag{6.20a}$$
$$\theta(t+1) = \theta(t) + \gamma(t)h(x(t)), \tag{6.20b}$$

with $f(x(t))$ and $h(x(t))$ as a subgradients of (6.19) at the current point $x(t)$.

Convex optimization problems can be solved by subgradient methods. Therefore the method is often used with decomposition methods in (large scale) dynamical systems to solve the problem in a distributed manner. However this subgradient method does not always ensure good convergence, as is depends so much on the chosen step size and choosing a proper step size can be difficult. This is due to the fact that a step size that is too small leads to slow convergence, while a learning rate that is too large can hinder convergence and cause a function to fluctuate around the minimum or even to diverge. Furthermore it is often thought that a main source of difficulty for this method to find the global minimum (or maximum) is the proliferation of local minima with much higher error than the global minimum [11]. Dual optimization problems can have these saddle points where the subgradients are close to zero and where the subgradient algorithm can get trapped in. To deal with these challenges, a tweaked version of the subgradient descent algorithm will be used in this research.

Momentum

Momentum is a method that helps to accelerate the subgradient algorithm in the relevant direction and dampens oscillations [41]. The algorithm determines the next update as a linear

combination of the subgradient and the previous update [41]. It does this by adding a fraction β of the update vector of the past time step to the current subgradient. Lets consider the problem depicted in (6.19), where dual variable θ is to be determined by the update rule. The update rule of the subgradient descent algorithm with *Momentum* is represented by

$$m_{avg}(t) = (1-\beta)m_{avg}(t-1) + \beta h(x(t)), \tag{6.21a}$$

$$\theta(t+1) = \theta(t) + \gamma * m_{avg}(t), \tag{6.21b}$$

where $h(x(t))$ is the subgradient of the dual variable at current point $x(t)$, γ is the step size and $\beta \in [0, 1)$ is the momentum term. The name stems from an analogy to momentum in physics. An intuitive understanding of momentum can be painted by a ball rolling down the hill. Its mass is constant all the way, but because of the gravitational pull, its velocity increases over time, making momentum increase. The same concept can be applied to cost minimization. When a subgradient from the previous time step "points" in the same direction as our current time step, "the speed" is increased by going "down hill." Therefore term, $\beta * m_{avg}(t-1)$, increases for dimensions whose subgradients point in the same directions and reduces updates for dimensions whose subgradients change directions. As a result, faster convergence and reduced oscillation is achieved. So *Momentum* is a tweaked version of subgradient descent that provides a faster convergence of θ to the optimal $\theta^\star$.

6.3 Dual decomposition method

In large scale optimization problems of a network with decision-makers/agents that have coupled dynamics (such as low voltage microgrids), a centralized solution is, as mentioned, infeasible, non-scalable, too costly, or too fragile [39]. Therefore the techniques in *Section 6.2* can be used to decompose the original large problem into distributed solvable subproblems. The idea is that each decision-maker solves its own subproblem, based only on local information, and together they arrive to the solution of the original problem [36].

The dual decomposition method, proposed in [43], can be used to achieve distributed control from a centralized control architecture. The theory uses the concept of the duality, (6.8), to decompose the problem, whereby the coupling constraints between decision-makers are relaxed by Lagrange multipliers. Each agent has access only to a local model and tries to optimize his control action based on a local cost function. The interaction with neighboring agents is handled through negotiations [42]. These negations are executed by means of neighbors updating the Lagrange multipliers that act as prices in a market mechanism trying to achieve mutual agreement between solutions of the subproblems. Thus by the introduction of prices, the agents can pay each other to modify the values and find a common equilibrium. The prices can be updated in a distributed manner using the subgradient method of *Section 6.2.2* which is coordinated by means of some kind of signaling.

A small example from [43] gives a visualization on dual decomposition and the utilization of prices. Let the centralized minimization problem be

$$J = \min_{x_i} \big(f_1(x_1, x_2) + f_2(x_2) + f_3(x_3, x_2)\big). \tag{6.22}$$

The three cost functions, that are all convex, depend on variables x_1, x_2 and x_3. The variable x_2 has influence on all three functions and this is the coupling that is to be relaxed by Lagrangian multipliers. The problem can be rewritten as

$$J = \max_{p_i} \min_{x_i, v_i} \big(f_1(x_1, v_2) + f_2(x_2) + f_3(x_3, v_3)\big). \tag{6.23}$$

Now it can be decomposed into five separate optimization problems:

$$\begin{aligned}
\text{Function 1:} \quad & \min_{x_1, v_1} \big(f_1(x_1, v_1) - \lambda_1 v_1\big), \\
\text{Function 2:} \quad & \min_{x_2, v_2} \big(f_2(x_2) + (\lambda_1 + \lambda_3) x_2\big), \\
\text{Function 3:} \quad & \min_{x_3, v_3} \big(f_3(x_3, v_3) - \lambda_3 v_3\big), \\
\text{Between 1 and 2:} \quad & \max_{\lambda_1} \big(\lambda_1 (x_2 - v_1)\big), \\
\text{Between 2 and 3:} \quad & \max_{\lambda_3} \big(\lambda_3 (x_2 - v_3)\big).
\end{aligned} \tag{6.24}$$

If the three functions above would minimize their own function, all three would probably have different values for x_2. Due to the Lagrangian multipliers or prices, λ_1 and λ_2, a consensus about the desirable value of x_2 among the functions can be achieved, because the agents can pay each other to modify the value. These Lagrangian multipliers or price variables are adjusted to take advantage of violations of the constraints; $x_2 = v_1$ and $x_2 = v_3$. The optimal values for the Lagrangian multipliers can be found using the subgradient search method, causing (6.24) to reach the global optimal of the original optimization problem [43],[42].

After decomposition the original problem is reformulated from a team-optimization problem to a non-cooperative game with additional players. The new players are market makers (last two lines of (6.24)) associated with state variables shared by agents. They adjust the prices to take advantage of violation of relaxed constraints [30]. In the next section the dual decomposition for MPC in a network of decision-makers is explained.

6.4 Dual Decomposition for general MPC schemes

The concept of dual decomposition from [43] in the previous section, is that dual variables are introduced in the optimization objectives. When variables of different subproblems are connected, the dual variables can be interpreted as prices in a market mechanism serving to achieve mutual agreement between the various subproblems. This method is applied to a general MPC scheme as in [17], [36] and [30]. The principles of MPC have already been introduced in *Section 4*. With MPC the optimization is solved, at each time step $t \in \mathcal{T}$ over a time horizon with step $k \in \mathcal{K}$, as defined in (4.1), to find the control sequence that minimizes the objective function over that prediction horizon. Where after the control input $u_i(t|t)$ for each agent in $i = 1, .., n$ is implemented.

6.4.1 General centralized MPC

Before the dual decomposition method is combined with MPC, a general centralized MPC problem is to be defined. A MPC objective function for a network of agents, $x \in \mathbb{R}^n$ for $i = 1, .., n$, can be defined as

$$V_u(t+k|t) = \sum_{i=1}^{n} \big(\hat{x}_i(t+k|t)\big)^2, \tag{6.25}$$

which is subject to prediction models, initial conditions and convex constraints. Therefore the generalized MPC problem becomes

$$\begin{aligned}
\min_{u} \quad & \sum_{k\in\mathcal{K}} V_u(t+k|t), && && \text{(6.26a)}\\
\text{subject to} \quad & \hat{x}(t|t) = x(t), && && \text{(6.26b)}\\
& \hat{x}(t+k|t) = A\hat{x}(t+k-1|t) + B\hat{u}(t+k-1|t), && \forall\ k \in \mathcal{K}\backslash\{0\} && \text{(6.26c)}\\
& \hat{x}(t+k|t) \in X, \hat{u}(t+k|t) \in U, && \forall\ k \in \mathcal{K} && \text{(6.26d)}
\end{aligned}$$

where $X, U \in \mathbb{R}$ are convex sets. Furthermore B is the $n \times n$ identity matrix, $B = I_{n\times n}$, and $A \in \mathbb{R}^{n\times n}$ is a stochastic adjacency matrix that has the following properties:

1. $A_{i,j} \geq 0$.
2. $A_{i,j} = 0$, if no connection of agent j to i.
3. $\sum_i A_{i,j} = 1$.

The matrix A is called the coupling matrix and due to its characteristics, in order to optimize, agent i needs to know the state $\hat{x}_j(t+k|t)$ of its connected neighbors, that itself depends on neighbors, therefore all agents are required to have access to the sequence of prediction states for each agent in the network. This central MPC can be distributed by dual decomposition and the subgradient method, whereby the dynamic constraints are decoupled and each agent can solve its own subproblems locally only with information of its connected neighbors [30].

Before the problem can be decomposed, the dynamic constraints in (6.26) need to be decoupled. First define $A_D = [a_{ii}]$ for $i = 1, .., n$ and $A_0 = A - A_D$. Now, a new variable is introduced; $\hat{v}(t+k|t) \in \mathbb{R}^n$, causing the coupling constraint to be

$$\hat{x}(t+k+1|t) = A_D\hat{x}(t+k|t) + \hat{v}(t+k|t) + B\hat{u}(t+k|t), \tag{6.27}$$

with additional constraint

$$\hat{v}(t+k|t) = A_0\hat{x}(t+k|t), \tag{6.28}$$

for all $k \in \mathcal{K}$ [36]. Which is used to augment the objective function, by a vector of weighted Lagrange multipliers $\hat{\lambda}(t+k|t) \in \mathbb{R}^n$, where $\hat{\lambda}$ can be seen as price signals between neighboring agents. This result in the centralized MPC dual function

$$\max_{\lambda}\min_{\hat{u},\hat{v}} \quad \sum_{k\in\mathcal{K}} \Big(V_{\hat{u}}(t+k|t) + \hat{\lambda}^T(t+k|t)\big(\hat{v}(t+k|t) - A_0\hat{x}(t+k|t)\big)\Big). \tag{6.29}$$

Strong duality between the primal function (6.26) and the dual function (6.29) is assumed. Therefore by minimizing the dual function over $\hat{u}(t+k|t)$ and $\hat{v}(t+k|t)$ and maximizing over $\hat{\lambda}(t+k|t)$ will obtain the same optimal value as the original problem.

6.4.2 General distributed MPC

To distributize the dual function, every agent needs to solve its own dual function (6.29). For each agent i the coupling constraint in (6.26) becomes

$$\hat{x}_i(t+k+1|t) = A_{ii}\hat{x}_i(t+k|t) + \hat{v}_i(t+k|t) + B_{ii}\hat{u}_i(t+k|t), \tag{6.30}$$

with additional constraint

$$\hat{v}_i(t+k|t) = \sum_{j\neq i} A_{ij}\hat{x}_j(t+k|t), \tag{6.31}$$

for all $k \in \mathcal{K}$ and $\hat{v}_i(t+k|t)$ can be seen as the influence agent i expects to receive from its neighbors [43]. The constraint (6.34) is added to the objective function by Lagrangian relaxation of Lagrange multipliers, $\hat{\lambda}_i(t+k|t) \in \mathbb{R}$ for $k \in \mathcal{K}$. By interchanging the sum over the agents $i = 1, .., n$ in (6.29), each agent i solves

$$\max_{\hat{\lambda}_i} \min_{\hat{u}_i, \hat{v}_i} \quad \sum_{k\in\mathcal{K}} \Big(V_{i,\hat{u}_i}(t+k|t) + \hat{\lambda}_i(t+k|t)\hat{v}_i(t+k|t) - \hat{x}_i(t+k|t)\Big[\sum_{j\neq i}\hat{\lambda}_j(t+k|t)A_{ji}\Big]\Big), \tag{6.32a}$$

$$\text{subject to} \quad \hat{x}_i(t|t) = x_i(t), \tag{6.32b}$$

$$\hat{x}_i(t+k|t) = A_{ii}\hat{x}_i(t+k-1|t) + \hat{v}_i(t+k-1|t) + B_{ii}\hat{u}_i(t+k-1|t), \qquad \forall\, k \in \mathcal{K}\backslash\{0\} \tag{6.32c}$$

$$\hat{x}_i(t+k|t) \in X_i, \quad \hat{u}_i(t+k|t) \in U_i, \qquad \forall\, k \in \mathcal{K} \tag{6.32d}$$

where X_i and U_i are convex sets. The structure of the original dual problem is preserved, while each agent solves its own separated contribution to the dual problem.

The Lagrangian multipliers are updated by the subgradient descent algorithm with *Momentum*, which has been introduced in *Section 6.2.2*. The dual function has only one element in its subdifferential, which means it is differentiable. Differentiability means that a small step size will yield convergence [5]. Only the information of connected agents $\hat{\lambda}_j(t+k|t)$ is needed to solve the MPC in (6.32), so that the prices are also updated in a distributed manner and thus the whole problem is distributed. The Lagrangian multipliers are updated according to

for all $k \in \mathcal{K}$

$$m^s_{avg,i}(t+k|t) = (1-\beta)m^{s-1}_{avg,i}(t+k|t) + \beta\big(\hat{v}^s_i(t+k|t) - \sum_{j\neq i} A_{ij}\hat{x}^s_j(t+k|t)\big), \tag{6.33a}$$

$$\hat{\lambda}^{s+1}_i(t+k|t) = \hat{\lambda}^s_i(t+k|t) + \gamma^s_i m^s_{avg,i}(t+k|t), \tag{6.33b}$$

Where s is the iteration counter and γ and β are the appropriately chosen constants, so that the Lagrangian multipliers converge to their optimum value.

In the distributed MPC scheme a local controller, present at each agent, makes decision only

depending on local information. The decision variables and prices are communicated between neighboring agents and determined iteratively. In this iterative process, each agent communicates their Lagrange multipliers over the prediction horizon to each of its neighbors. Whereafter the various agents can solve their decoupled minimization problem based on the neighboring prices and can communicate their decision variables over the control horizon to its neighbors. With those decision variables from its neighbors each agent can compute the prices, whereafter the loop repeats itself. In pseudo-code, this can be represented by

Algorithm 1 subgradient Method to Solve the Dual

given initial conditions λ and m_{avg}

repeat

Optimize (6.32) separately using the Lagrange multipliers.

Each agent sends the state variable to neighbors.

Update Lagrange multipliers according to (6.33).

Each agent sends Lagrange multipliers to neighbors.

Ideally, this repetition is done till,

$$\hat{v}_i(t+k|t) - \sum_{j \neq i} A_{ij} \hat{x}_j(t+k|t) = 0, \quad \forall k \in \mathcal{K} \tag{6.34}$$

is met. Which means that (6.34) and the solutions of the distributed MPC (6.32) are converged to the solutions of the centralized MPC (6.26).

6.5 Distributed MPC in the DC Microgrid

In accordance with *Section 6.4*, the dynamical system as represented in *Section 5.2.1* is approximated by dual decomposition and subgradient iterations. The DC microgrid is a coupled system, because the update of one node's variables also depends on information of the currents that flow between the neighbours in constraint (5.9b). The currents are shared between the nodes through the weighted Laplacian in the equation below. It is important to note that the weighted Laplacian represents the physical network, therefore the communication network between the controllers for distributed MPC must be identical to the physical network to represent interactions with the physical neighbors.

$$\hat{V}(t+k|t) = \hat{V}(t+k-1|t) + TC_s^{-1}\Big(\hat{I}_s(t+k-1|t) - I_L - \boldsymbol{\mathcal{B}R^{-1}\mathcal{B}^T}\hat{V}(t+k-1|t)\Big), \quad \forall k \in \mathcal{K}\backslash\{0\} \tag{6.35}$$

Firstly define,

$$A := I_{n\times n} - TC_s^{-1}(\mathcal{B}R^{-1}\mathcal{B}^T), \tag{6.36}$$

as the to be decoupled matrix, where $I_{n\times n}$ is the identity matrix. If the sampling time is smaller than the inverse of the highest element in weighted Laplacian,

$$T < 1/\max\{C_s^{-1}(\mathcal{B}R^{-1}\mathcal{B}^T)\}, \tag{6.37}$$

the matrix A has in discritized form stochastic characteristics, necessary for dual decomposition[2]. This is due to the fact the $C_s^{-1}(\mathcal{B}R^{-1}\mathcal{B}^T)$ is a weighted Laplacian matrix. By setting, T, smaller than the inverse of the maximum element, the Laplacian is scaled so that the diagonal entries have values in the range of $(0, 1)$. The off-diagonal entries have therefore values in the range of $(-1, 0)$. Noticeably subtracting this scaled weighted Laplacian from an identity matrix results in a stochastic matrix.

Again let $A_D = [a_{ii}]$ for $i = 1, .., n$, be the matrix of self-weights, where $A_0 = A - A_D$. Each agent introduces a local variable, which is a local guess that represents the neighbor influence on the relevant agent. The introduced vector of variables can be denoted as $v \in \mathbb{R}^n$, so that constraint (6.35) becomes

for all $k \in \mathcal{K}\backslash\{0\}$.

$$\hat{V}(t+k|t) = A_D\hat{V}(t+k-1|t) - \hat{v}(t+k-1|t) + TC_s^{-1}\Big(\hat{I}_s(t+k-1|t) - I_L\Big). \tag{6.38}$$

With additional constraint

$$\hat{v}(t+k-1|t) = A_0\hat{V}(t+k-1|t). \tag{6.39}$$

Therefore, for each agent i the neighbor influence is replaced by local guess $\hat{v}_i(t+k-1|t) \in \mathbb{R}$, constrained by

$$\hat{v}_i(t+k-1|t) = \sum_{j\neq i} A_{ij}\hat{V}_j(t+k-1|t), \quad \forall k \in \mathcal{K}\backslash\{0\} \tag{6.40}$$

and this portrays the expected influence of the connected neighbors. Then the problem is decomposed by applying a Lagrangian relaxation with respect to (6.40), with Lagrangian multiplier

[2] $A_{i,j} \geq 0$, $A_{i,j} = 0$ (if no information is sent from agent j to i) and $\sum_i A_{i,j} = 1$.

