THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e. g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

Spatial and Temporal Hierarchical Decomposition Methods for the Optimal Power Flow Problem

Rodrigo García Nava

Doctor of Philosophy

The University of Edinburgh

2022

Declaration

The work described in this thesis is the original work of the author and was carried out without the assistance of others, except where explicit credit is given in the text. It has not been submitted, in whole or in part for any other degree at any university.

Rodrigo García Nava
School of Mathematics
The University of Edinburgh
April 28, 2022

To my wife, Maria

Acknowledgments

I would like to thank my supervisor Ken McKinnon for the immense support provided these years; in our countless talks, he helped me understand concepts and challenged me to do better and be better. I learned how to think critically by his example. He was a fundamental guide on the ideas explored in this thesis and improved by his comments.

I would like to thank my beloved family, especially my parents, Margarita and Marco; they are present with me wherever I go and whatever I do. I would also like to thank Maria for putting up with me on the rollercoaster of emotions that is doing a PhD; her unconditional support is the main reason why I am submitting this thesis today. My deep gratitude also goes to Luis and Lety, who helped us design and attain a life that we love and cherish.

Further, I would like to thank anyone close to me in recent years. I have enjoyed this long journey with a fantastic group of friends who have become my Edinburgh family. I would like to thank the outstanding ERGO, particularly those involved in energy topics that have helped me expand my understanding of the energy universe. I would also like to thank the members of CESI for giving me interdisciplinary points of view.

Abstract

The subject of this thesis is the development of spatial and temporal decomposition methods for the optimal power flow problem, such as in the transmission-distribution network topologies. In this context, we propose novel decomposition interfaces and effective methodology for both the spatial and temporal dimensions applicable to linear and non-linear representations of the OPF problem.

These two decomposition strategies are combined with a Benders-based algorithm and have advantages in model building time, memory management and solving time. For example, in the 2880-period linear problems, the decomposition finds optimal solutions up to 50 times faster and allows even larger instances to be solved; and in multi-period non-linear problems with 48 periods, close-to-optimal feasible solutions are found 7 times faster.

With these decompositions, detailed networks can be optimized in coordination, effectively exploiting the value of the time-linked elements in both transmission and distribution levels while speeding up the solution process, preserving privacy, and adding flexibility when dealing with different models at each level.

In the non-linear methodology, significant challenges, such as active set determination, instability and non-convex overestimations, may hinder its effectiveness, and they are addressed, making the proposed methodology more robust and stable.

A test network was constructed by combining standard publicly available networks resulting in nearly 1000 buses and lines with up to 8760 connected periods; several interfaces were presented depending on the problem type and its topology using a modified Benders algorithm. Insight was given into why a Benders-based decomposition was used for this type of problem instead of a common alternative: ADMM.

The methodology is useful mainly in two sets of applications: when highly detailed long-term linear operational problems need to be solved, such as in planning frameworks where the operational problems solved assume no prior knowledge; and in full AC-OPF problems where prior information from historic solutions can be used to speed up convergence.

Lay Summary

The electricity system is a fundamental component of our societies and modern life. It is a highly complex interconnected system: even switching on a single light in our home has a small immediate effect throughout the system and the energy consumed must be supplied elsewhere and transferred through the system. The system is composed of thousands of generators and demands for power which are connected by a network of lines, and each component of the system had its own capabilities, behaviour and cost of use.

Finding the power flows that satisfy the laws of physics and meet the power demands and system capabilities is known as the Power Flow problem (PF). When an objective function is added to the PF problem, such as minimizing the operational cost, the problem is known as the Optimal Power Flow problem (OPF). The OPF problem is the heart of a cost-effective modern electricity system.

The OPF problem is challenging to solve. Different modelling approaches and solution methods have been developed to solve variations of this problem. Some models focus on accurate physics, others on approximations of the flow physics, and others on grouping elements to keep the problem size small. There is always a trade-off between speed and precision in mathematical optimization.

In recent years, the electricity system has been evolving with the introduction of renewable generation and storage distributed over the system. This has increased the size of the problems and the precision required in their modelling and solution.

In mathematical optimization, when a problem grows in size, it may be convenient to solve it in parts in what is called a decomposition. Decompositions can exploit the structure of the problem, potentially accelerating the problem's solution.

In this work, we present the development of new spatial and temporal decompositions for different variations of the OPF problem. The spatial decomposition separates the problem based on the location of its elements, and the temporal decomposition separates it depending on the moment it is happening. The decomposition is coordinated by a master problem that uses iterative sampling to build progressive approximations of the parts (subproblems) and converges to the optimal of the entire

system. Another benefit of the proposed decompositions is their ability to solve the problem without having to gather all the detailed information of the different regions in one place, so allowing the details of the regions to remain private.

Contents

List of Figures

List of Tables

List of Algorithms

Notation

Model

Sets

Sym.	Ch.	Description
$\mathscr{B}$	5,6	Set of buses indexed by b
$\mathscr{B}_r^A$	5,6	expanded network, with new aux. buses for region.
$\mathscr{D}$	5,6	Set of all demands, indexed by d
$\mathscr{D}^R$	5,6	Subset of load demands, indexed by d
$\mathscr{D}^N$	5,6	Subset of negative demands (renewable), indexed by d
$\mathscr{F}$	5,6	Set of fuels, indexed by f
$\mathscr{G}$	5,6	Set of all generators, indexed by g
$\mathscr{G}_r^{Ex}$	5,6	subset of exchange generators in region r
$\mathscr{G}_r^F$	5,6	subset of flexibility generators in region r
$\mathscr{G}_r^{FS}$	5,6	subset of flexibility storage in region r
$\mathscr{G}^T$	5,6	Subset of thermal generators, indexed by g
$\mathscr{G}^R$	5,6	Subset of renewable generators, indexed by g
$\mathscr{G}^S$	5,6	Subset of stores, indexed by g
$\mathscr{G}_r^{SCr}$	5,6	subset of coordinator storage in region r
$\mathscr{G}_r^{SCd}$	5,6	subset of coordinated storage in region r
$\mathscr{G}_r'$	5,6	expanded set of generators in region r, including auxiliary generators for every tie-line and for coordinator/coordinated storage
$\mathscr{L}$	5,6	Set of lines, indexed by l
$\mathscr{L}_r'$	5	subset of lines including the tie-lines with adjusted properties

$\mathscr{P}$	5,6	Set of demand profiles, indexed by p
$\mathscr{P}_r$	5	set of profiles in region r
$\mathscr{R}^{DN}$	5,6	set of distribution network regions
$\mathscr{R}^{TN}$	5,6	Transmission network region where $\mathscr{R} = \mathscr{R}^{TN} \cup \mathscr{R}^{DN}$
$\mathscr{S}$	5,6	Set of time slices, indexed by s
$\mathscr{T}_s$	5,6	Set of periods in slice s, indexed by t

Parameters

Sym.	Ch.	Description
β	5,6	System base
B_l	6	Line susceptance; $l \in \mathscr{L}$
B_b^B	6	Bus shunt suceptance; $b \in \mathscr{B}$
B_d^P	5,6	Base load for real power for demand d
B_d^Q	5,6	Base load for reactive power for demand d
B_l^{Sh}	6	Shunt line susceptance; $l \in \mathscr{L}$ $= \frac{-X_l}{R_l^2 + X_l^2}$
C^{CO_2}	5,6	CO_2 tax
C_g^{Curt}	5,6	Penalty cost for curtailment of generation; $g \in \mathscr{G}$
C_g^{Fix}	5,6	Pro Rata (hourly) fixed generation cost; ; $g \in \mathscr{G}$
C_g^{Flex}	5	Penalty costs for flexibility, $C^{Feas} >> C^{Shed}$; $g \in G_r^F \cup G_r^{FS}$
C_g^{FeasG}	6	Penalty costs for gen. feasibility, $C^{FeasG} >> C^{Shed}$; $g \in G_r^F \cup G_r^{FS}$
C_b^{FeasV}	6	Penalty costs for voltage feasibility, $C^{FeasV} >>> C^{Shed}$; $b \in \{\hat{b}_r\}$
C_f^{Fuel}	5,6	Fuel cost; $f \in \mathscr{F}$
C_g^L	5,6	Linear operational generation cost; ; $g \in \mathscr{G}$
C_d^{Shed}	5,6	Penalty cost for demand shed; $d \in \mathscr{D}$
C_p^{Shed}	6	Shed cost of profile p
Δt	5,6	Time step
D_{pst}	6	Demand level for profile p, in slice s and period t
D_{pst}^P	5	Profile level (unitless) for real demand in period t
η_g	5,6	Generator / storage round-trip efficiency; $g \in \mathscr{G}$
E_f	5,6	Fuel emission factor; $f \in \mathscr{F}$