$\hat{\lambda}_i \in \mathbb{R}$. Furthermore, let $\mathcal{Q} = [q_{ii}]$ for $i = 1, .., n$, be a diagonal matrix consisting of weights (same holds for $\mathcal{R}_1, \mathcal{R}_2$). Which leads to the dual objective[3] for each node i,

$$\min_{\hat{u}_i, \hat{v}_i} \sum_{k \in \mathcal{K}} \Big(\sum_{i,j \in \mathcal{C}} \overbrace{\frac{1}{2} q_{ii} \big(w_i \hat{I}_{s_i}(t+k|t) - w_j \hat{I}_{s_j}(t+k|t) \big)^2}^{\text{Proportional current sharing}} \Big) + \overbrace{\hat{\lambda}_i(t+k|t)\hat{v}_i(t+k|t) - \hat{V}_i(t+k|t) \sum_{j \neq i} A_{ji} \hat{\lambda}_j(t+k|t)}^{\text{Dynamic decoupling}} +$$
$$\underbrace{\hat{u}_i^T(t+k|t) r_{1_{ii}} \hat{u}_i(t+k|t)}_{\text{Cost on controllable input}} + \underbrace{\big(\hat{v}_i(t+k|t) - v_i^\star \big)^T r_{2_{ii}} \big(\hat{v}_i(t+k|t) - v_i^\star \big)}_{\text{Voltage balancing}}. \tag{6.41}$$

Which is subject to all initial conditions and constraints, for each i, given by

$$\hat{I}_{s_i}(t|t) = I_{s_i}(t), \tag{6.42a}$$
$$\hat{V}_i(t|t) = V_i(t), \tag{6.42b}$$
$$\hat{I}_{s_i}(t+k|t) = \hat{I}_{s_i}(t+k-1|t) + TL_{s_i}^{-1} \Big(- R_{s_i} \hat{I}_{s_i}(t+k-1|t) - \hat{V}_i(t+k-1|t) + \hat{u}_i(t+k-1|t) \Big), \quad \forall\, k \in \mathcal{K} \backslash \{0\} \tag{6.42c}$$
$$\hat{V}_i(t+k|t) = A_{ii} \hat{V}_i(t+k-1|t) + \hat{v}_i(t+k-1|t) + TC_{s_i}^{-1} \Big(\hat{I}_{s_i}(t+k-1|t) - I_{L_i} \Big), \quad \forall k \in \mathcal{K} \backslash \{0\} \tag{6.42d}$$
$$\hat{V}_{min} \leq V_i(t+k|t) \leq \hat{V}_{max}, \quad \forall\, k \in \mathcal{K} \tag{6.42e}$$
$$\hat{u}_{min} \leq u_i(t+k|t) \leq \hat{u}_{max}, \quad \forall\, k \in \mathcal{K}. \tag{6.42f}$$
$$\tag{6.42g}$$

And where the Lagrange multipliers are updated according to the subgradient iterations

for all $k \in \mathcal{K}$

$$m_{avg,i}^s(t+k|t) = (1-\beta) m_{avg,i}^{s-1}(t-1) + \beta \big(\hat{v}_i^s(t+k|t) - \sum_{j \neq i} A_{ij} \hat{V}_j^s(t+k|t) \big), \tag{6.43a}$$
$$\hat{\lambda}_i^{s+1}(t+k|t) = \hat{\lambda}_i^s(t+k|t) + \gamma_i^s m_{avg,i}^s(t+k|t). \tag{6.43b}$$

Where γ is the manually set step size, β is momentum term and s is the iteration counter. The last two terms in (6.41) are called a penalty terms [2]. The second to last term in (6.41) is added to the objective function, so that the controller will capture the cost of the controllable input and this will help the system to reach steady state. The last term is added so that the controller establishes a form of voltage balancing from (3.9). Let, $v_i^\star$ be defined by a constant reference voltage $V_i^\star$ which is equal for all nodes, as $v_i^\star = \sum_{j \neq i} A_{ij} V_i^\star$. By virtue of this term, the objective function as defined will ensure that the system's load voltages will stay close to the desired voltage level of the overall network, depending on the weights given to identity matrix $\mathcal{R}_2$. Furthermore adding a penalty term in quadratic form to the objective function increases the convexity of a function. By increasing the convexity of the objective function the iterative descent algorithm's rate of convergence is favoured [3]. Due to the convexity of the original optimal control problem, the solution of (6.41) converges to the centralized solution in (5.7).

For each time step, t, the local controller computes a buck converter output voltage that serves as an input to the DGU concerned. Decisions by the controller are only made on local information, so no central entity is needed for reaching consensus in the DC microgrid. Distributed MPC is an iterative process, whereby each controller communicates only with its neighbors. The Lagrangian multipliers are initialized at zero and the auxiliary variable, v_i serves as a local guess

[3] A property of the Laplacian is that it can be expressed in quadratic form, $x^T \mathcal{L} x = \sum_{i,j \in \mathcal{C}} (x_i - x_j)^2$. Therefore the weighted Laplacian has the properties, $x^T W \mathcal{L} W x = \sum_{i,j \in \mathcal{C}} (w_i x_i - w_j x_j)^2$.

that estimates the expected influence of its neighbors' currents. After the first optimization the controller sends information on its currents to its physical neighbors and receives their respective currents. Thereafter the local controller can update the Lagrangian multipliers, by means of the subgradient algorithm, as a function of the error between the expected and the actual influence of the neighbors' currents. These Lagrangian multipliers are then sent to each connected DGU. With these Lagrangian multipliers, that act as prices, the local controller optimizes and a new v_i estimates the expected influence of the currents based on the new prices. Prices are adjusted to take advantage of violation of constraints (6.40). The controllers can now pay each other to modify v_i [43]. This process repeats itself until convergence; when the local guesses and the actual influence of external currents are identical. In *Algorithm 2* the pseudo-code for the distributed MPC in DC microgrids is given.

Algorithm 2 Distributed Model Predictive Control for DC Microgrid

Input: $I_{s_i}(t), V_i(t), I_L(t)$ where $1 \leq i \leq n$
Output: $u_i(t)$
Parameters: Iteration number $\{S_k\}$ and subgradient stepsize γ_i^r
for $t = 1, 2, ..., tmax$ **do**
  Initialize Lagrange multiplier $\hat{\lambda}_i^0(t+k|t)$ and moving average $m_i^0(t+k|t)$;
  for $s = 1, 2, ..., S_k$ **do**
    for $k = 0, ..., N-1$ **do**
      for $i = 1, 2, ..., n$ **do**
        Solve the optimization problem (6.41) ;
        Subject to (6.42) ;
      end for
      Each node sends $\hat{V}_i(t+k|t)$, $\forall k \in \mathcal{K}$ to connected nodes ;
      for $i = 1, 2, ..., n$ **do**
        subgradient update (6.43) ;
      end for
      Each node sends $\hat{\lambda}_i^{s+1}(t+k|t)$, $\forall k \in \mathcal{K}$ to connected nodes ;
    end for
  end for
  $u_i(t|t) = \hat{u}_i^{S_k}(t|t)$;
end for

6.5.1 Distributed Current Sharing

In the manner that the problem is proposed, the proportional current sharing objective introduced in *Section 3.1*, is incorporated in the objective function (6.41). To achieve proportional current sharing in the power system, the current is shared proportionally to the generation capacity of their respective energy source as expressed in (3.5). In the centralized MPC scheme, the controller has information on all system variables and according to the particular objective function the controller can optimize the controllable inputs so that current is proportionally shared in the DC microgrid.

In a distributed MPC based controller each i-th DGU cannot know the generated currents at their respective neighbour. Therefore these values have to be estimated, for similar reasons why the coupled dynamics are approximated by dual decomposition. Nevertheless the theory from [43] with dual decomposition and subgradient iterations is only applicable to coupling constraints. For that reason another estimating method has to be examined and in this research a non-iterative approach is considered (see *Scenario 7.5.7*). Each node i can use the actual system

output, $I_{s_j}(t)$, from its neighbours as an estimate for the generated current over the prediction horizon. Hence for each i-th DGU,

$$\hat{I}_{s_j}(t+k|t) = I_{s_j}(t|t), \quad k \in \mathcal{K} \tag{6.44}$$

represents the estimate for the neighbours' generated currents. Hereby the system is fully distributed and each MPC controller makes decisions only depending on local information.

6.5.2 Step Size

The chosen step size for subgradient descent is important as it makes sure that the distributed problem converges to the optimal solution. The step size for distributed MPC problems is usually set to $\gamma_i^s = \frac{\gamma_0}{\sqrt{s}}$, with the value γ_0 chosen by fine-tuning as in [30], [36]. However this is not the optimal strategy for updating the step size in a DMPC for DC microgrids. Because, if γ_i^s is chosen to be very small, it would take long time to converge and become computationally expensive. Whereas if the step size is assigned relatively large, it may fail to converge and overshoot the minimum [34]. Due to this trade-off between accuracy and speed, the allocation of an ideal step size for subgradient method in a DMPC for DC microgrid is difficult. Therefore the *Momentum* term is utilized for the subgradient based optimization, as outlined in (6.43). By the incorporation of this term, step size selection still requires fine-tuning, however faster convergence and reduced oscillation is achieved. Thereby increasing the range of step size values that give fast convergence with respect to regular subgradient iterations. Furthermore it is more robust to noisy subgradient information and therefore better suited for the DMPC of DC microgrids [59].

6.5.3 Stopping Criteria

If a centralized optimization problem is distributed by means of dual decomposition, it is important to keep the amount of communication between subsystems as small as possible. As has been depicted in *Algorithm 2*, every agent is taking turns to communicate its Lagrangian multipliers and currents to its neighbors. However, at each real time step, it might not be feasible to wait for the dual algorithm to converge and the algorithm might be needed to be terminated prematurely [48]. The number of iterations, s, before the distributed solution reaches its optimal value needs to be minimized. Although the number of iterations must be enough to give a feasible solution to the optimization and to guarantee stability of the closed loop system. Therefore a stopping condition, where the algorithm terminates early, that guarantees this property is required [19]. Use of a stopping criterion will therefore entail that the KKT conditions of *Section 6.2* are not satisfied. Varying stopping criteria have been utilized in literature. For example in [36] and [30] if the Lagrangian updates stay within a certain bound ϵ, the dual algorithm is terminated. In [48], it is figured that according to the receding horizon principle the only variable of interest is the first input. So the dual algorithm quits when the mean residual on a window of some iterations with respect to input u_i falls below a given threshold.

Chapter 7

Simulations

7.1 Introduction

The distributed MPC algorithm as described in *Section* 6.5 to control the network of nodes in a DC microgrid will be implemented and tested in this chapter. Simulations of different scenarios must give clarity on the topic. Hereby measuring the effect of the different parameters, variables and constrains in the optimization. Since this research has been done to validate the distributed MPC algorithm, the results will be compared to the solution of the centralized MPC algorithm as defined in *Chapter 5*.

The implementation is done in MATLAB with a *SDPT3* solver from the CVX package [20]. The problem is not implemented within parallel loops and therefore might not be the fastest way for optimization. However it suits to prove the concept of the distributed implementation. In this section firstly the model as implemented in MATLAB will be explained, whereafter the network will be defined and then the scenarios will be outlined. These scenarios will be tested and the results will be discussed afterwards.

7.2 Model

The model as implemented in MATLAB is presented in this section, so the results can be reproduced and further work can be done with it. To validate the distributed algorithm, the performance of the distributed controller is analyzed with respect to its centralized counterpart. The controllable buck converter output voltage generated by the MPC scheme serves as an input to the each respective DGU in the microgrid system. The model represents a feedback loop, where the outputs of the system serve as inputs for the controller, which then generate the input for the system based on the MPC scheme.

In short the model does the following. A **Network** is defined that consist of the nodes and their parameters. The **MPC** loops through time steps 1 to t_{max} with a prediction horizon of N. After each optimization, the inputs are sent to the **System**, which updates its system parameters according to the given input, whereafter the **MPC** is repeated for the next time step with new requirements. The bold words indicate the models' files:

- Network
- Centralized MPC
- Distributed MPC
- System

The model as a whole can be found in *Appendix B* and additional information on the parameters used in the model are portrayed there.

7.3 Network details

To test the MPC scheme for the DC microgrid, the proposed MPC scheme is applied to an example microgrid consisting of 4 DGUs as portrayed in *Figure 7.1.* The arrows indicate the positive direction of the currents through the network. Furthermore this network operates separately from the grid.

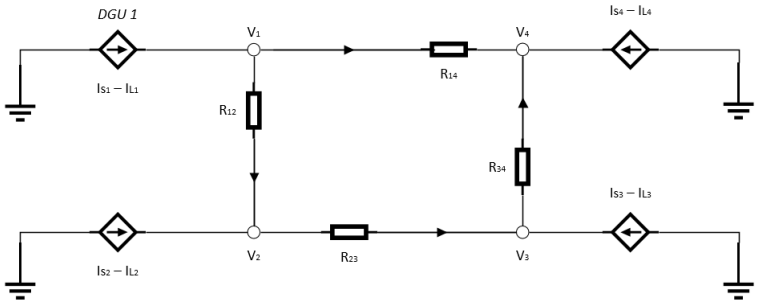

FIGURE 7.1: The microgrid with 4 buck converters used for simulations

The system parameters are adopted from [10] and [38] and can found in *Appendix A*. This Appendix will also give additional information on the system and other parameters used. The system will be initialized at steady state, then at some time instant, t, the load current changes denoted by $\Delta I_L := I_L(t^+) - I_L(t)$.

The simulations are done for $t_{max} = 100$ and the prediction horizon is $N = 4$, with sampling time $T = 0.0005s$. The sampling time for the discretization of the continuous system dynamics is set to this value so the coupling matrix has stochastic characteristics[4]. As mentioned these stochastic characteristics are a prerequisite for decomposition as proposed in [43].

$$\begin{aligned} t_{max} &= 100 \\ N &= 4 \\ T &= 0.0005s. \end{aligned}$$

Furthermore the centralized as well as the distributed MPC scheme optimizes an objective function over a prediction horizon. A compact formulation of the centralized objective function is

[4] $T < 1/\max\{C_s^{-1}(\mathcal{B}R^{-1}\mathcal{B}^T)\}$.

given by:

$$\min_{u,v} \sum_{k\in\mathcal{K}} I_s^T(t+k|t)W\mathcal{B}\mathcal{Q}\mathcal{B}^T I_s(t+k|t) + u^T(t+k|t)\mathcal{R}_1 u(t+k|t) + \Big(A_o V(t+k|t) - v^\star\Big)^T \mathcal{R}_2 \Big(A_o V(t+k|t) - v^\star\Big). \tag{7.1}$$

The weighted identity matrices ($\mathcal{Q}, \mathcal{R}_1$ and $\mathcal{R}_2$) that make up the objective function (7.1) determine the priority that is given to what specific goal. The weight assigned to these matrices influences in chronological order; proportional current sharing, cost on the controllable input and voltage balancing. In this research the following values are used:

$$\begin{aligned}\mathcal{Q} &= 10^1 I_{n\times n}\\ \mathcal{R}_1 &= 10^{-2} I_{n\times n}\\ \mathcal{R}_2 &= 10^1 I_{n\times n}.\end{aligned}$$

It is important to note that the chosen values also influence the convexity of the problem. By assigning the weights correspondingly, the problems becomes strictly convex. Strict convexity, in general, facilitates the subgradient algorithm to achieve a satisfactory convergence rate. Also the order of eigenvalues from these matrices should not differ some factor as this determines the condition number of the Hessian matrix. This can result in an ill-conditioned problem, which can possibly influence the subgradient descent's convergence rate. The values of the identity matrices are equivalent for the centralized and distributed objective function.

For distributed implementation it is chosen to set the step size in (6.43) to be $\gamma = 10$. By means of fine-tuning another step size could be chosen, that would increase the speed of convergence. However this step size is satisfying and suits to prove the concept of distributed MPC. The algorithm is terminated if the distance between the subgradient is smaller than $\epsilon = 0.001$.

$$\big|\hat{v}_i^s(t+k|t) - \sum_{j\neq i} A_{ij}\hat{V}_j^s(t+k|t)\big| < \epsilon, \quad \forall k \in \mathcal{K}, i \in \mathcal{V} \tag{7.2}$$

Other stopping criteria could also be used, but this condition provides a feasible solution. If the algorithm could not be terminated earlier, the maximum number of iterations is chosen to be $S_k = 1000$. Furthermore the Lagrangian multipliers and moving averages are all initialized at zero and the momentum term is set to $\beta = 0.5$.

7.4 Scenarios

Subgradient methods can be slow to converge or have poor convergence properties, because problems are not strongly convex or well-conditioned. Also the step size parameter in the gradient scheme must be chosen appropriately to get a good performance and the convergence rate is generally hard to determine in advance [18]. Moreover, dual decomposition is often performed on more simpler coupling dynamics. In this research the dual decomposition is performed on the physical network of the DC microgrid, which is substantially more complex and has varying parameters that can influence the subgradient algorithm. Therefore the distributed MPC is tested according to a few scenarios in which various parameters will be altered to see their effect. Hereby the characteristics of the distributed MPC scheme for DC microgrids will become evident.

In this section the different tests are outlined. If not explicitly expressed all values as stated in *Section 7.3* and *Appendix A* are used. The distributed and central solution are compared for the following scenarios:

Scenario 1 - Reference This scenario, with all variables left unchanged, will compare the centralized and distributed MPC algorithms for the DC microgrid. It will serve as a reference state for the other scenarios.

Scenario 2 - MPC prediction horizon (N) The prediction horizon is initially set at $N = 4$, which is a very short window. This scenario will give insight in how increasing the prediction horizon to $N = 8$, as the controller will take into account more future time instance, will affect the distributed MPC algorithm.

Scenario 3 - Priority for objective parameters ($\mathcal{Q}, \mathcal{R}_1, \mathcal{R}_2$) Allocating weights to these matrices will give priority to one of the three control objectives of *Section 3.1*. These parameters are altered in this scenario so that the voltage balancing is more prioritized than the proportional current sharing. In addition the influence of this modification on the subgradient algorithm is also analyzed.

Scenario 4 - Fixed number of subgradient iterations ($\mathbf{S}_k$) The sampling time is set to $T = 0.0005s$, in some situations it can not be feasible to wait for the distributed algorithm to converge. Therefore the algorithm might be needed to be terminated prematurely, before the stopping condition (7.2) is reached. It is interesting to see if the solution at the point of termination is close to the optimal solution.

Scenario 5 - Sampling Time (T) The continuous system dynamics are discretized according to a sampling time. This scenario will test the influence of the sampling time on the distributed MPC algorithm. The sampling time T is decreased from the initial $0.0005s$ to $0.00005s$.

Scenario 6 - Non-stochastic coupling matrix (A) In [43] the dual decomposition is performed on a stochastic optimal control problem. The coupling matrix that depicts the shared currents between nodes is stochastic by means of the initially chosen sampling time. This scenario will test if the dual decomposition for distributed optimization by [43] is also possible for a non-stochastic coupling matrix.

Scenario 7 - Estimating generated currents In the previous scenarios it is claimed that the generated currents from the neighbouring nodes of DGU_i are known at each of the local controllers. This is not fully distributed and therefore a strong assumption. In this scenario the neighbour's generated currents at each node are estimated.

7.5 Results

Additional information of the results with respect to the scenarios can be found in *Appendix C*. In the plots where the distributed and centralized solutions are studied; the distributed ones will be portrayed by a dotted line in the same color as its respective centralized counterpart. Furthermore the black dash-dot line in the load voltages plots denotes the average voltage for the system.