$\bar{E}_g^G$	5,6	Energy capacity for store g; $g \in \mathscr{G}^S$
f	6	map from line l to the bus b on the from-side; $f : l \mapsto b$
G_l	6	Line conductance; $l \in \mathscr{L}$ $= \frac{R_l}{R_l^2 + X_l^2}$
G_b^B	6	Bus shunt conductance; $b \in \mathscr{B}$
G_l^{Sh}	6	Shunt line conductance; $l \in \mathscr{L}$
H	6	Curvature matrix of cut
m	6	Mapping function from demand d to profile p; $m : d \mapsto p$
n	6	Mapping function from generator g to fuel f; $n : g \mapsto f$
o	6	Map from ren. gen. g to profile p; $m : g \mapsto p$
L_l^{Ratio}	6	Tap ratio; $l \in \mathscr{L}$
P_{dst}^D	5,6	Real power demand, $d \in \mathscr{D}^R, s \in \mathscr{S}, t \in \mathscr{T}_s$ $= B_d^P D_{m(d)st}, \quad m : d \mapsto p$
$\underline{P}_{\_}^G{}_g$	5,6	Minimum real power generation of generator; $g \in \mathscr{G}$
$\bar{P}_g^G$	5,6	Maximum real power generation / discharge of generator; $g \in \mathscr{G}$
$\bar{P}_g^{G_{in}}$	5,6	Maximum real power discharge of store; $g \in \mathscr{G}^S$
Q_{dst}^D	5,6	Reactive power demand, $d \in \mathscr{D}^R, s \in \mathscr{S}, t \in \mathscr{T}_s$ $= B_d^Q D_{m(d)st}, \quad m : d \mapsto p$
$\underline{Q}_{\_}^G{}_g$	6	Min. reactive power gen. (abs.) of generator / react. device; $g \in \mathscr{G}^T$
$\bar{Q}_g^G$	6	Max. reactive power gen. (abso.) of generator / react. device; $g \in \mathscr{G}^T$
R_l	6	Line resistance; $l \in \mathscr{L}$
$\bar{S}_l$	5,6	Line thermal limit; $l \in \mathscr{L}$
t	6	map from line l to the bus b on the to-side; $t : l \mapsto b$
$\underline{V}_b$	6	Voltage magnitude lower bound; $b \in \mathscr{B}$
$\bar{V}_b$	6	Voltage magnitude upper bound ; $b \in \mathscr{B}$
V_{bst}^{Req}	6	Voltage prop. by the MP at bus b; $b \in \{\hat{b}_r\}, s \in \mathscr{S}, t \in \mathscr{T}_s$
v_{bst}^{Req}	6	Voltage square proposed by the MP at bus b; $b \in \{\hat{b}_r\}, s \in \mathscr{S}, t \in \mathscr{T}_s$ $= \left(V_{bst}^{Req}\right)^2$
X_l	5,6	Line reactance; $l \in \mathscr{L}$