7.5.1 Scenario 1 - Reference

This scenario serves as a reference state for the other scenarios. So that the other scenarios can be compared to this one, on different aspects. This scenario will analyze the distributed and centralized MPC algorithms, for the conditions defined in *Section 7.3* and *Appendix A*. In the distributed implementation each DGU optimizes only based on local information, which is coordinated by the subgradient descent iterations as described in *Chapter 6*. The distributed solution converges to the centralized solution as the problem is convex. This is why the two control algorithms are compared in terms of their state trajectories corresponding to the generated currents (I_s), load voltage (V) and controllable inputs (u). Furthermore a plot will be given that displays the number of iterations necessary before the algorithm is terminated. The simulations run for $0.05s$, where at time instant $t = 0.01s$ the load demand changes.

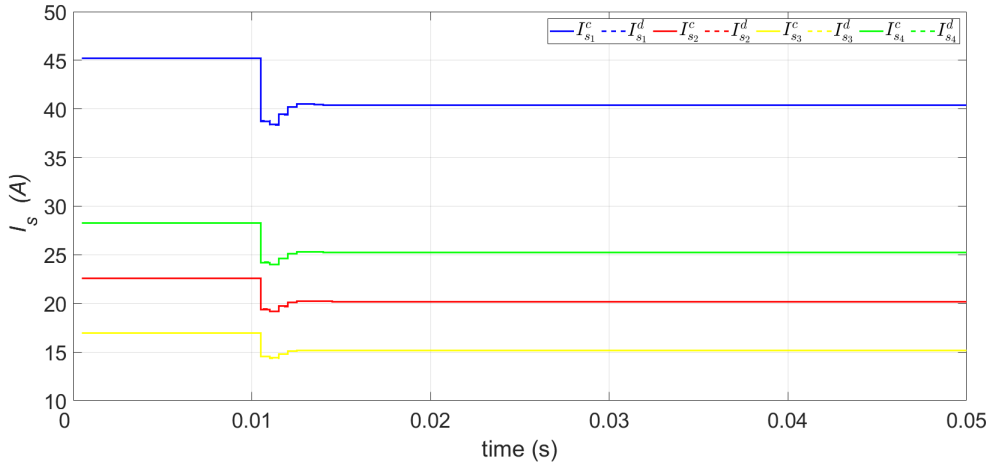

FIGURE 7.2: State trajectories for the generated current I_s - Scenario 1

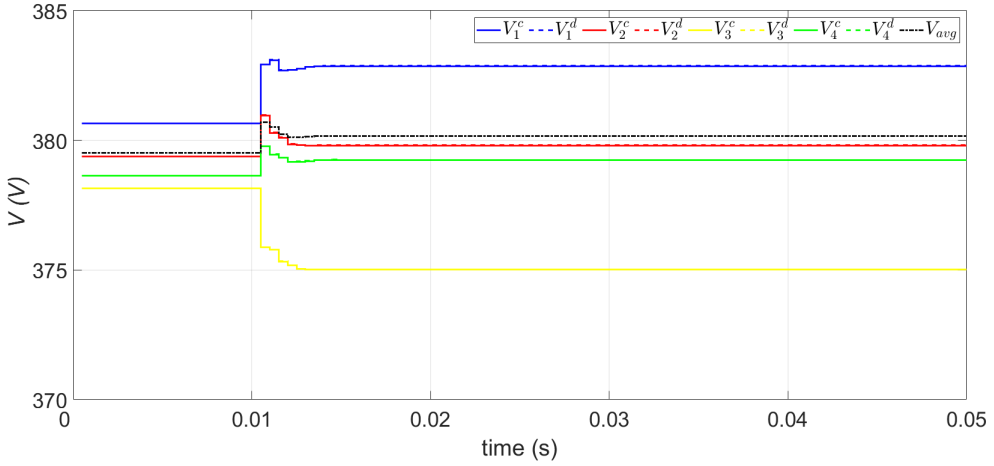

FIGURE 7.3: State trajectories for the load voltage V - Scenario 1

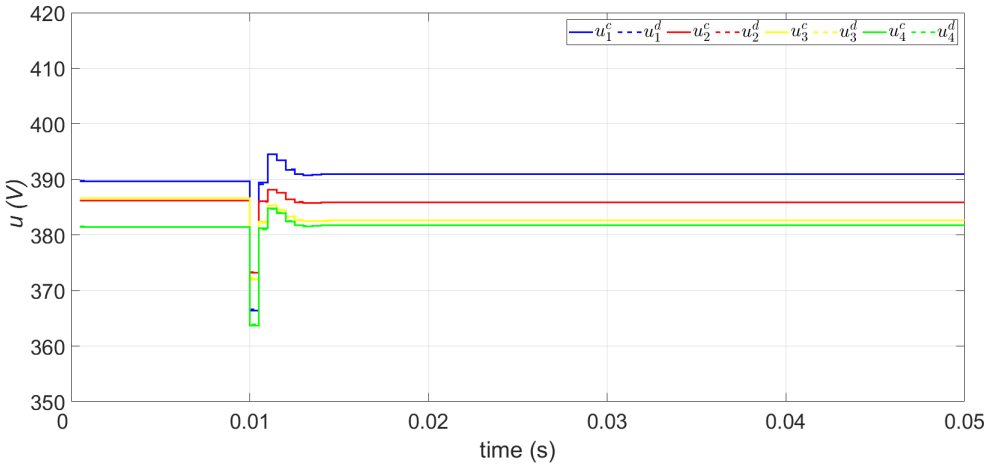

FIGURE 7.4: State trajectories for the control inputs u - Scenario 1

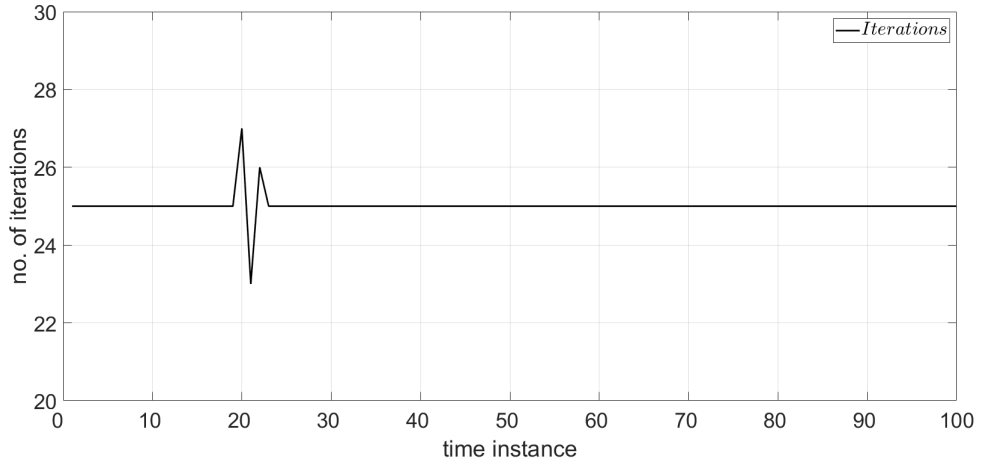

FIGURE 7.5: Number of iterations until termination - Scenario 1

Discussion and Conclusion

The figures above indicate that the distributed and centralized solutions are almost identical. The configuration of the weighted identity matrices make that the problem is convex, subsequently the distributed solutions coincide with those of to the centralized. This suggests that the stopping criterion, proposed in (7.2), allows for very good convergence upon termination. In addition, it can be seen from *Figure 7.5*, that the number of iterations needed for the distributed algorithm to convergence stays around 25. This is a very acceptable number of iterations taken into consideration the sampling time $T = 0.0005s$. The transient response from the change in load demand shows under-damped behavior and the settling time is about $0.0025s$. This could possibly be decreased by increasing the prediction horizon. Furthermore with this default setting, current sharing is achieved (see *Figure C.3*), however the load voltages deviate from $V^\star$. This can be amended by shifting the priority to voltage balancing, through increasing weight of the identity matrix $\mathcal{R}_2$. Altogether, this scenario proves that MPC via dual decomposition and subgradient iterations is a suitable design approach for reaching consensus in the DC microgrid in a distributed manner.

7.5.2 Scenario 2 - MPC prediction horizon

In this scenario the prediction horizon, N, of the MPC scheme is doubled, from 4 ($0.002s$) to 8 ($0.004s$). The effect of changing the horizon on the centralized as well as the distributed solutions is analyzed. The aim of this section is thereby twofold. Firstly it is to be determined how the incorporation of more future instances by the controllers affects the transient response for the centralized solutions and thus the distributed solutions as the problem is still convex. Secondly the effect of the increase in prediction horizon on the convergence properties of the distributed formulation is analyzed.

It is generally known that the length of the prediction horizon predominately determines the numerical effort in order to solve an optimal control problem in a MPC iteration [56]. While distributed control algorithms are commonly utilized to diminish the computational efforts, the prediction horizon chosen should not be too large. Nevertheless the stability and feasibility of closed-loop system must be ensured. Selecting the prediction horizon is for that reason often seen as a trade-off between computation time and accuracy of the result [30]. In other research on DC microgrids [38], [23], which have MPC-based control algorithms, the prediction horizon is chosen to be around $N = 4$ just as in the reference scenario.

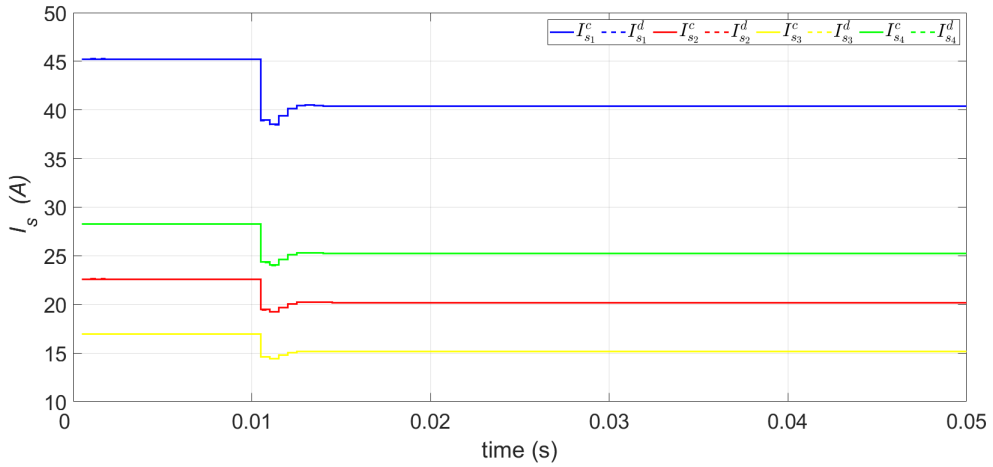

FIGURE 7.6: State trajectories for the generated current I_s - Scenario 2

Similar to the reference scenario, the distributed and centralized solutions are compared in terms of their state trajectories corresponding to the generated currents, load voltage and controllable inputs. It is important to note, that for the distributed solution to converge, the step size had to be lowered to $\gamma = 5$. Due to the increase of numerical computation complexity the distributed formulation could not converge by means of the subgradient descent algorithm with the previous step size ($\gamma = 10$).

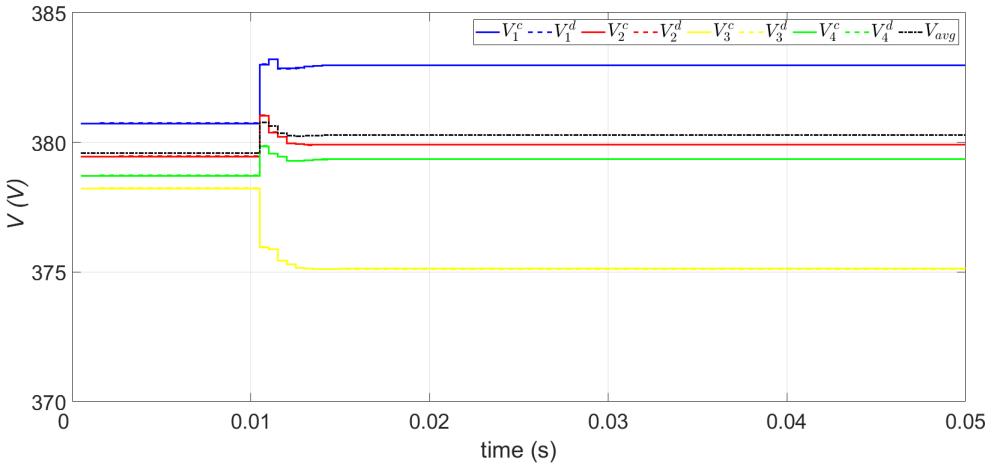

FIGURE 7.7: State trajectories for the load voltage V - Scenario 2

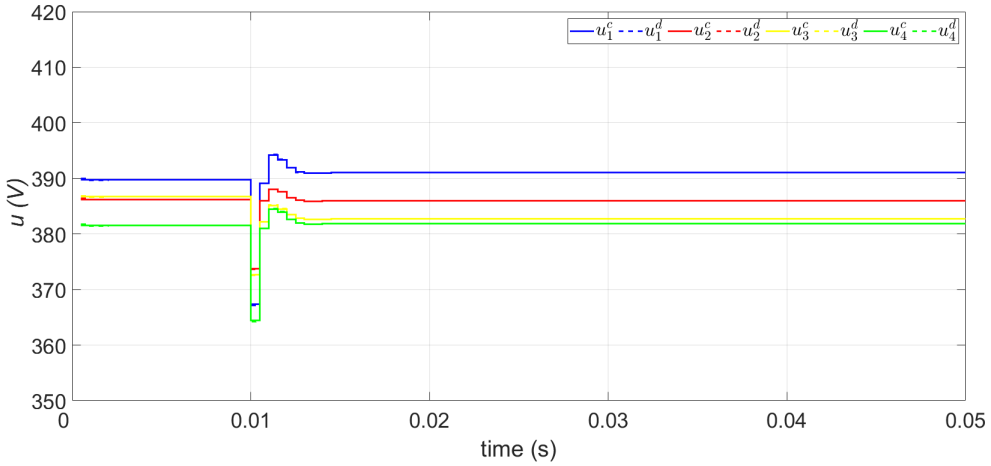

FIGURE 7.8: State trajectories for the control inputs u - Scenario 2

Discussion and Conclusion

The elongation of the prediction horizon for the MPC controllers has little effect on the performance of the system. As the plots for the state trajectories corresponding to the generated currents, load voltage and controllable inputs are compared to those in *Scenario 7.5.1*, it can be seen that the transient responses show little dissimilarities. They are not equivalent as the overshoot after the load demand variation is a bit tempered. While the system behavior is not much different, *Figure C.4* displays that current sharing is still achieved. In conclusion, a bigger vision into the future for MPC controllers, does not bring change to the system performance.

As for the convergence of the distributed solutions towards the centralized solutions. It has already been stated that the step size was decreased to $\gamma = 5$, due to the allocation of $N = 8$. *Figure C.5* shows this doubles the number of iterations upon termination with respect to $N = 4$. In addition the decline of the convergence rate could be induced by simply increasing the prediction horizon. In [14] it is said that the convergence rate of Lagrangian multipliers depends also on the convergence rate of the Lagrange multipliers under the responsibility of its neighbours. By that theory, as more Lagrangian multipliers are handled by increasing the prediction horizon,

this increases the communication between the nodes. As a result the numerical complexity is effectually increased, whereby the convergence rate is negatively influenced. Setting the prediction horizon correspondingly does not only slow down in terms of convergence rate, but increasing the prediction horizon inherently increases the computational efforts for the controllers. Whilst the computation cost of solving the MPC problem grows significantly with the length of the prediction horizon. Due to the fact that increasing the horizon has no significant impact on the system behavior, decreases the rate of convergence and increases the computational efforts it can be concluded that this modification does not have the desired effect. Therefore, $N = 4$, will be considered as the optimal prediction horizon.

7.5.3 Scenario 3 - Priority for objective parameters

Assigning the weights to the objective parameters $(\mathcal{Q}, \mathcal{R}_1, \mathcal{R}_2)$ will determine the priority that is given by the MPC controller to proportional current sharing or voltage balancing. The weights will be identical to all nodes as it has been in the previous scenarios. The aforementioned priority depends mainly on the ratio between Q and R_2. In this scenario four simulations are run to indicate the impact of these ratios on the voltage balancing and current sharing. The following values will be used for the weighted matrices; $\mathcal{Q} = \mathcal{R}_1 = 10^{-2} I_{n\times n}$. Whereas the values for $\mathcal{R}_2$ will alter according to the following schedule.

Scenario 3a : $\mathcal{R}_1 = 10^1 I_{n\times n}$ and $\gamma = 10$. [5]

Scenario 3b : $\mathcal{R}_1 = 10^2 I_{n\times n}$ and $\gamma = 20$.

Scenario 3c : $\mathcal{R}_1 = 10^3 I_{n\times n}$ and $\gamma = 35$.

Scenario 3d : $\mathcal{R}_1 = 10^4 I_{n\times n}$ and $\gamma = 40$.

One must keep in mind that increasing the ratio between the matrices will increase the condition number of the Hessian matrix. Research in, [40], [9] and [1], has indicated that an ill-conditioned objective function slows down convergence of descent methods for optimization. Therefore the aim of this scenario is twofold again. First off it is to be determined what the effect of the different ratios, $\frac{R_2}{Q}$, is with respect to the correlation between voltage balancing and distributed current sharing. Secondly this scenario's purpose is to establish a relation between the condition number of the Hessian matrix of the objective function, due to allocating the weighting parameters, and the speed of convergence for the distributed formulation.

The distributed against the centralized state trajectories corresponding to the load voltage will be displayed (figures for the generated current and controllable inputs can be found in *Appendix C.3*). Also a plot portraying the proportional current sharing at with respect to each node will be incorporated. The simulations are initialized at their respective steady states and the simulation time is set to $t = 0.01s$, where at time step $t = 0.005s$ the load demand is changed. In this scenario the simulation time is shortened with respect to the previous scenarios as it suits to prove the claim.

[5] The step sizes in this section are determined by fine-tuning and are close to their respective maximum value.

Scenario 3a

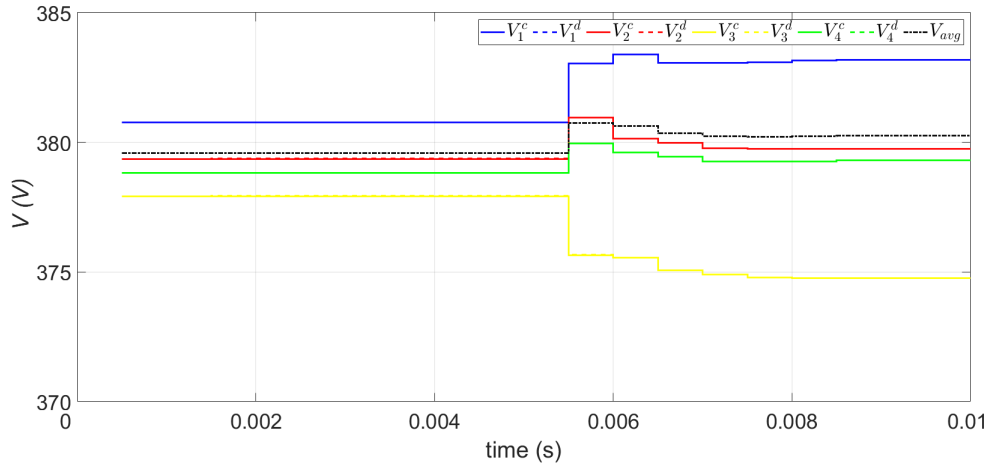

FIGURE 7.9: State trajectories for the load voltage V - Scenario 3a

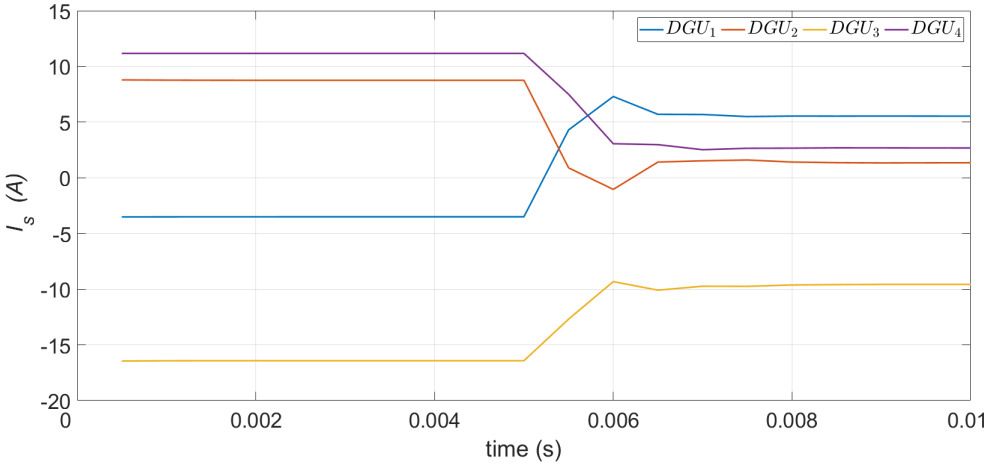

FIGURE 7.10: Current sharing at each DGU ($\mathcal{L}WI(t)$) - Scenario 3a

It is desired for the voltages at each node to be close to the desired system voltage $V^\star$. From *Figure 7.9* it can be seen that the voltages at the nodes deviate from this reference value with $\pm 5V$. Therefore by setting the quotient, $\frac{R_2}{Q} = 1000$, voltage balancing is not achieved. Furthermore the impact of assigning the objective parameters in such a way implies that preference for current sharing is reduced with respect to the reference state (see *Figure 7.10*).

Scenario 3b

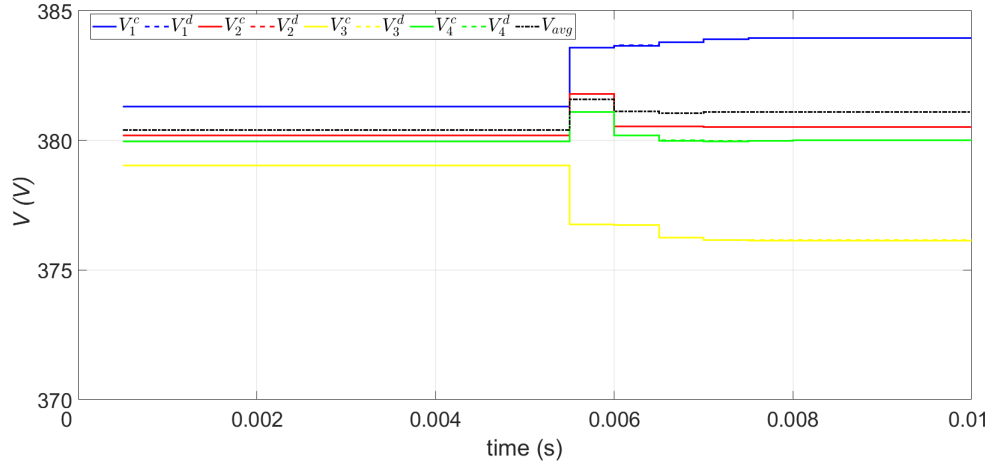

FIGURE 7.11: State trajectories for the load voltage V - Scenario 3b

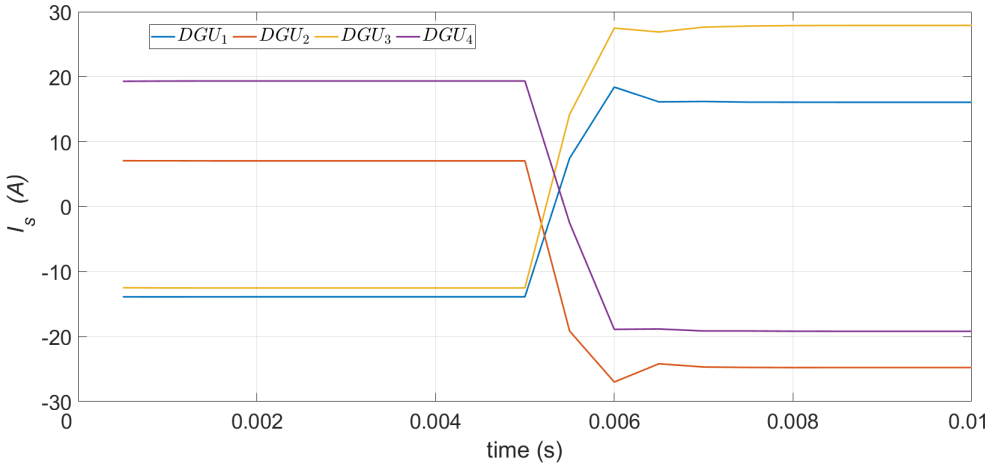

FIGURE 7.12: Current sharing at each DGU ($\mathcal{L}WI(t)$) - Scenario 3b

When $\frac{R_2}{Q} = 10,000$, the figures indicate that voltage balancing has increased relative to the previous scenario. The side effect is that the imbalance for current sharing at each DGU has also increased. By increasing the priority for voltage balancing in the objective function, current sharing automatically is less prioritized. This demonstrates that there is negative correlation between voltage balancing and current sharing.

Scenario 3c

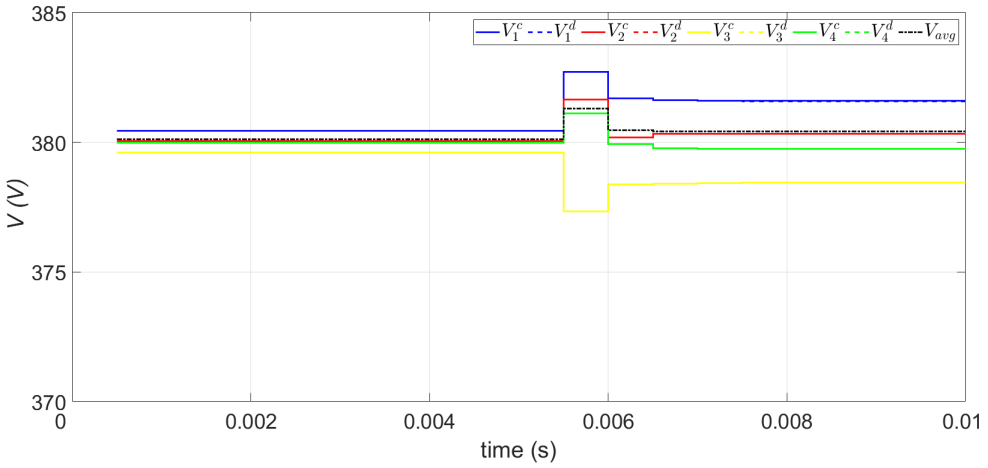

FIGURE 7.13: State trajectories for the load voltage V - Scenario 3c

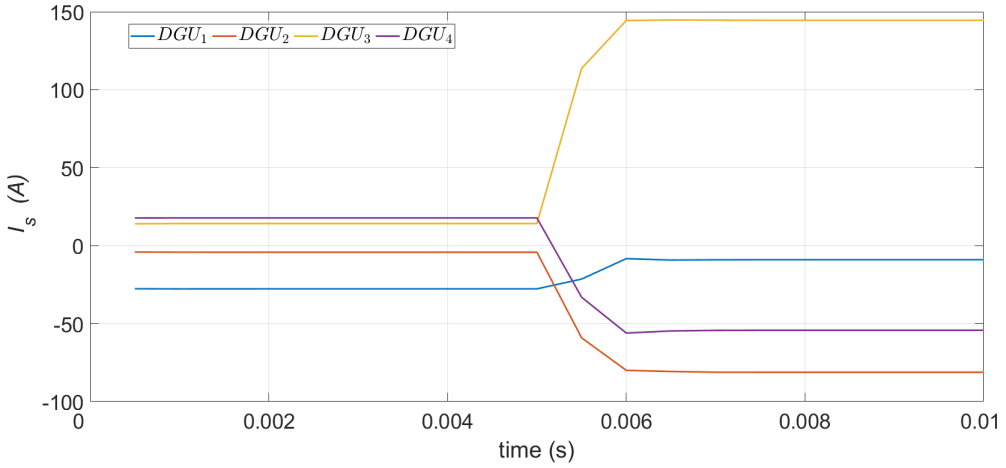

FIGURE 7.14: Current sharing at each DGU ($\mathcal{L}WI(t)$) - Scenario 3c

The ratio between the weighted matrices is set to $\frac{R_2}{Q} = 100,000$. At this point, the balancing of the voltages around $V^\star$ is improved. The voltages stay within $\pm 2V$ of the desired reference voltage. This indicates that voltage balancing is sufficiently attained and again the negative correlation between voltage balancing and current sharing is showed. Because while the voltage balancing at the DGUs is increased, the current sharing is decreased with respect to previous two scenarios.

Scenario 3d

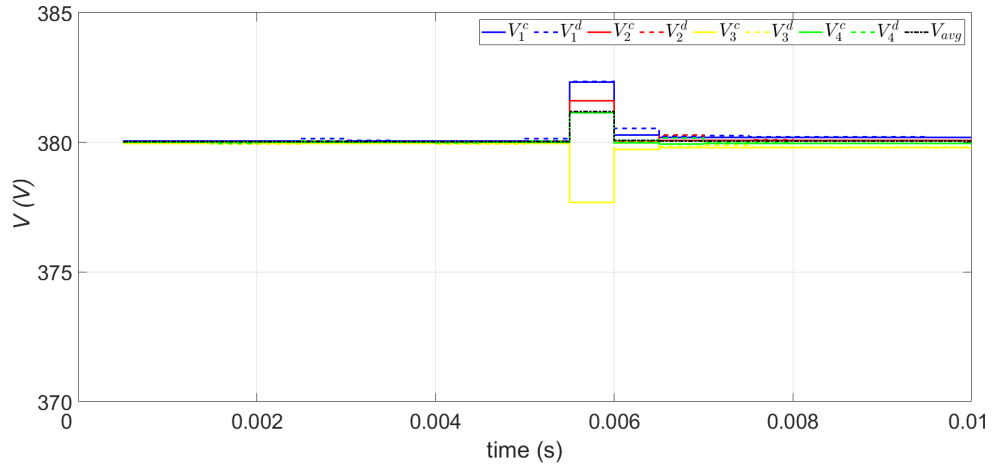

FIGURE 7.15: State trajectories for the load voltage V - Scenario 3d

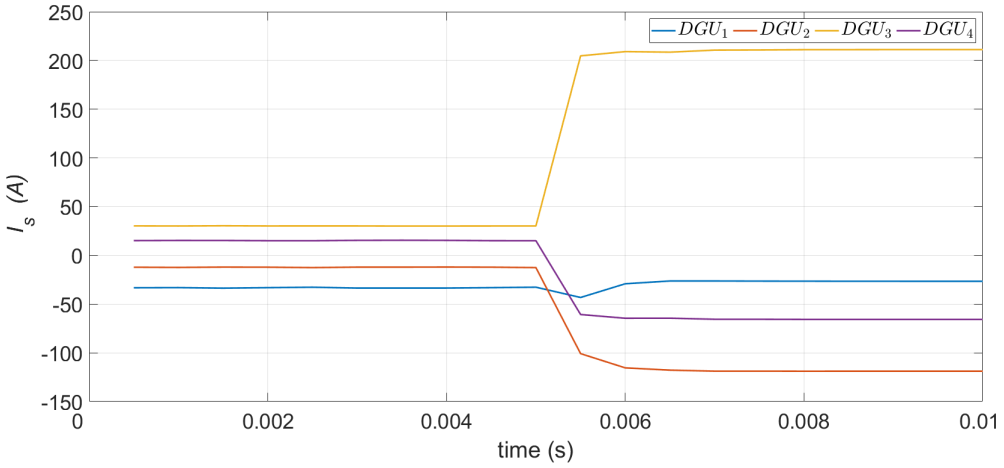

FIGURE 7.16: Current sharing at each DGU ($\mathcal{L}WI(t)$) - Scenario 3d

In this scenario, $\frac{R_2}{Q} = 1,000,000$, the voltage balancing objective is hugely favoured. *Figure 7.15* shows that voltage balancing at each node is achieved. Except when the load demand changes there is a transient response, however in two time steps the DGU load voltages are again at the desired reference voltage. The requirement for $\overline{V} = V^\star$ does not permit for current sharing [10], that's why *Figure 7.16* displays such an imbalance in proportional current sharing at each DGU.

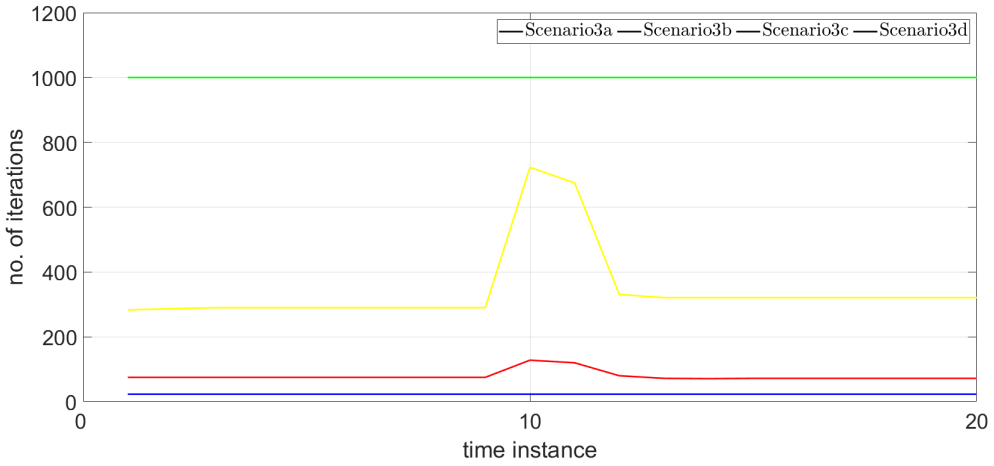

FIGURE 7.17: Number of iterations until termination - Scenario 3

The figure above portrays the number of iterations necessary for each time step of the aforementioned scenarios until the stopping criterion is reached. The stopping criterion is defined in (7.2) and the maximum number of iterations,S_k, is set at 1000. It can be concluded that the number of iterations increases as the ratio, $\frac{R_2}{Q}$, expands. In the first three scenarios the convergence behavior of the distributed solutions coincides with the centralized formulation. Opposite to the last scenario, where the pre-set maximum number of iterations is reached before the stopping condition is satisfied. *Figure C.12* and *Figure C.13* portray the trajectories of the generated current and control inputs for this scenario. These figures suggest that the distributed solution is close to the optimum upon termination.

Discussion and Conclusion

Different ratios between the weighted identity matrices, R_2 and Q, have been tested. These parameters illustrate the significance the MPC controller allocates to voltage balancing and proportional current sharing. To sum up, as the ratio, $\frac{R_2}{Q}$, increases the goal of voltage balancing is progressively achieved. When the quotient is set to $1,000,000$, all load voltages are almost identical to the desired reference voltage. Nonetheless this comes at a cost of the proportional current sharing, while as the voltage balancing is accomplished the imbalance in proportional current sharing in the system grows. This scenario has shown that the requirement for $\overline{V} = V^\star$ does not permit for current sharing.

Furthermore by increasing the ratio between the aforementioned matrices the Hessian matrix' condition number also increases. The more ill-conditioned the problem becomes, the more iterations are necessary to provide convergence. Therefore the condition number of the Hessian certainly determines the subgradient descent's speed of convergence and as the condition number is increased, the convergence rate declines. This correlation is depicted in *Figure 7.17*. The figure reveals that as the ratio is increased by factor 10, the number of iterations needed to reach the termination condition grows.

7.5.4 Scenario 4 - Fixed number of subgradient iterations

In *Scenario 7.5.1* it has been concluded that the stopping condition for the subgradient iterations, (7.2), guaranteed a good level of performance. Upon termination the solutions of the distributed algorithm are almost identical to those of the centralized formulation. This is due to the fact that the tolerance for the error is set to a very low value. In some situations it might not be feasible to wait for the distributed algorithm to reach this criterion. Especially considering the low sampling time. As the sampling time is set to $T = 0.0005s$, the distributed algorithm in some instances is needed to terminate prematurely before this stopping criteria could be attained.

For example, in the last scenario from *Scenario 7.5.3*, the subgradient descent required more than 1000 iterations to obtain the prescribed stopping condition. While, as a result of the scaling between parameters in the objective function, the convergence speed declined. However it was determined that the distributed solution was close to the optimum at the time of termination. When using dual decomposition methods the convergence behaviours of the dual iterations does not necessarily easily coincide exactly to that of the centralized formulation [17]. Therefore it is interesting to see what happens if the algorithm is terminated prematurely and if a form of suboptimality could be attained. In this scenario the same default settings as in *Scenario 3d* are used, only the maximum number of subgradient iterations is altered to $S_k = 100$. A simulation is run for $t = 0.025$, where at $t = 0.01$ the load demand varies. Below a figure of the state trajectories corresponding to the controllable inputs (u) for the distributed and centralized MPC algorithms can be found (state trajectories for generated currents and load voltage in *Appendix C.4*).