Variables

Sym.	Ch.	Description	
α_{rst}	5,6	Approx. function of the cost for DN region r;	$r \in \mathscr{R}^{DN}, s \in \mathscr{S}, t \in \mathscr{T}_s$
d_{pst}	6	Aux. variable, for duals on profile level;	$r \in \mathscr{P}, s \in \mathscr{S}, t \in \mathscr{T}_s$
d^P_{pst}	6	Aux. var. to adjust the cut for prof. level p;	$p \in \mathscr{D}^P, s \in \mathscr{S}, t \in \mathscr{T}_s$
ϵ^P_{bst}	5	Dummy variable for LMP at b;	$b \in \mathscr{B}, s \in \mathscr{S}, t \in \mathscr{T}_s$
e^G_{gst}	5,6	Energy in store g in period t;	$g \in \mathscr{G}^S, s \in \mathscr{S}, t \in \mathscr{T}_s$
λ^p_{gst}	5,6	Dual var. for fixing real power;	$p \in \mathscr{G}^F$
λ^{pS}_{gst}	5,6	Dual var. for fixing coord. storage;	$p \in \mathscr{G}^{FS}$
λ^D_{pst}	5,6	Dual var. for fixing demand profile levels;	$p \in \mathscr{D}^P$
λ^x_{rst}	5,6	Dual var. concat. (Subsec. 6.3.7)	
l^{Shed}_{pst}	6	Profile level shed proportion;	$p \in \mathscr{P}$
p^G_{gst}	5,6	Real power generation (inj.) from gen/store g;	$g \in \mathscr{G}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^G_{gst}	5,6	real power discharge of coordinator storage g;	$g \in \mathscr{G}^{SCr}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{Curt}_{gst}	5,6	Real power curt. of generator g;	$g \in \mathscr{G}^R, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{F+}_{gst}	5,6	Real power positive feas. deviation;	$g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{F-}_{gst}	5,6	Real power negative feas. deviation;	$g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{in}_{gst}	5,6	Real power charging of store g;	$g \in \mathscr{G}^S, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{in}_{gst}	5,6	real power charging of coordinator storage g;	$g \in \mathscr{G}^{SCr}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{Net}_{gst}	5,6	net power requested of coordinator storage g;	$g \in \mathscr{G}^{SCr}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^L_{lst}	5	Real power transfer through line l;	$l \in \mathscr{L}, s \in \mathscr{S}, t \in \mathscr{T}_s$
$p^{L^f}_{lst}$	6	Real power coming out "from" bus of line l;	$l \in \mathscr{L}, s \in \mathscr{S}, t \in \mathscr{T}_s$
$p^{L^t}_{lst}$	6	Real power "entering" the "to" bus of line l;	$l \in \mathscr{L}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{Shed}_{dst}	5	Real power shed;	$d \in \mathscr{D}^R, s \in \mathscr{S}, t \in \mathscr{T}_s$
q^{F+}_{gst}	6	Reactive power positive feas. deviation;	$g \in \mathscr{G}^F, s \in \mathscr{S}, t \in \mathscr{T}_s$
q^{F-}_{gst}	6	Reactive power negative feas. deviation;	$g \in \mathscr{G}^F, s \in \mathscr{S}, t \in \mathscr{T}_s$
q^G_{gst}	6	Reactive power gen./abs. from generator g;	$g \in \mathscr{G}, s \in \mathscr{S}, t \in \mathscr{T}_s$
$q^{L^f}_{lst}$	6	Reactive power coming "from" bus of line l;	$l \in \mathscr{L}, s \in \mathscr{S}, t \in \mathscr{T}_s$
$q^{L^t}_{lst}$	6	Reactive power going "to" bus of line l;	$l \in \mathscr{L}, s \in \mathscr{S}, t \in \mathscr{T}_s$
θ_{bst}	5,6	Voltage angle at bus b;	$b \in \mathscr{B}, s \in \mathscr{S}, t \in \mathscr{T}_s$
v_{bst}	6	Voltage magnitude at bus b;	$b \in \mathscr{B}, s \in \mathscr{S}, t \in \mathscr{T}_s$

v_{bst}	6	Voltage squared at bus b;	$b \in \{\hat{b}_r\}, s \in \mathscr{S}, t \in \mathscr{T}_s$
v^{F+}_{bst}	6	Voltage squared positive feas. deviation;	$b \in \{\hat{b}_r\}, s \in \mathscr{S}, t \in \mathscr{T}_s$
v^{F-}_{bst}	6	Voltage squared negative feas. deviation;	$b \in \{\hat{b}_r\}, s \in \mathscr{S}, t \in \mathscr{T}_s$
x_{rst}	5,6	variable concat. (Subsec. 6.3.7)	