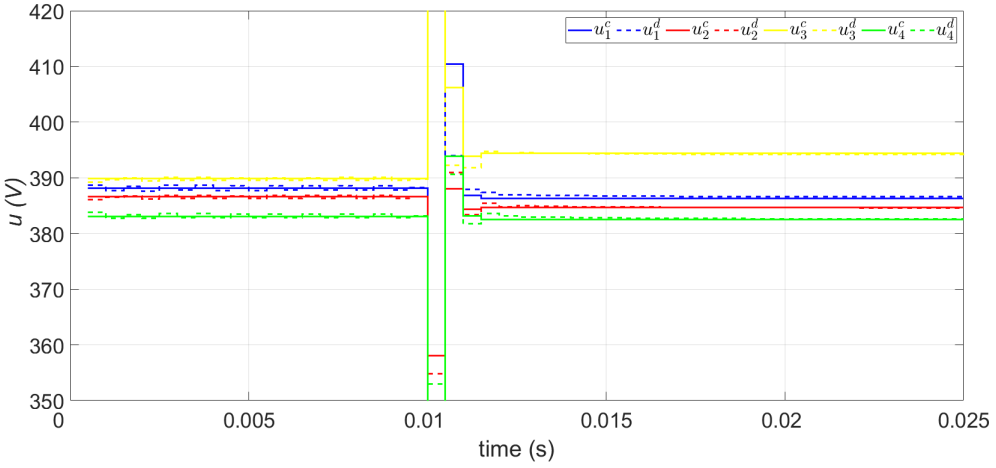

FIGURE 7.18: State trajectories for the control inputs u - Scenario 4

The figure suggests that the distributed controllable inputs are within close distance to the optimal solution. As mentioned the convergence, for this configuration, to the exact optimal solution is slow. Nonetheless the distributed solutions get close to the optimal solutions relatively fast.

Discussion and Conclusion

Time restrictions make it unfeasible to wait for the distributed algorithm to converge to the optimal solution. This scenario showed that the convergence of some optimization problem can be moderate due to i.e. increased ill-conditioning of the objective function. But that the

distributed algorithm would get close to the optimal solution relatively fast. Therefore the distributed algorithm could be prematurely terminated and still achieve a form of sub-optimality. So for ill-conditioned problems the subgradient based optimization can achieve sub-optimality within a reasonable number of iterations. However a problem with this theory is that it is hard to make any statement on how close the solution is to the optimum if the distributed algorithm is terminated prematurely in this manner. Future research could go into more detail and establish a termination guideline that indicates how many iterations are needed to ensure a certain suboptimality guarantee, i.e., distance to optimality [14].

7.5.5 Scenario 5 - Sampling time

In this scenario, the sampling time, T, is decreased from $0.0005s$ to $0.00005s$. The sampling time is required for the discretization of the DC microgrid system (3.1), so that an MPC scheme could be designed [38]. Generally the sampling time has to be chosen sufficiently small to give an exact representation of the dynamical system. Therefore the sampling time is adjusted to test how this influences the convergence behaviour of the distributed control algorithm.

The stochastic features of the decoupling matrix remain and the same objective parameters from the reference state are used in this simulation. This configuration showed strictly convex characteristics and caused the problem to be well-conditioned, allowing for good and fast convergence. The system will be initialized at steady state conditions. Furthermore the simulation will entail only 1 time step, wherein the trajectory of objective function of the distributed as an error of the centralized algorithm's objective function is plotted. It is chosen to test like this as it suits to prove the theory. Different step sizes will be utilized to show the implications of adjusting the sampling time like this.

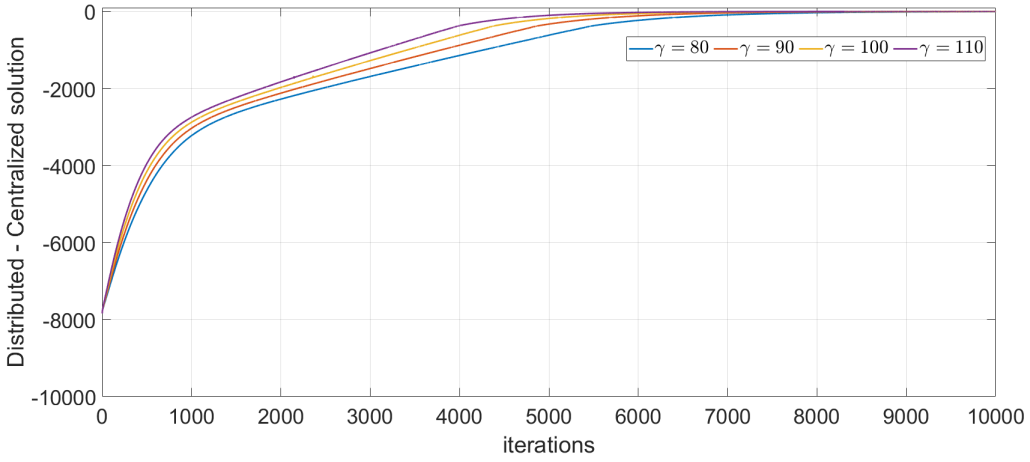

FIGURE 7.19: Evolution of the distributed objective solution vs. the centralized solution for one time step

The maximum number of iterations for this scenario is set at $10,000$. The simulations for the four step sizes have not been terminated and thus the stopping condition (7.2) has not been satisfied. With the new sampling time, the convergence goes extremely slow. Particularly if the figure is compared to the the same trajectory of the reference state, see *Figure C.1*, which

is also at steady state. Furthermore the figure demonstrates that the speed of convergence is improved by increasing the step size. However if the step size is further enlarged to $\gamma = 120$, the subgradient algorithm is not able to converge to the optimal solution (see *Figure C.16*). This implicates that by $\gamma = 110$ the fastest convergence is achieved and it still takes more than $10,000$ iterations to provide a good coincidence of the distributed and centralized formulation.

It is important to note that with the configuration of sampling time, $T = 0.00005$, the solution does not get close to the optimal solution relatively fast as in *Scenario 7.5.4*. In the previous scenario the subgradient descent had trouble finding the exact optimal solution, yet it converged within 100 iterations to a distance close to optimality. That this argument does not hold for this scenario is portrayed by the figure below. The figure portrays the error between the distributed and centralized controllable input, again for 1 time step and with step size $\gamma = 110$.

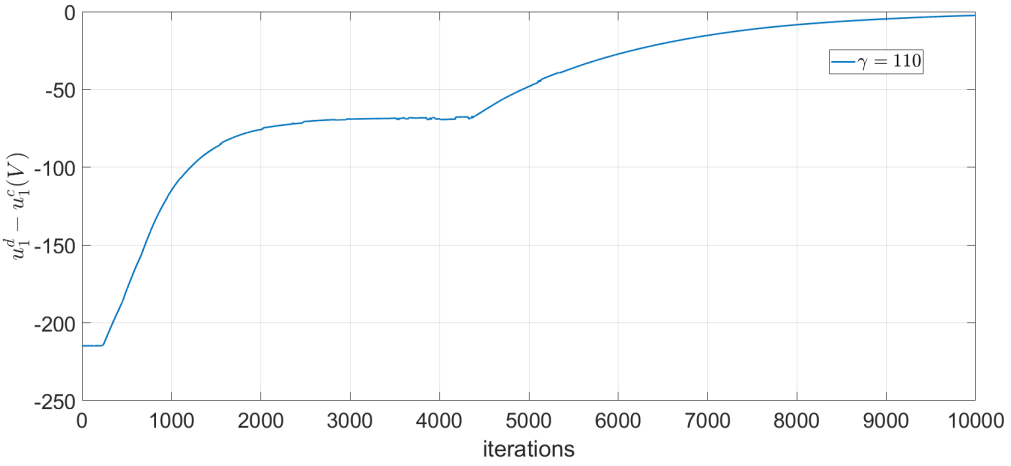

FIGURE 7.20: Evolution of the distributed vs. the centralized control input for one time step at DGU_1

The figure implicates that the distributed controllable input only gets close to the optimal value around iteration $10,000$.

Discussion and Conclusion

It can be concluded from this scenario, by setting the sampling time to 0.00005, with the default settings for other parameters as specified in *Section 7.3* and *Appendix A*, the subgradient algorithm will take a very long time to converge. This is unfavourable, while as the sampling time is set to this aforementioned value the controllers must have substantial processing power. Due to the fact, that the approximately $10,000$ subgradient iterations are needed to ensure a form of sub-optimality, discretizing the microgrid system dynamics with a sampling time, $T = 0.00005s$, is practicably infeasible.

A great part of this research has been dedicated to experiment with different default settings, for example changing the line resistances, to increase the speed of convergence with the chosen sampling time. However, consistently the same behaviour was encountered. In some instances the number of iterations upon termination was brought down, but these results are still considered infeasible. As subgradient descent is a heuristic method, it is hard to put the finger on the exact reason for this convergence behaviour. The condition number of the Hessian matrix

is identical to that of the reference state, thus ill-conditioning of the problem is excluded as a cause for this behaviour. The presumption is that by assigning the sampling time like this, the optimization problem becomes badly scaled. Further research could be aimed at increasing the rate of convergence for this problem by accelerated gradient methods.

7.5.6 Scenario 6 - Non-stochastic coupling matrix

In this research the original centralized MPC scheme for DC microgrids, [38], is decomposed into distributed subproblems. The subsystems employ distinct MPC controllers that only solve local control problems using local information from neighboring subsystems. In [43] the dual decomposition is performed on a stochastic control problem and in [15], [35], [36] and [30] a stochastic information matrix was decoupled. These information matrices naturally have stochastic properties. In this research the communication between the localized MPC based controllers is equivalent to the physical network, in the previous scenarios the coupling matrix that represented the shared currents between nodes was stochastic by means of the initially chosen sampling time. The sampling time, T, was required to be smaller than the inverse of the maximum value of the weighed Laplacian representing the network. Now, this scenario's aim is to determine if the dual decomposition theory with subgradient iterations for distributed optimization proposed in [43] is also suitable for non-stochastic coupling matrices.

The coupling matrix, A, is defined by $A = I_{n\times n} - TC^{-1}\mathcal{B}R^{-1}\mathcal{B}^T$. With $T = 0.007s$ and all default settings as in the reference state, the coupling matrix becomes

$$A = \begin{bmatrix} 0.0152 & 0.4545 & 0 & 0.5303 \\ 0.4545 & -0.0909 & 0.6364 & 0 \\ 0 & 0.6364 & -0.0341 & 0.3977 \\ 0.5303 & 0 & 0.3977 & 0.0720 \end{bmatrix}$$

This matrix has negative entries and therefore lost it stochastic properties. The non-stochastic matrix serves as the coupling matrix and is decomposed as described in *Section 6.5*. A simulation is done in order to test if the dual decomposition theory with subgradient iterations can be performed on non-stochastic coupling matrices. The distributed solution should converge to the centralized solution as the problem is still convex. Therefore the distributed and centralized control algorithms are compared in terms of their state trajectories corresponding to the controllable inputs (other state trajectories can be found in *Appendix C.6*). The simulations represent $0.07s$, where at time instant $t = 0.007s$ the load demand changes. The step size is set to, $\gamma = 3$, whereas all other parameters are identical to those in the reference state.

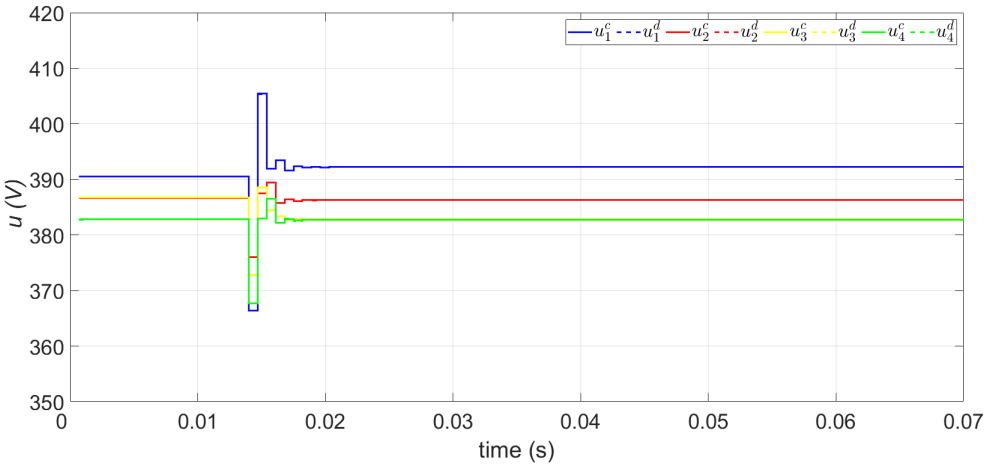

FIGURE 7.21: State trajectories for the control inputs u - Scenario 6

Discussion and Conclusion

The figure suggests that the distributed and centralized solutions are almost identical. From this scenario can be concluded that non-stochastic coupling matrices are applicable for dual decomposition techniques with subgradient iterations to achieve a distributed MPC scheme for DC microgrids. It must however be noted that the theories in [43] specifically prescribe stochastic control problems. The fact that the dual decomposition works for some non-stochastic coupling matrices in DC microgrid does not mean it holds for all non-stochastic matrices. Furthermore the number of iterations necessary to reach the stopping condition was around 40. It could be seen that the sampling time could be increased without having an effect on the speed of convergence. As opposed to the previous scenario where a decrease in sampling time resulted in an extremely slow convergence rate. Nonetheless it must be stated that increasing the sampling time influences the control quality. Each sampling time, through the system model, the controller calculates the system output from that time instance until the prediction horizon. Therefore the sampling time is generally chosen sufficiently small to give an exact representation of the dynamical system.

7.5.7 Scenario 7 - Estimating generated currents

In this scenario a fully distributed MPC scheme for DC microgrids will be tested. In the previous scenarios it has been claimed that each DGU knew the exact values of its neighbours generated currents. This is a strong assumption and means that the system is not perfectly distributed. To achieve a fully distributed MPC, the neighbouring generated currents for each DGU need to be estimated as explained in *Section 6.5.1*. It was proposed take the actual system output from its neighbours as an estimate for the generated current over the prediction horizon. Hence for each i-th DGU,

$$\hat{I}_{s_j}(t+k|t) = I_{s_j}(t|t), \quad k \in \mathcal{K} \tag{7.3}$$

represents the estimate for the neighbours' generated currents.

It is expected that the convergence behaviour for the distributed formulation with this new configuration is similar to that of the previous scenarios. On the contrary, the system behaviour is predicted to be different to that of the earlier scenarios. To test this, again the distributed and

centralized MPC schemes are compared in terms of their state trajectories corresponding to the generated currents, load voltages and controllable inputs. The simulations run for $0.05s$, where at time instant $t = 0.01s$ the load demand changes. The system is initialized at steady state and the step size for the subgradient algorithm is, $\gamma = 20$. It must be noted that the default setting for the parameters in the objective function are; $Q = 10^{-1}I_{n\times n}$, $R_1 = 10^{-2}I_{n\times n}$ and $R_2 = 10^{1}I_{n\times n}$.

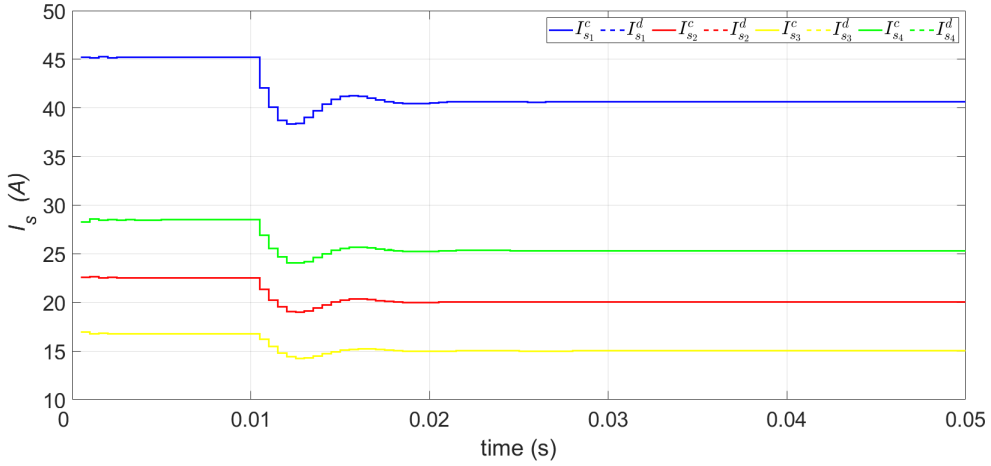

FIGURE 7.22: State trajectories for the generated current I_s - Scenario 7

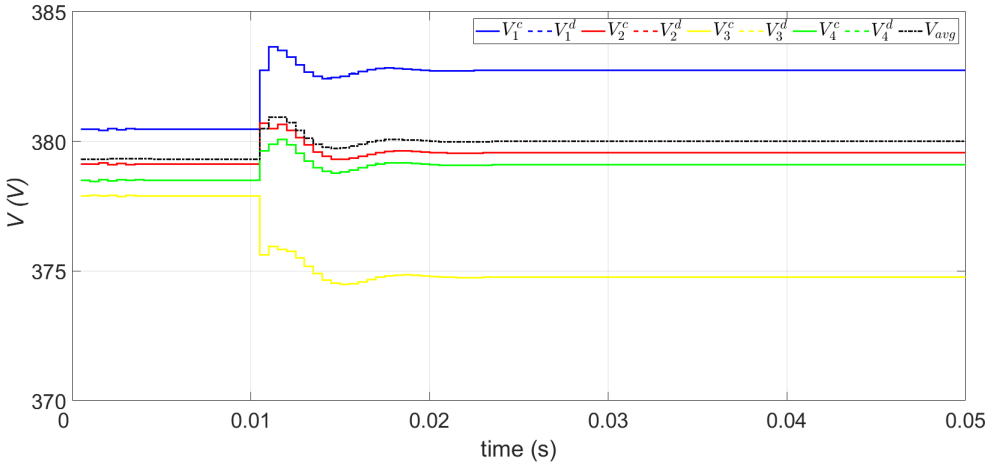

FIGURE 7.23: State trajectories for the load voltage V - Scenario 7

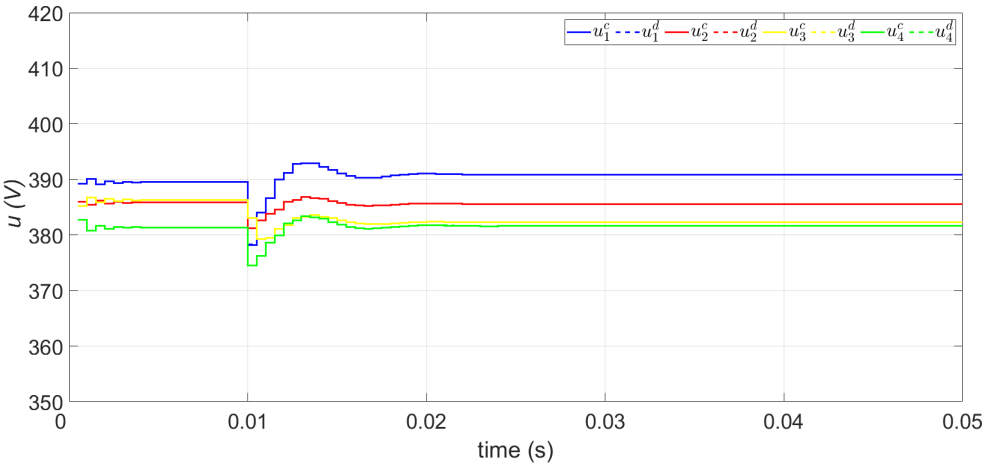

FIGURE 7.24: State trajectories for the control inputs u - Scenario 7

The figures above indicate that after the load demand variation the settling time is about $0.01s$. Which is an increase with respect to *Scenario 7.5.1.* Furthermore a plot will be given that displays the proportional current sharing with respect to each node.