Algorithms

Sym.	Ch.	Description
α_{rst}	5,6	Approx. variable for the height of obj. of subproblem SP_{rst}
α^*_{rst}	5,6	value of approximation α_{rst}
$\mathscr{A}^i_r$	5,6	Unique active sets for region r, found up to iteration i
B^0	6	Starting matrix for SR1
e_s	5	No. of samples for the exploration strategy for a given period length
e_l	5	No. of periods in the auxiliary subproblem
C^{Act}	6	List of cuts defined as active
C^{Check}	6	Flag to indicate that the active cuts have been checked
C^{Cut}_{rast}	6	Clearance of cut
$\mathscr{E}$	5	Progressive exploration strategy (set of tuples)
ϵ^{Act}	6	Tolerance for a cut to be defined as active
ϵ^{AS}	6	Tolerance for the classification of an active set
ϵ^{Conv}	6	Tolerance for subproblem to be classified as “converged”
ϵ^{Norm}	6	Distance threshold for SR1
ϵ^{Rel}	5,6	Desired relative gap of convergence
ϵ^{SR1}	6	SR1 minimum matrix update
G^{Rel}	5,6	Relative gap between bounds for the master problem
$\hat{G}$	5,6	Sum of gaps between subproblem real height and approximation
H_{rast}	6	Fiited approx. curvature of the active set a, for region r.
i	5,6	Iteration counter
$\bar{I}$	5,6	Maximum desired number of Benders iterations
$\mathscr{I}_r$	6	List of inequalities of the subproblem in region r

$\mathscr{I}_r^+$	6	List of greater-than inequalities of the subproblem in region r
$\mathscr{I}_r^-$	6	List of less-than inequalities of the subproblem in region r
K^{Pen}	6	Coefficient (or vector of coefficients) for the stabilisation term
L_{rast}	6	Gradient for the active set a and region r
L^{Act}	6	List of cuts classified as active
λ_{rst}^{I*}	5,6	Multipliers of the inequality constraints in the subproblem SP_{rst}.
λ_{rst}^{SP*}	5,6	Gradients for the interface at the optimal solution
MP_i	5,6	Master problem instance, with information up to iteration i
$\mathscr{P}^{Conv}$	6	Set of converged subproblems
$\mathscr{P}^{SlViol}$	6	Set of slightly violated subproblems
$\mathscr{P}^{Viol}$	6	Set of violated subproblems
p_i^{Stab}	6	Penalty component in the master problem objective
p_i^{Stab*}	6	Value of the penalty of the master problem objective
$\mathscr{R}^{DN}$	5,6	Set of regions for the distribution networks
$\mathscr{S}$	5,6	Set of slices
SP_{rst}	5,6	Subproblem instance for region r, slice s and period t
$\mathscr{T}_s$	5,6	Set of time periods in slice s
W_s	5,6	Weight of slice s
x_{rst}	5,6	Variables to be fixed by the MP in the subproblem SP_{rst}
x_{rst}^*	5,6	Value of the variables fixed by the MP to the subproblem SP_{rst}
x_{rst}^{i*}	6	Value of the fixed x_{rst}^* in iteration i
x^{i*}	6	Value of the fixed x_{rst}^* for all subproblems in iteration i
X_{rast}	6	Historic fixed variable value for active set a and region r.
$\mathscr{X}_{ra}$	5,6	Information in the sample dictionary for region r and active set a
$\mathscr{X}_r$	5,6	Information in the dictionary of samples for region r, $\mathscr{X}_r \cup_{a\in} \mathscr{X}_{ra}$
χ_{rst}	5,6	New information from SP_{rst} for the dictionary of samples
χ_j^X	6	Point "x" component in sampled point j
χ_j^L	6	Gradient "λ" component in sampled point j
$\underline{\chi}_{rst}^X$	6	Guidance lower bound for interface variables
$\bar{\chi}_{rst}^X$	6	Guidance upper bound for interface variables
$\underline{Z}$	5,6	Master problem lower bound