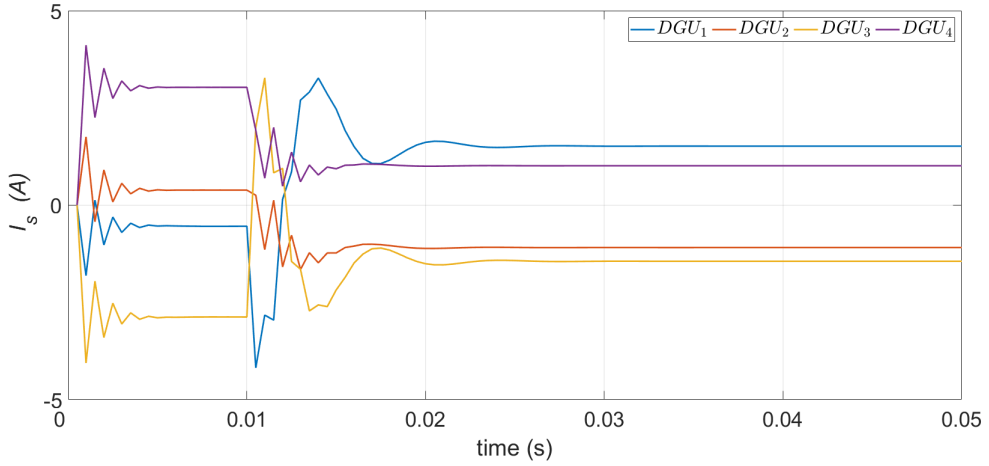

FIGURE 7.25: Current sharing at each DGU ($\mathcal{L}WI(t)$) - Scenario 7

Suggesting that the microgrid achieves a form of proportional current sharing as well as it reaches steady state.

Discussion and Conclusion

The distributed and centralized formulation seem to coincide and the number of iterations upon termination was around 20. The objective functions at each node do not change within this new configuration, therefore the convergence behaviour is similar to that of the previous scenarios. Accordingly all previous outcomes with respect to the convergence characteristics are relevant to this new configuration as well.

Nonetheless, the system behaviour is affected by estimating the generated currents in this new configuration. The way in which the generated currents are estimated brings extra uncertainty into the system, as a result the transient response is longer as well. Furthermore the default

setting for the objective parameters does not exclusively have an effect on the priority for current sharing and voltage balancing (see *Scenario 7.5.3*). Because the ratio between the parameters also influences the settling time after a change in load demand. In *Appendix C.7* two state trajectories for the generated currents are given with different default settings. They indicate that as the ratio becomes smaller, and current sharing is more favoured, the transient response becomes longer. Therefore this configuration does not allow for perfect current sharing with a short settling time. Future research could be aimed at finding an alternative approach for estimating the neighbouring generated currents. Possibly an iterative method that brings less uncertainty in the system, whereby the system performs more similar to the behaviour as has been displayed in the previous scenarios.

Chapter 8

Conclusions

DC microgrids are attracting growing interest and receive much research attention. For the reason that DC microgrids are highly efficient, reliable and in compliance with RES. A centralized model predictive control (MPC) based consensus algorithm proposed in [38] is to achieve proportional current sharing and voltage regulation in the microgrid. Due to the form of the communication network, as the number of DGUs increases, centralized control becomes infeasible, nonscalable, too costly, or too fragile. Therefore in *Chapter 2* the following goal of this research was formulated: Development of a distributed MPC algorithm that achieves proportional current sharing and voltage regulation in the DC microgrid and examining the effects of different parameters upon this system.

This research introduced the DC network and its control objectives in *Chapter 3*. Whereafter the basic principles behind MPC were explained in *Chapter 4*. In *Chapter 5* a classical centralized MPC approach is described to solve the optimal control problem of the DC microgrid. Wherein the centralized MPC controller optimizes the controllable buck converter output over a future horizon such that the control objectives are met, whilst considering grid constraints.

In the distributed MPC scheme, the total system consists of several subsystems that have their own local controllers. These local controllers, present at each agent, make decision only depending on local information. When variables of different subproblems are connected by constraints, dual variables can be interpreted as prices serving to achieve mutual agreement between the subproblems. The decision variables and prices are communicated between neighboring agents and determined iteratively by means of subgradient iterations. The theory from [43] which can be used for decomposition and distributed optimization of control optimization is applied to the centralized MPC scheme for DC microgrids in *Chapter 6*.

The network was implemented and tested in *Chapter 7*. The solutions of distributed algorithm were compared to the centralized formulation's solutions. Simulations of different scenarios had to give clarity to the topic. In the scenarios 1 to 7 the effects of different parameters, variables and constraints were tested on the distributed MPC algorithm for DC microgrids. The dual decomposition is performed on the physical system, as the DGUs share current with each other. Dual decomposition is usually performed on a simpler matrix [36],[30], which makes the convergence of the subgradient iterations easier. In this research dual decomposition is performed on a more complex coupling matrix. As a result, it was discovered throughout the scenarios

that the convergence properties of the subgradient algorithms utilized were greatly affected by varying parameters, variables and constraints of the system in the optimization. In scenario 3 the priority for optimization was allocated to voltage balancing instead of proportional current sharing. Resulting in an ill-conditioned problem were the convergence rate declined as a consequence of the increased condition number. However in scenario 4 it was observed that for ill-conditioned distributed formulations, its solution would get close to the optimal solution relatively fast. Therefore the distributed algorithm for ill-conditioned problems could be prematurely terminated and still achieve a form of sub-optimality. Future research could go into more detail and establish a termination guideline that indicates how many iterations are needed to ensure a certain suboptimality guarantee, i.e., distance to optimality.

Furthermore the sampling time, T, was also observed to influence the subgradient convergence properties for the DMPC in the DC microgrid. For sampling times in the order of $T = 0.00005$, the subgradient algorithm will take a very long time to converge. On the other hand, sampling times of the order $T = 0.0005$ and larger, converge relatively fast to their optimal solutions.

In scenario 7, a method for generated current estimation was presented, where each DGU_i uses the actual system output for its neighbours' generated currents as an estimate over the prediction horizon. Hereby, making the controller fully distributed. Even though with this configuration similar convergence properties have been experienced, the system behaviour was different to that of the other scenarios.

Altogether, MPC via dual decomposition and gradient iterations is a suitable design approach for reaching consensus in the DC microgrid in a distributed manner. This research has answered the research question and on its outcomes future research can be based. Suggestions for further research can be found in *Chapter 9*.

Chapter 9

Further Research

In this chapter the limitations of this research will be discussed. These will leave room for future research.

The main direction for further research would be to propose an alternative method to estimated the neighbouring generated currents at each node. In scenario 7 the controller used the actual system output as an estimated for the neighbouring generated currents, this non-iterative estimation approach brought more uncertainty in the system and could not accomplish proportional current sharing combined with a fast transient responses after a load demand variation. In future research the possibility of an iterative method that brings less uncertainty in the system can be researched. Whereby the system behaviour becomes more similar to that of the scenarios where knowledge of each others system was known.

Secondly the load demands in the system are said to be known. If these load would not be known, an observer could be used to estimate these loads. Future research could be aimed at the incorporation of an observer in the distributed MPC for DC microgrids. Furthermore at sampling time $T = 0.00005$ the subgradient algorithm took a lot of time to convergence to the optimal solution. Future research could be aimed at employing accelerated gradient methods to speed up the convergence.

Also in scenario 4 it was seen that the for ill-conditioned optimization problems, due to prioritizing voltage balancing, the distributed solutions converged close to the optimal solution. Although it took time to actually reach this solution. Future research could go into more detail and establish a termination guideline that would indicate how many iterations are needed to ensure a certain suboptimality guarantee, i.e., distance to optimality.

One of the most fundamental problems in MPC is the lack of guaranteed stability and feasibility [32], contrasting to other control approaches. Stability and feasibility of the MPC controller in this research is generally attained due to definition of the optimization problem. However the computations of optimal inputs may work for some time, but then, all of a sudden, the MPC controller has driven the state to a region where the optimization problem has no solution [32]. Therefore future work can be aimed at developing a certificate for this MPC controller in the DC power network that proves the problem is recursively feasible.

Appendix A

Parameter Specification

A.1 Network

DGU		**1**	**2**	**3**	**4**
R_{s_i}	(Ω)	0.2	0.3	0.5	0.1
L_{s_i}	(mH)	1.8	2.0	3.0	2.2
C_{s_i}	(mF)	2.2	2.2	2.2	2.2
w_i	(-)	0.40^{-1}	0.20^{-1}	0.15^{-1}	0.25^{-1}
$V_i(0)$	(V)	380.66	379.37	378.15	378.64
$I_{s_i}(0)$	(A)	45.20	22.60	16.95	28.25
$I_{L_i}(0.01)$	(A)	40	22	20	31
$\Delta I_{L_i}(0.01^+)$	(A)	-10	-7	10	-5
$V_i^\star$	(V)	380	380	380	380

Table A.1: Microgrid parameters and current demand.

Line		**1,2**	**2,3**	**3,4**	**4,1**
R_{ij}	(Ω)	0.7	0.5	0.8	0.6

Table A.2: Line parameters.

Bounds		
$\overline{V}$	(V)	390
$\underline{V}$	(V)	370
$\overline{u}$	(V)	800
$\underline{u}$	(V)	0

Table A.3: Voltage and input constraints.

A.2 MPC

$$
\begin{aligned}
t_{max} &= 100 \\
N &= 4 \\
T &= 0.0005s \\
\mathcal{Q} &= 10^{1} I_{n \times n} \\
\mathcal{R}_1 &= 10^{-2} I_{n \times n} \\
\mathcal{R}_2 &= 10^{1} I_{n \times n}
\end{aligned}
$$

A.3 Gradient update

$$
\begin{aligned}
\gamma &= 10 \\
\epsilon &= 0.001 \\
\beta &= 0.5 \\
S_k &= 1000
\end{aligned}
$$

Appendix B

Code

B.1 Network

Network parameters and other model specifics

```
%Filter Resistors
R1 = 0.2;
R2 = 0.3;
R3 = 0.5;
R4 = 0.1;
RR = diag([R1; R2; R3; R4]);

% Filter Inductance
L1 = 1.8e-03;
L2 = 2.0e-03;
L3 = 3.0e-03;
L4 = 2.2e-03;
L =  diag([L1; L2; L3; L4]);

%Shunt Capacitor
C1 = 2.2e-03;
C2 = 1.9e-03;
C3 = 2.5e-03;
C4 = 1.7e-03;
CC = diag([C1; C1; C1; C1]);

%Resistive Lines
RL = diag([0.70;0.50;0.80;0.60]);

%Coupling Matrix
Inc = [-1 0 0 -1; 1 -1 0 0; 0 1 -1 0; 0 0 1 1];      %Incidence Matrix
Lapp = Inc * inv(RL) * Inc';
LL = T *inv(CC)* Lapp;
```

```
A = eye(4) - LL;                                        %Coupling matrix
Ao = A - diag(diag(A));                                 %Off-diagonal coupling matrix
Ad = A - Ao;                                            %Diagonal coupling matrix

% Weighting Matrix
one = [1;1;1;1];                                                    %Vector of ones
W = diag([ inv(0.40) ; inv(0.2) ; inv(0.15) ; inv(0.25) ]);         %Weighted matrix
WW = inv(W)*one*(one');

%Objective Function
V_star = one *380;          %Desired voltage load
v_star = Ao * V_star;       %Desired auxilary variable
Q = 1e1 * eye(4);           %Weighted identity matrix for proportional current sharing
R_1 = 1e-2 * eye(4);        %Weighted identity matrix for penalizing controllable input
R_2 = 1e1 * eye(4);         %Weighted identity matrix for voltage balancing
Lap = Inc * Q * Inc';       %Laplacian matrix

%Gradient iterations
gamma = 10;                 %Step size
epsilon = 0.001;            %Error
beta = 0.5;                 %Momentum term
```

Starting values and bounds

```
%Starting values load voltages
V1_0 = 380.2;
V2_0 = 379.74;
V3_0 = 379.71;
V4_0 = 379.9;
V_0 = [V1_0; V2_0; V3_0; V4_0];

%Starting values generated current
I1_0 = 45.2;
I2_0 = 22.6;
I3_0 = 16.95;
I4_0 = 28.25;
I_0 = [I1_0; I2_0; I3_0; I4_0];

%First values of I_L
I1_L0 = 40;
I2_L0 = 22;
I3_L0 = 20;
I4_L0 = 31;
I_L0 = [I1_L0; I2_L0; I3_L0; I4_L0];

%Second values of I_L
I1_L = 30;
```

```
I2_L = 15;
I3_L = 30;
I4_L = 26;
I_L11 = [I1_L; I2_L; I3_L; I4_L];

%Bounds
Vmax = 390;
Vmin = 370;
umin = 0;
umax = 800;
```

B.2 Centralized MPC algorithm

```
%Primal MPC
cvx_begin quiet

    %Variables
    variables I(n,N+1)              %Generated Current
    variables V(n,N+1)              %Load Voltage
    variables u(n,N)                %Buck Convert Output Voltage

    %Expressions
    expression obj(N)               %Objective function

    for k = 1:N
                obj(k) =   0.5*Q(1,1)*((W(1,1))*I(1,k) - (W(2,2)*I_0(2,1)))^2 + ...
                           0.5*Q(1,1)*((W(1,1))*I(1,k) - (W(4,4)*I_0(4,1)))^2 + ...
                           0.5*Q(1,1)*((W(2,2))*I(2,k) - (W(1,1)*I_0(1,1)))^2 + ...
                           0.5*Q(1,1)*((W(2,2))*I(2,k) - (W(3,3)*I_0(3,1)))^2 + ...
                           0.5*Q(1,1)*((W(3,3))*I(3,k) - (W(2,2)*I_0(2,1)))^2 + ...
                           0.5*Q(1,1)*((W(3,3))*I(3,k) - (W(4,4)*I_0(4,1)))^2 + ...
                           0.5*Q(1,1)*((W(4,4))*I(4,k) - (W(1,1)*I_0(1,1)))^2 + ...
                           0.5*Q(1,1)*((W(4,4))*I(4,k) - (W(3,3)*I_0(3,1)))^2 + ...
                           u(:,k)'*R_1*u(:,k) + (Ao*V(:,k)-v_star)'*R_2*(Ao*V(:,k)-v_star);

                %Initial Conditions
                I(:,1) == I_0;
                V(:,1) == V_0;

                %System Dynamics
                I(:,k+1) == I(:,k) + T * inv(L) * (-RR*I(:,k) - V(:,k) + u(:,k));
                V(:,k+1) == A*V(:,k) + T * inv(CC) * (I(:,k) - I_L1);

                % Voltage Bounds
                V <= Vmax;
                V >= Vmin;
```

```
                u <= umax;
                u >= umin;
                I >= 0;

    end

    minimize( sum(obj(1:N)) )

cvx_end
p = cvx_optval;  %Primal solution
ps(1,t) = p;     %Primal solutions in an array
```

B.3 Distributed MPC algorithm

```
%Distributed MPC
for i = 1:R %Iteration number

    cvx_begin quiet

        %Variables
        variables I(n,N+1)              %Generated Current
        variables V(n,N+1)              %Load Voltage
        variables u(n,N)                %Buck Convert Output Voltage
        variables v(n,N)                %Auxilary variable

        %Expressions
        expression obj(N)               %Objective function

        for k = 1:N
                    obj(k) =   0.5*Q(1,1)*((W(1,1))*I(1,k) - (W(2,2)*I_0(2,1)))^2 + ...
                               0.5*Q(1,1)*((W(1,1))*I(1,k) - (W(4,4)*I_0(4,1)))^2 + ...
                               0.5*Q(1,1)*((W(2,2))*I(2,k) - (W(1,1)*I_0(1,1)))^2 + ...
                               0.5*Q(1,1)*((W(2,2))*I(2,k) - (W(3,3)*I_0(3,1)))^2 + ...
                               0.5*Q(1,1)*((W(3,3))*I(3,k) - (W(2,2)*I_0(2,1)))^2 + ...
                               0.5*Q(1,1)*((W(3,3))*I(3,k) - (W(4,4)*I_0(4,1)))^2 + ...
                               0.5*Q(1,1)*((W(4,4))*I(4,k) - (W(1,1)*I_0(1,1)))^2 + ...
                               0.5*Q(1,1)*((W(4,4))*I(4,k) - (W(3,3)*I_0(3,1)))^2 + ...
                               u(:,k)'*R_1*u(:,k) + (v(:,k)-v_star)'*R_2*(v(:,k)-v_star) + ...
                               lambda(:,k)'*(v(:,k) - Ao*V(:,k));

                    %Initial conditions
                    I(:,1) == I_0;
                    V(:,1) == V_0;

                    %System dynamics
                    I(:,k+1) == I(:,k) + T * inv(L) * (-RR*I(:,k) - V(:,k) + u(:,k));
```

```
                V(:,k+1) == Ad*V(:,k) + v(:,k) + T * inv(CC) * (I(:,k) - I_L1);

                %Bounds
                V <= Vmax;
                V >= Vmin;
                u <= umax;
                u >= umin;
                I >=    0;

    end

    minimize( sum(obj(1:N)) )

cvx_end
d = cvx_optval; %Dual solution
ds(1,i) = d;    %Dual solutions in an array

%Plotting purposes
v1s(1:N,i) = v(1,1:N)';
v2s(1:N,i) = v(2,1:N)';
v3s(1:N,i) = v(3,1:N)';
v4s(1:N,i) = v(4,1:N)';

I1s(1:N,i) = I(1,1:N)';
I2s(1:N,i) = I(2,1:N)';
I3s(1:N,i) = I(3,1:N)';
I4s(1:N,i) = I(4,1:N)';

u1s(1:N,i) = u(1,1:N)';
u2s(1:N,i) = u(2,1:N)';
u3s(1:N,i) = u(3,1:N)';
u4s(1:N,i) = u(4,1:N)';

V1s(1:N,i) = V(1,1:N)';
V2s(1:N,i) = V(2,1:N)';
V3s(1:N,i) = V(3,1:N)';
V4s(1:N,i) = V(4,1:N)';

    %Lagrangian updates
    for k = 1:N
                subg(:,k) = v(:,k) - Ao*V(:,k);

                lambdas1(k,i) = lambda(1,k)';
                lambdas2(k,i) = lambda(2,k)';
                lambdas3(k,i) = lambda(3,k)';
                lambdas4(k,i) = lambda(4,k)';
                lambdas11(1,t) = lambdas1(1,end);
```

```
                    lambdas21(1,t) = lambdas2(1,end);
                    lambdas31(1,t) = lambdas3(1,end);
                    lambdas41(1,t) = lambdas4(1,end);

                    subgs1(k,i) = subg(1,k)';
                    subgs2(k,i) = subg(2,k)';
                    subgs3(k,i) = subg(3,k)';
                    subgs4(k,i) = subg(4,k)';

                    %Momentum
                    e(:,k) = (beta)*(subg(:,k)) + (1-beta) * e(:,k);
                    lambda(:,k) = lambda(:,k) + gamma *e(:,k);

        end

    %Stopping criteria
    iterations(t) = i;
        if      ( abs(subg(1,1)) < epsilon && abs(subg(1,2)) < epsilon ) && ...
                ( abs(subg(2,1)) < epsilon && abs(subg(2,2)) < epsilon ) && ...
                ( abs(subg(3,1)) < epsilon && abs(subg(3,2)) < epsilon ) && ...
                ( abs(subg(4,1)) < epsilon && abs(subg(4,2)) < epsilon )
                  break
        end


end
```

B.4 Run

Time specifations

```
N = 8;          %Prediction horizon
T = 0.0005;     %Sampling time T = 0.00005;
n = 4;          %Nodes
R = 100;        %Iteration number
tmax = 100;       %Time horizon
```

Centralized MPC

```
run('Network')

for t = 1:tmax

    %Load Demand
    if t < 20
```

```
        I_L1 = (I_L0);
    else
        I_L1 = I_L11;
    end

    %MPC
    run('CentralizedMPC')

    %System
    run('RealSystem')

    %Plotting purposes
    I1c(1:N,t) = I(1,1:N);
    I2c(1:N,t) = I(2,1:N);
    I3c(1:N,t) = I(3,1:N);
    I4c(1:N,t) = I(4,1:N);

    V1c(1:N,t) = V(1,1:N);
    V2c(1:N,t) = V(2,1:N);
    V3c(1:N,t) = V(3,1:N);
    V4c(1:N,t) = V(4,1:N);

    u1c(1:N,t) = u(1,1:N);
    u2c(1:N,t) = u(2,1:N);
    u3c(1:N,t) = u(3,1:N);
    u4c(1:N,t) = u(4,1:N);
end
```

Distributed MPC

```
run('Network')

for t = 1:tmax

    %Initialization at zero
    lambda = zeros(n,N);
    e = zeros(n,N);

    %Load Demand
    if t < 20
        I_L1 = (I_L0);
    else
        I_L1 = I_L11;
    end

    %MPC
    run('DistributedMPC')
```

```
    %System
    run('RealSystem')

    %Plotting purposes
    I1ss(1:N,t) = I(1,1:N)';
    I2ss(1:N,t) = I(2,1:N)';
    I3ss(1:N,t) = I(3,1:N)';
    I4ss(1:N,t) = I(4,1:N)';

    V1ss(1:N,t) = V(1,1:N)';
    V2ss(1:N,t) = V(2,1:N)';
    V3ss(1:N,t) = V(3,1:N)';
    V4ss(1:N,t) = V(4,1:N)';

    u1ss(1:N,t) = u(1,1:N)';
    u2ss(1:N,t) = u(2,1:N)';
    u3ss(1:N,t) = u(3,1:N)';
    u4ss(1:N,t) = u(4,1:N)';

end
```

Plots

```
%Generated Current
figure(1)
set(gcf,'color','w');
stairs(1:tmax,I1c(1,1:tmax),'b','DisplayName','I1c')
hold on
stairs(1:tmax,I1ss(1,1:tmax),'--b','DisplayName','I1d')
stairs(1:tmax,I2c(1,1:tmax),'r','DisplayName','I2c')
stairs(1:tmax,I2ss(1,1:tmax),'r--','DisplayName','I2d')
stairs(1:tmax,I3c(1,1:tmax),'y','DisplayName','I3c')
stairs(1:tmax,I3ss(1,1:tmax),'y--','DisplayName','I3d')
stairs(1:tmax,I4c(1,1:tmax),'g','DisplayName','I4c')
stairs(1:tmax,I4ss(1,1:tmax),'g--','DisplayName','I4d')
xlabel('time (s)')
ylabel('\it I_s  (A)','Interpreter','tex')
legend({'$$I^c_{s_1}$$','$$I^d_{s_1}$$','$$I^c_{s_2}$$','$$I^d_{s_2}$$','$$I^c_{s_3}$$','$$I^d_{s_3}$
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
set(legend,'Location','northeast')
legend
ylim([10 50])
set(gca,'fontsize',26)
set(gca,'XTick',0:20:(tmax*1));
set(gca,'XTickLabel',0:0.01:(tmax*T));
```

```
grid on
hold off

%Load Voltage
figure(2)
set(gcf,'color','w');
stairs(1:tmax,V1c(1,1:tmax),'b','DisplayName','V1c')
hold on
stairs(1:tmax,V1ss(1,1:tmax),'--b','DisplayName','V1d')
stairs(1:tmax,V2c(1,1:tmax),'r','DisplayName','V2c')
stairs(1:tmax,V2ss(1,1:tmax),'r--','DisplayName','V2d')
stairs(1:tmax,V3c(1,1:tmax),'y','DisplayName','V3c')
stairs(1:tmax,V3ss(1,1:tmax),'y--','DisplayName','V3d')
stairs(1:tmax,V4c(1,1:tmax),'g','DisplayName','V4c')
stairs(1:tmax,V4ss(1,1:tmax),'g--','DisplayName','V4d')
stairs(1:tmax, one'*inv(W)*[V1c(1,1:tmax); V2c(1,1:tmax); V3c(1,1:tmax); V4c(1,1:tmax)],'-.k'
xlabel('time (s)')
ylabel('\it V (V)','Interpreter','tex')
legend({'$$V^c_{1}$$','$$V^d_{1}$$','$$V^c_{2}$$','$$V^d_{2}$$','$$V^c_{3}$$','$$V^d_{3}$$','$
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
set(legend,'Location','northeast')
legend
ylim([370 385])
set(gca,'fontsize',26)
set(gca,'XTick',0:20:(tmax*1));
set(gca,'XTickLabel',0:0.01:(tmax*T));
grid on
hold off

%Controllable Input
figure(3)
set(gcf,'color','w');
stairs(1:tmax,u1c(1,1:tmax),'b','DisplayName','u1c')
hold on
stairs(1:tmax,u1ss(1,1:tmax),'--b','DisplayName','u1d')
stairs(1:tmax,u2c(1,1:tmax),'r','DisplayName','u2c')
stairs(1:tmax,u2ss(1,1:tmax),'r--','DisplayName','u2d')
stairs(1:tmax,u3c(1,1:tmax),'y','DisplayName','u3c')
stairs(1:tmax,u3ss(1,1:tmax),'y--','DisplayName','u3d')
stairs(1:tmax,u4c(1,1:tmax),'g','DisplayName','u4c')
stairs(1:tmax,u4ss(1,1:tmax),'g--','DisplayName','u4d')
xlabel('time (s)')
ylabel('\it u (V)','Interpreter','tex')
legend({'$$u^c_{1}$$','$$u^d_{1}$$','$$u^c_{2}$$','$$u^d_{2}$$','$$u^c_{3}$$','$$u^d_{3}$$','$
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
```

```
set(legend,'Location','northeast')
legend
ylim([350 420])
set(gca,'fontsize',26)
set(gca,'XTick',0:20:(tmax*1));
set(gca,'XTickLabel',0:0.01:(tmax*T));
grid on
hold off

%No. of iterations
figure(4)
grid on
set(gcf,'color','w');
plot(1:tmax, iterations(1:tmax),'k','DisplayName','iterations')
xlabel('time instance')
ylabel('no. of iterations')
legend({'$$Iterations$$'})
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
set(legend,'Location','northeast')
legend
ylim([15 25])
set(gca,'fontsize',26)
set(gca,'XTick',0:10:(tmax*1));
set(gca,'XTickLabel',0:10:(tmax*1));
grid on

%Dual vs. Primal
figure(5)
set(gcf,'color','w');
plot(1:i-1, ds(1:i-1)-ps(t),'DisplayName', 'dual-primal')
xlabel('iterations')
ylabel('Distributed-Centralized solution')
set(gca,'fontsize',26)
set(gca,'XTick',0:5:(i*1));
set(gca,'XTickLabel',0:5:(i*1));
grid on

%Current Sharing
ppp=Inc*Inc'*W*[I1ss(1,1:tmax); I2ss(1,1:tmax); I3ss(1,1:tmax); I4ss(1,1:tmax)]; %
figure(6)
set(gcf,'color','w');
plot(1:tmax, ppp(1,1:tmax),'DisplayName','CS1')
hold on
plot(1:tmax, ppp(2,1:tmax),'DisplayName','CS2')
plot(1:tmax, ppp(3,1:tmax),'DisplayName','CS3')
plot(1:tmax, ppp(4,1:tmax),'DisplayName','CS4')
```

```
xlabel('time (s)')
ylabel('\it I_s  (A)','Interpreter','tex')
legend({'$$DGU_1$$','$$DGU_2$$','$$DGU_3$$','$$DGU_4$$'})
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
set(legend,'Location','northeast')
legend
%ylim([])
set(gca,'fontsize',26)
set(gca,'XTick',0:20:(tmax*1));
set(gca,'XTickLabel',0:0.01:(tmax*T));
grid on
hold off

%Lagrangian multipliers, Node 1
figure(7)
set(gcf,'color','w');
plot(1:i-1,lambdas1(1,1:i-1),'DisplayName', 'lambdas1')
hold on
plot(1:i-1,lambdas1(2,1:i-1),'DisplayName', 'lambdas1')
plot(1:i-1,lambdas1(3,1:i-1),'DisplayName', 'lambdas1')
xlabel('iterations')
ylabel('Lagrange multipliers')
legend({'$$\lambda_1(k=0) $$','$$\lambda_1(k=1) $$','$$\lambda_1(k=2) $$'})
set(legend,'Interpreter','latex')
set(legend,'Orientation','horizontal')
set(legend,'Location','northwest')
legend
%ylim([-20 20])
set(gca,'fontsize',26)
set(gca,'XTick',0:5:(i*1));
set(gca,'XTickLabel',0:5:(i*1));
grid on
hold off
```

Appendix C

Complementary Data

C.1 Scenario 1

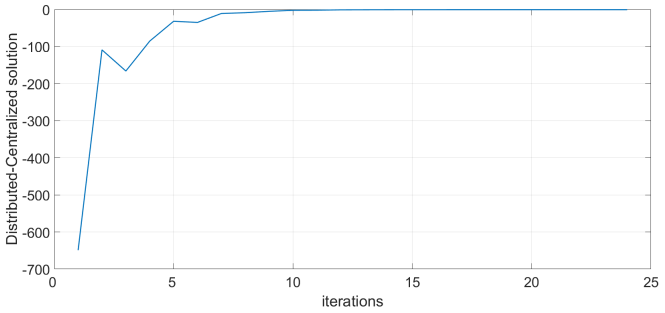

FIGURE C.1: Evolution of the distributed objective function solution vs. the centralized solution at time step 100.

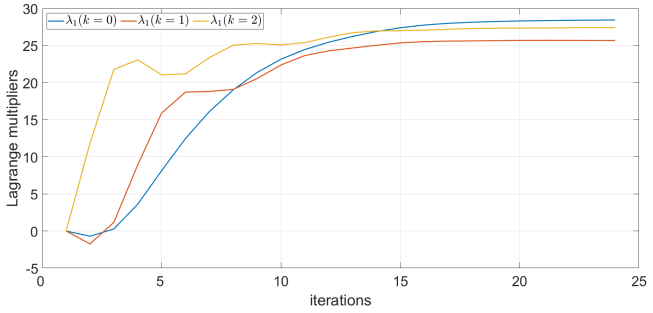

FIGURE C.2: Evolution of DGU_1's Lagrangian multipliers at time step 100.

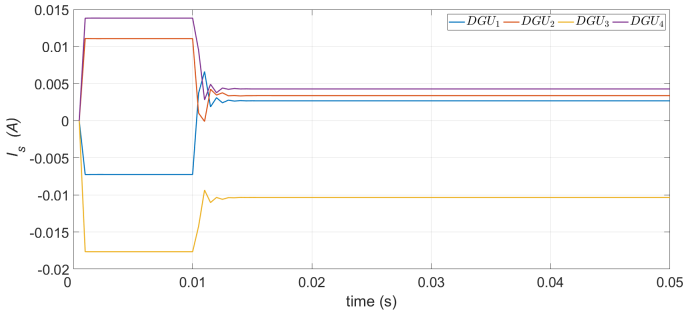

FIGURE C.3: Current sharing at each DGU ($\mathcal{LWI}(t)$).

C.2 Scenario 2

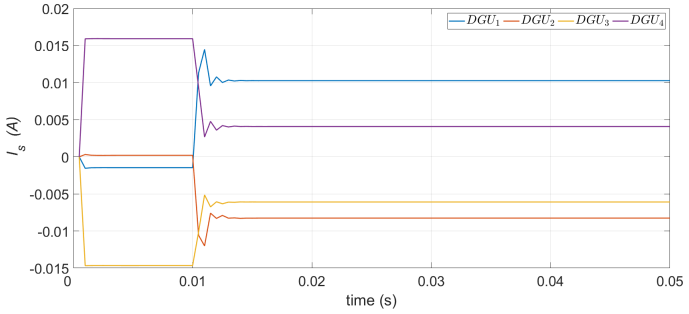

FIGURE C.4: Current sharing at each DGU ($\mathcal{LWI}(t)$).