$\bar{Z}$	5,6	Master problem upper bound
Z_{rast}	6	Historical objective value for active set a and region r.
Z^{Best}	6	Best upper bound
z^{MP}	6	Objective function of the MP as defined in Expression (6.7a)
z_i^{MP}	6	Objective function of the MP in iteration i with stab. pentalty
z_i^{MP*}	5,6	Optimal objective value of the MP in iteration i
z_{rst}^{SP*}	5,6	Objective value of the subproblem

Chapter 1

Introduction

The energy systems around the world are changing and changing fast, and its future uncertain, it is enough evidence to see the numerous future scenarios produced by different energy agencies: IEA [2], BP [3], NatGrid [4], among others; what seems certain is that the dynamism present in the energy system today will increase [5], calling for faster and more detail-rich methods for planning and operating it.

The electric power system is not an exception. The challenge of managing it has grown in size and complexity as the penetration of renewable technologies increases, and the system starts to be deeply cross-linked with different energy vectors; which need to be co-optimized [6] in order to achieve the sustainable energy targets set in COP26 [7].

Today, the power system incorporates intermittent generation, introducing uncertainty in how these systems are operated and planned. We have different market structures combined with environmental constraints calling for efficiency and optimization methods. The power is no longer flowing in one direction, and now at the distribution levels, power generation can flow to transmission; storage starts to impact strategies to dispatch power [8].

The power system has transformed (and keeps changing) from a “static” top-down approach that was centrally planned and operated to a more dynamic market-driven multi-directional distributed one, integrated with new technologies and new agents [9]. The data revolution that has changed our lives in the last 20 years is starting to permeate into power systems, with things such as smart meters, and smart controls

for rooftop PV panels and home batteries [10],[11]. Electrification of transport is growing every year, it is enough to see the trend on electric car units from 1.2m in 2015 to an expected 269m in 2030, and the electrification of heat is an important target [12].

With the inminent change in load profiles (e.g. charging electric vehicles and local storage) and the possible change in market structures [13] much of the attention needs to be devoted to understanding the dynamics between transmission and distribution levels [14].

The evolution of the power system has motivated our interest in two inter-twined research directions: first, it is imperative to work with accurate representations of distribution networks because they are growing in complexity for managing and planning operations, and unlike before, they can profoundly affect the development and operation of transmission networks; and second, with the growth in size, complexity, communication and distributed processing power, the problems need to be solved efficiently, reformulated in decomposable structures potentially assisted by parallel solution approaches. Distributed methods applied to power systems are a topic of many researchers today with many recent publications [15].

The revolution in the distribution networks (DNs) requires more and more detailed representations when connecting them to the Transmission Network (TN), since DNs have become active agents that can not only import power but can also export real and reactive power to support energy requirements and other ancillary services anywhere in the network [14]. To be able to unleash the full potential of the new distributed generation technologies, detailed down-the-network representations combined with robust and efficient methods are required [16]; e.g. if the TN can make good use of the reactive power regulation capabilities of a device in a distribution network, it may be possible to benefit the whole system operating at a lower cost.

The integration of distributed energy resources in the DNs affects deeply the operation of the power system [17]. There are potential technical and economic benefits from the distributed resources, such that different support schemes have been implemented in numerous countries [18]. In the UK, for example, the Smart Export Guarantee pays small producers for renewable electricity they have generated

and put into the grid [19].

Even if the benefits that distributed energy resources (in the DNs) can provide to the system are valuable, there are challenges for a coordinated TN-DNs operation [20]. The coordination is critical to exploit the potential flexibility and benefits from this interaction. The benefits can include for example voltage control, network congestion management, power loss reductions, and other ancillary services, ultimately enabling a more cost-effective and power efficient system [21].

As the TN and DNs are operated, controlled, and planned by different agents [22], the optimization problem calls for distributed methods that use the responsible agent's expertise, confidentiality, and processing power. The TN-DN power optimization problem is a distributed problem with interconnected networks, and algorithms for decomposed formulations need to be developed to exploit and preserve these objectives.

The structure of the thesis is as follows:

Chapter 2 presents a general review of common optimal power flow (OPF) representations, relaxations and approximations that will be referenced later in the thesis.

Chapter 3 presents a review of decomposition methods that have been applied to distributed OPF problems.

Chapter 4 presents the test case and the time-series used to test the proposed algorithms.

Chapter 5 introduces the proposed methodology applicable to (linear) OPF problems using the DC approximation and Chapter 6 extends the methodology for Full AC-OPF problems.

Chapter 7 briefly discusses the use of a simple ADMM method for the problem of interest, and finally, Chapter 8 summarises the conclusions and gives possible extensions of the research.

Additionally, Appendix A illustrates the proposed methodology from Chapter 6 in a 1-variable problem and illustrates the concept of "dictionary of samples", Appendix B explains the concept of demand, demand profile and base load used in the OPF models and Appendix C contains additional tables omitted from the main body of the thesis for compactness.

Chapter 2

Optimal Power Flow

According to [1], the heart of a realistic and economically efficient power system relies on the so-called optimal power flow problem (OPF); OPF-based problems are paramount for both operation and investment planning of energy systems [23], whether, for example, to define suitable places for new generation plants that comply with network capacity, or to determine locational marginal prices. In [24], the authors present a tool tailored for the Northeast US Market using OPF for investment planning. Other authors have focused on OPF based problems for analysing Locational Marginal Prices (LMP) and the effect of congestions in them, such as in [25].

The OPF problem was first presented by Carpentier in 1962 [26]. As the name implies, the objective of this problem is to find a feasible steady-state solution that optimizes some objective (commonly total operational cost or line losses), respecting the physics of power flow and other constraints (such as generator capacity or voltage limits) while fulfilling a required power demand.

As it is an optimization problem, it can be written as a general constrained optimization problem:

$$\min_{x \in \mathbb{R}^n} f(x) \tag{2.1a}$$

subject to:

$$h(x) \leq 0, \tag{2.1b}$$

where f is the objective function, x are the optimization variables defined in a domain $\mathbb{R}^n$, and h is the general vector of constraints.

The general OPF problem is here rewritten in terms of an objective function (e.g. cost minimization), generation and bus bounds and a power flow model as follows:

$$\min \quad \sum_{g \in \mathscr{G}} \left[C_g^q (p_g)^2 + C_g^l (p_g) + C_g^z \right] \tag{2.2a}$$

subject to:

$$\underline{P}_g \leq p_g \leq \bar{P}_g \qquad \forall\, g \in \mathscr{G} \tag{2.2b}$$

$$\underline{Q}_g \leq q_g \leq \bar{Q}_g \qquad \forall\, g \in \mathscr{G} \tag{2.2c}$$

$$\underline{V}_b^2 \leq \|v_b\| \leq \bar{V}_b^2 \qquad \forall\, b \in \mathscr{B} \tag{2.2d}$$

$$f_{ik}(v_i, v_k) \leq \bar{I}_{i,k} \qquad \forall\, (i,k) \in \mathscr{L} \tag{2.2e}$$

and power balance equations (2.2f)

where the set $\mathscr{B}$ represents the set of buses in the network, $\mathscr{G}$ the set of generators, and $\mathscr{L}$ the set of lines. $\underline{P}$, $\bar{P}$, $\underline{Q}$, $\bar{Q}$ are the parameters for minimum and maximum real and reactive power for a generator. $\bar{I}_{i,k}$ is the electrical current limit (or transferred power limit) in a line connecting bus i to bus k, $\underline{V}$ and $\bar{V}$ are the voltage limits for a bus, and finally the parameters C^q, C^l, C^z are the cost coefficients for quadratic, linear and constant respective parts for each generator. $f_{ik}(v_i, v_k)$ represents an appropriate power flow function for the line. The variables for this formulation are the active and reactive power of the generators p, q, voltages of the buses v.