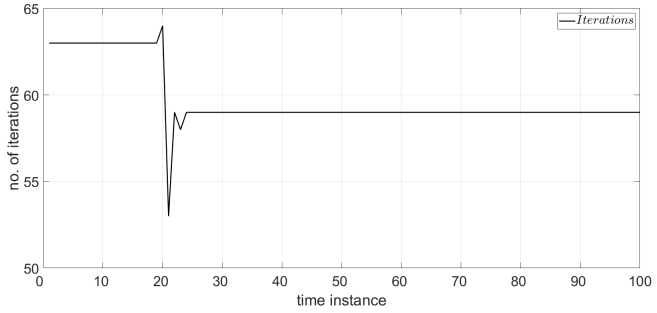

FIGURE C.5: Number of iterations until termination - Scenario 2

C.3 Scenario 3

Scenario 3a

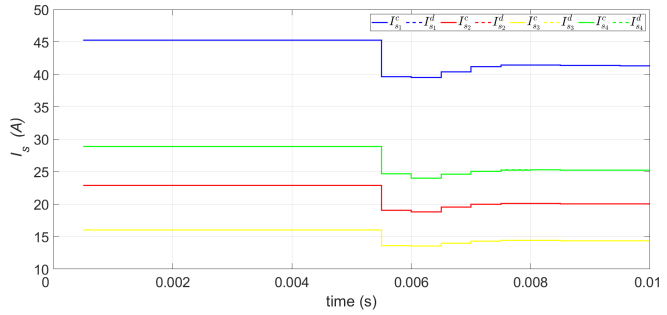

FIGURE C.6: State trajectories for the generated current I_s - Scenario 3a

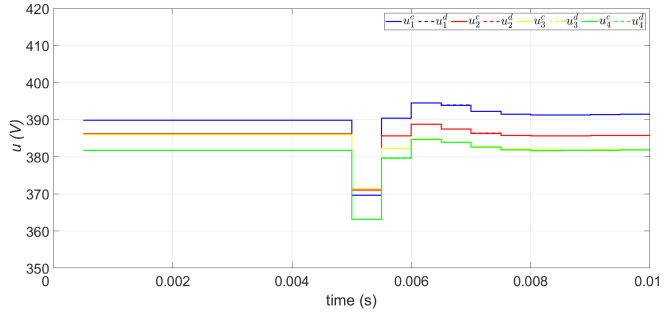

FIGURE C.7: State trajectories for the control inputs u - Scenario 3a

Scenario 3b

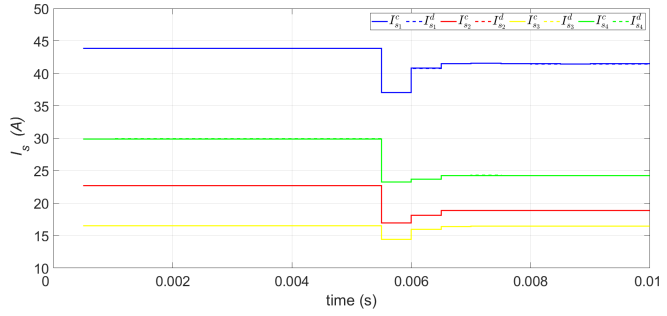

FIGURE C.8: State trajectories for the generated current I_s - Scenario 3b

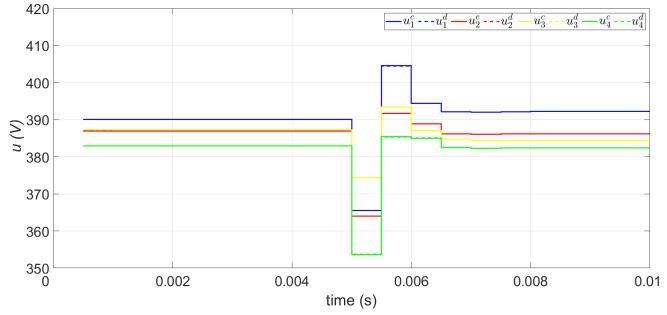

FIGURE C.9: State trajectories for the control inputs u - Scenario 3b

Scenario 3c

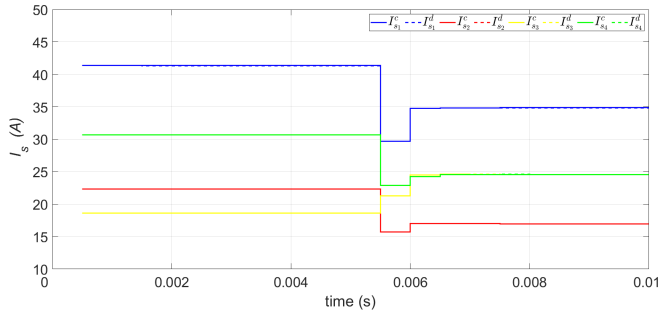

FIGURE C.10: State trajectories for the generated current I_s - Scenario 3c

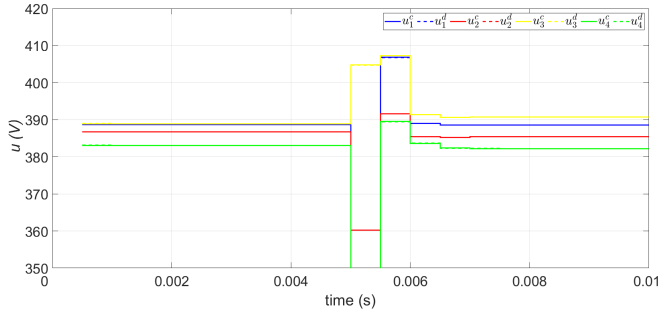

FIGURE C.11: State trajectories for the control inputs u - Scenario 3c

Scenario 3d

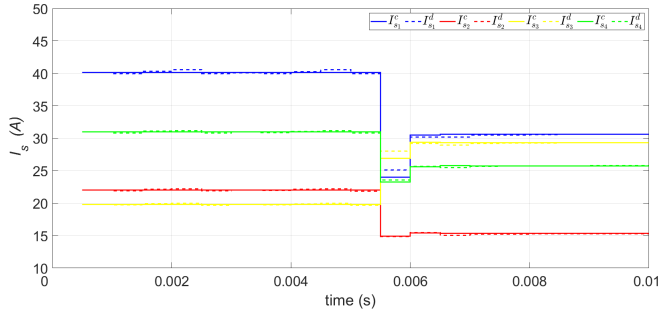

FIGURE C.12: State trajectories for the generated current I_s - Scenario 3d

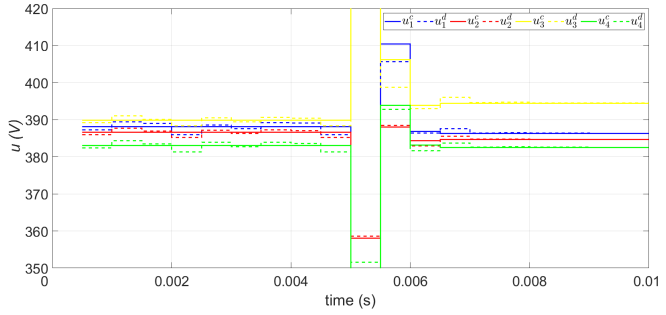

FIGURE C.13: State trajectories for the control inputs u - Scenario 3d

C.4 Scenario 4

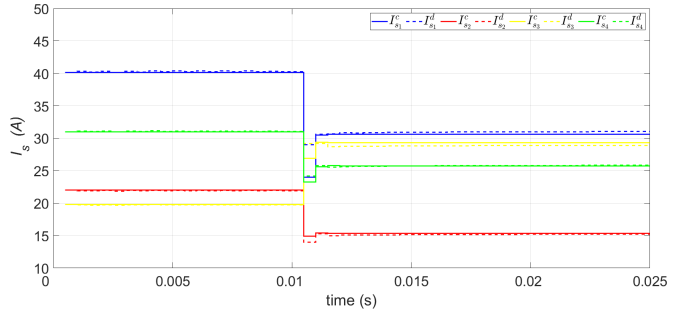

FIGURE C.14: State trajectories for the generated current I_s - Scenario 4

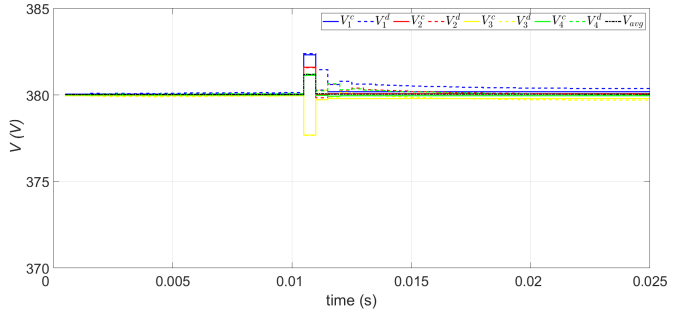

FIGURE C.15: State trajectories for the load voltage V - Scenario 4

C.5 Scenario 5

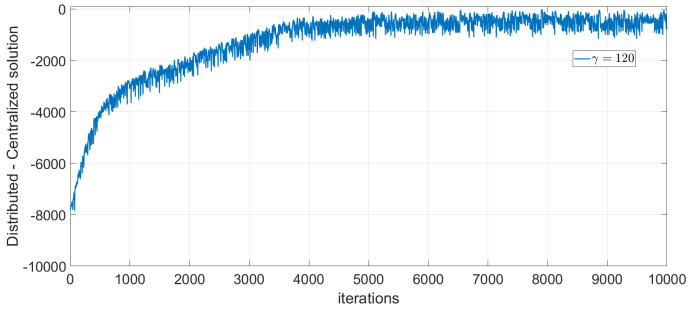

FIGURE C.16: Evolution of the dual solution vs. the primal solution for one time step with $\gamma = 120$

C.6 Scenario 6

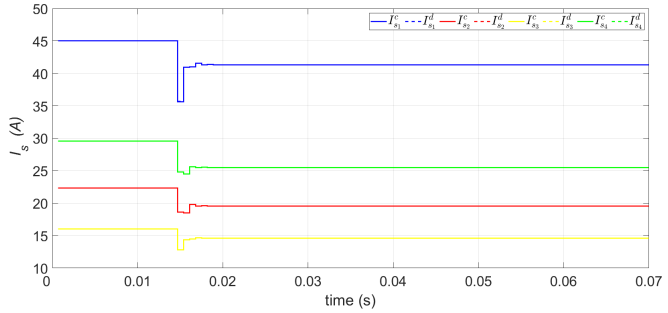

FIGURE C.17: State trajectories for the generated current I_s - Scenario 6

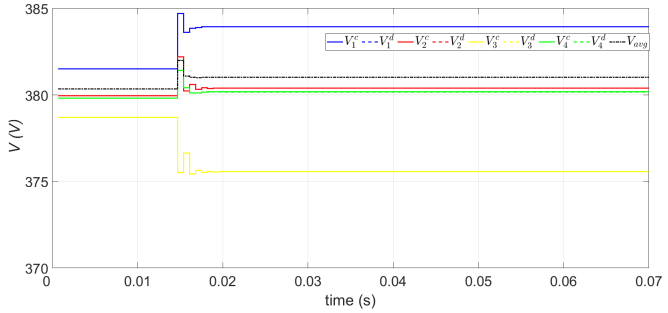

FIGURE C.18: State trajectories for the load voltage V - Scenario 6

C.7 Scenario 7

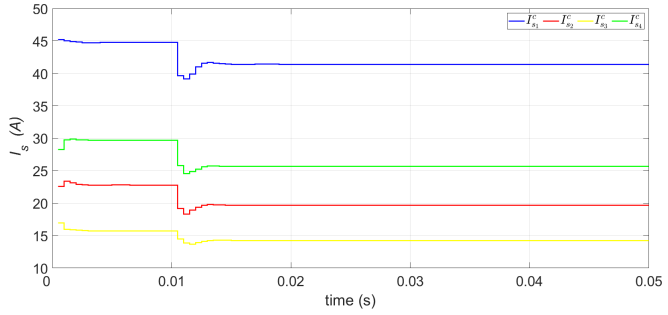

FIGURE C.19: State trajectories for the generated current I_s, $\frac{R_2}{Q} = 1,000$ - Scenario 7

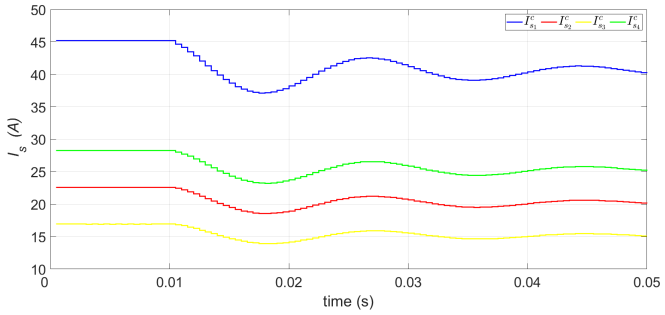

FIGURE C.20: State trajectories for the generated current I_s, $\frac{R_2}{Q} = 10$ - Scenario 7

Bibliography

[1] Dimitri Bertsekas. Partial conjugate gradient methods for a class of optimal control problems. *IEEE Transactions on Automatic Control*, 19(3):209–217, 1974.

[2] Dimitri P Bertsekas. Combined primal-dual and penalty methods for constrained minimization. *SIAM Journal on Control*, 13(3):521–544, 1975.

[3] DP Bertsekas. Convexification procedures and decomposition methods for nonconvex optimization problems. *Journal of Optimization Theory and Applications*, 29(2):169–197, 1979.

[4] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[5] Stephen Boyd, Lin Xiao, and Almir Mutapcic. Subgradient methods. *lecture notes of EE392o, Stanford University, Autumn Quarter*, 2004:2004–2005, 2003.

[6] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[7] Eduardo Camponogara, Dong Jia, Bruce H Krogh, and Sarosh Talukdar. Distributed model predictive control. *IEEE Control Systems*, 22(1):44–52, 2002.

[8] Sudipta Chakraborty, Manoja D Weiss, and Marcelo Godoy Simoes. Distributed intelligent energy management system for a single-phase high-frequency ac microgrid. *IEEE Trans. Industrial Electronics*, 54(1):97–109, 2007.

[9] Bruce Desmond Craven. *Mathematical programming and control theory*. Springer Science & Business Media, 2012.

[10] Michele Cucuzzella, Sebastian Trip, Claudio De Persis, Xiaodong Cheng, Antonella Ferrara, and Arjan van der Schaft. A robust consensus algorithm for current sharing and voltage regulation in dc microgrids. *IEEE Transactions on Control Systems Technology*, 2018.

[11] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

[12] Enrique Rodriguez Diaz, Xiaoling Su, Mehdi Savaghebi, Juan C Vasquez, Minxiao Han, and Josep M Guerrero. Intelligent dc microgrid living laboratories-a chinese-danish cooperation project. In *DC Microgrids (ICDCM), 2015 IEEE First International Conference on*, pages 365–370. IEEE, 2015.

[13] Chuong B Do. Convex optimization overview (cnt'd). 2009.

[14] Minh Dang Doan, Tamás Keviczky, and Bart De Schutter. An iterative scheme for distributed model predictive control using fenchel's duality. *Journal of Process Control*, 21(5): 746–755, 2011.

[15] Alessandro Falsone, Kostas Margellos, Simone Garatti, and Maria Prandini. Dual decomposition for multi-agent distributed optimization with coupling constraints. *Automatica*, 84: 149–158, 2017.

[16] Marcello Farina, Giulio Betti, Luca Giulioni, and Riccardo Scattolini. A review on distributed predictive control: basic ideas, extensions, and applications.

[17] Farhad Farokhi, Iman Shames, and Karl H Johansson. Distributed mpc via dual decomposition and alternative direction method of multipliers. In *Distributed model predictive control made easy*, pages 115–131. Springer, 2014.

[18] P Giselsson, MD Doan, T Keviczky, B De Schutter, and A Rantzer. A distributed optimization algorithm with convergence rate o (1 k2) for distributed model predictive control. *Automatica. Submitted*, 2011.

[19] Pontus Giselsson and Anders Rantzer. On feasibility, stability and performance in distributed model predictive control. *IEEE Transactions on Automatic Control*, 59(4):1031–1036, 2014.

[20] Michael Grant, Stephen Boyd, and Yinyu Ye. Cvx: Matlab software for disciplined convex programming, 2008.

[21] Maheswaran Gunasekaran, Hidayathullah Mohamed Ismail, Bharatiraja Chokkalingam, Lucian Mihet-Popa, and Sanjeevikumar Padmanaban. Energy management strategy for rural communities' dc micro grid power system structure with maximum penetration of renewable energy sources. *Applied Sciences*, 8(4):585, 2018.

[22] Jan Fredrik Hansen, John O Lindtjørn, Klaus Vanska, and O Abb. Onboard dc grid for enhanced dp operation in ships. In *Dynamic Positioning Conference, Houston*, 2011.

[23] Gian Paolo Incremona, Michele Cucuzzella, Antonella Ferrara, and Lalo Magni. Model predictive control and sliding mode control for current sharing in microgrids. In *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*, pages 2661–2666. IEEE, 2017.

[24] Youichi Ito, Yang Zhongqing, and Hirofumi Akagi. Dc microgrid based distribution power generation system. In *Power Electronics and Motion Control Conference, 2004. IPEMC 2004. The 4th International*, volume 3, pages 1740–1745. IEEE, 2004.

[25] H Kakigano, M Nomura, and T Ise. Loss evaluation of dc distribution for residential houses compared with ac system. In *Power Electronics Conference (IPEC), 2010 International*, pages 480–486. IEEE, 2010.

[26] Hiroaki Kakigano, Yushi Miura, and Toshifumi Ise. Low-voltage bipolar-type dc microgrid for super high quality distribution. *IEEE transactions on power electronics*, 25(12):3066–3075, 2010.

[27] Faridaddin Katiraei, Mohammad Reza Iravani, and Peter W Lehn. Micro-grid autonomous operation during and subsequent to islanding process. *IEEE Transactions on power delivery*, 20(1):248–257, 2005.

[28] A Senthil Kumar and Zainal Ahmad. Model predictive control (mpc) and its current issues in chemical engineering. *Chemical Engineering Communications*, 199(4):472–511, 2012.

[29] Prabha Kundur, Neal J Balu, and Mark G Lauby. *Power system stability and control*, volume 7. McGraw-hill New York, 1994.

[30] Gunn Kristine Holst Larsen. *Distributed Control of a Network with Multiple Electricity Producers and Consumers*. University of Groningen Library][Host], 2014.

[31] Chun-Hao Lo and Nirwan Ansari. Decentralized controls and communications for autonomous distribution networks in smart grid. *IEEE transactions on smart grid*, 4(1): 66–77, 2013.

[32] Johan Löfberg. Oops! i cannot do it again: Testing for recursive feasibility in mpc. *Automatica*, 48(3):550–555, 2012.

[33] Thomas Morstyn, Branislav Hredzak, and Vassilios G Agelidis. Dynamic optimal power flow for dc microgrids with distributed battery energy storage systems. In *Energy Conversion Congress and Exposition (ECCE), 2016 IEEE*, pages 1–6. IEEE, 2016.

[34] Angelia Nedić and Dimitri Bertsekas. Convergence rate of incremental subgradient algorithms. In *Stochastic optimization: algorithms and applications*, pages 223–264. Springer, 2001.

[35] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.

[36] D Bao Nguyen, Jacquelien MA Scherpen, and Frits Bliek. Distributed optimal control of smart electricity grids with congestion management. *IEEE Transactions on Automation Science and Engineering*, 14(2):494–504, 2017.

[37] Navid Noroozi and Alireza Khayatian. Time-scale separation redesign for stabilisation and performance recovery of sampled-data non-linear systems. *IET Control Theory & Applications*, 9(5):801–810, 2015.

[38] Navid Noroozi, Sebastian Trip, and Roman Geiselhart. Model predictive control of dc microgrids: current sharing and voltage regulation. *IFAC-PapersOnLine*, 51(23):124–129, 2018.

[39] Daniel Pérez Palomar and Mung Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, 2006.

[40] Anna Pyzara, Beata Bylina, and Jarosław Bylina. The influence of a matrix condition number on iterative methods' convergence. In *2011 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 459–464. IEEE, 2011.

[41] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[42] Anders Rantzer. On prize mechanisms in linear quadratic team theory. In *Decision and Control, 2007 46th IEEE Conference on*, pages 1112–1116. IEEE, 2007.

[43] Anders Rantzer. Dynamic dual decomposition for distributed control. In *American Control Conference, 2009. ACC'09.*, pages 884–888. IEEE, 2009.

[44] Jacques Richalet and Donal O'Donovan. *Predictive functional control: principles and industrial applications*. Springer Science & Business Media, 2009.

[45] Daniel Salomonsson, Lennart Soder, and Ambra Sannino. An adaptive control system for a dc microgrid for data centers. In *Industry Applications Conference, 2007. 42nd IAS Annual Meeting. Conference Record of the 2007 IEEE*, pages 2414–2421. IEEE, 2007.

[46] Riccardo Scattolini. Architectures for distributed and hierarchical model predictive control–a review. *Journal of process control*, 19(5):723–731, 2009.

[47] Dale E Seborg, Duncan A Mellichamp, Thomas F Edgar, and Francis J Doyle III. *Process dynamics and control*. John Wiley & Sons, 2010.

[48] Andreas Themelis, Silvia Villa, Panagiotis Patrinos, and Alberto Bemporad. Stochastic gradient methods for stochastic model predictive control. In *Control Conference (ECC), 2016 European*, pages 154–159. IEEE, 2016.

[49] Dan T Ton and Merrill A Smith. The us department of energy's microgrid initiative. *The Electricity Journal*, 25(8):84–94, 2012.

[50] Sebastian Trip, Michele Cucuzzella, Xiaodong Cheng, and Jacquelien Scherpen. Distributed averaging control for voltage regulation and current sharing in dc microgrids. *IEEE Control Systems Letters*, 2018.

[51] Energy Information Administration (US) and Government Publications Office. *International Energy Outlook 2016: With Projections to 2040*. Government Printing Office, 2016.

[52] Ton JJ van den Boom and TCPM Backx. Model predictive control. *DISC Course, Lecture Notes*, 16, 2010.

[53] Sergio Vazquez, Jose I Leon, Leopoldo G Franquelo, Jose Rodriguez, Hector A Young, Abraham Marquez, and Pericle Zanchetta. Model predictive control: A review of its applications in power electronics. *IEEE Industrial Electronics Magazine*, 8(1):16–31, 2014.

[54] Aswin N Venkat, Ian A Hiskens, James B Rawlings, and Stephen J Wright. Distributed mpc strategies with application to power system automatic generation control. *IEEE transactions on control systems technology*, 16(6):1192–1206, 2008.

[55] Erik Roelf Arjen Weitenberg. *Control of electrical networks: robustness and power sharing*. Rijksuniversiteit Groningen, 2018.

[56] Karl Worthmann. Estimates on the prediction horizon length in model predictive control. In *Proceedings of the 20th International Symposium on Mathematical Theory of Networks and Systems, CD–ROM, MTNS2012*, volume 112, 2012.

[57] Qiang Yang, Le Jiang, Hailin Zhao, and Hongmei Zeng. Autonomous voltage regulation and current sharing in islanded multi-inverter dc microgrid. *IEEE Transactions on Smart Grid*, 2017.

[58] Uğur YüUzgeç, Ahmet Palazoglu, and Jose A Romagnoli. Short-term planning model for petroleum refinery production: Using model predictive control. *IFAC Proceedings Volumes*, 43(5):308–313, 2010.

[59] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[60] Jinxin Zhao and Florian Dörfler. Distributed control and optimization in dc microgrids. *Automatica*, 61:18–26, 2015.

[61] Luis Eduardo Zubieta. Are microgrids the future of energy?: Dc microgrids from concept to demonstration to deployment. *IEEE Electrification Magazine*, 4(2):37–44, 2016.