For the OPF problem, approximations, relaxations and optimization techniques have been developed and implemented, enabling the solution of instances from different applications and with different complexities. According to [1] some of today's problems require the solution techniques to be sped up by 3 to 5 orders of magnitude.

The OPF problem has been thoroughly studied since Carpentier, but it is still a very active research area. This is because: first, by it is a complex problem (NP-

Hard in general [27]), and second, by it is the core of many practical problems today, such as energy market pricing, capacity expansion and topology planning, robust operation in security-constrained networks, energy re-dispatch, unit commitment, network control and post-contingency restoration.

There have been efforts to understand and solve OPF problems exactly and also to solve them approximately, and to develop techniques and approaches that make large-scale problems manageable such as relaxations that give bounds to the solutions or provide certificates of infeasibility [28].

It is essential to realise that different formulations of power flow require different optimization techniques: some of them are general non-linear non-convex optimizations, while others use the techniques of semidefinite programming (SPD), conic programming (CP), linear programming (LP) and mixed-integer linear and non-linear programming (MILP, MILNP).

There is no single taxonomy for power flow models. In [29] the authors do an extensive survey of the recent developments in the field, and similar to Low [30], they classified the power flow models in two groups: bus injection models (BIM) and branch flow models (BFM). Most of the power flow representations in literature [1] belong to the bus injection model in their rectangular or polar form. The classification here presented is in line with that survey.

The notation of this chapter is described within the chapter after it is presented (unlike Chapters 5 and 6 where all their notation is found at the beginning of the thesis).

2.1 Bus Injection Models

Bus injection models of the power flow relate the flow to the electrical quantities at each bus. For these models, there are two different approaches, namely the I-V formulation and the voltage-based formulation, each of which has two different versions: rectangular and polar [29].

2.1.1 I-V formulation

The I-V formulation is based on two fundamental characteristics of AC power systems: 1) the linear relationship between the voltage phasors and current injection phasors and 2) the definition of complex power. The rectangular formulation is then:

$$I_i = \sum_{k \in \mathscr{B}} \mathbf{Y}_{ik} v_k \qquad \forall\, i \in \mathscr{B} \tag{2.3a}$$

$$p_i + j q_i = v_i \bar{I}_i \qquad \forall\, i \in \mathscr{B} \tag{2.3b}$$

where I is the complex current injected at bus i, v is the complex voltage, and p and q represent the real and the reactive power injected at the bus, i and k refer to the nodes in the system and $\bar{(\cdot)}$ is the complex conjugate operator. $\mathbf{Y}$ is to the admittance matrix, built accordingly to the network topology and the physical characteristics of the line and is defined as $\mathbf{Y} = \mathbf{G} + \mathbf{jB}$.

In $\mathbf{Y}$, $\mathbf{G}$ is the conductance of the lines and $\mathbf{B}$ is their susceptance, $\bar{(\cdot)}$ is the complex conjugate operator and v_i and v_k are the complex voltages at bus i and k respectively.

Note that the non-linearities of the formulation are isolated in the bilinear terms of the complex power definition (2.3b), and they are only associated with a single bus. This formulation facilitates the straightforward representations of devices whose current flows cannot be expressed as functions only of their terminal voltage (ideal transformers and ideal circuit breakers) [29], on the other hand, this formulation has more variables than the ones that are voltage based.

2.1.2 Voltage-based formulations (PQV)

In the Rectangular PQV formulation, each bus has an real and reactive power injection expressed by a complex number (phasor), and the voltage is also defined in complex numbers as $|v_i|e^{\mathbf{j}\theta}$. The power injections are represented into their real and imaginary parts, and the squared voltage magnitude is defined as $v_i \bar{v}_i = |v_i|^2$.

By substituting the current injection (Eq. 2.3a) equation into the power injection equation (Eq. 2.3b) the voltage formulation model is obtained:

$$p_i + jq_i = v_i \sum_{b \in \mathscr{B}} \bar{\mathbf{Y}}_{ik} \bar{v}_k \qquad \forall\, i \in \mathscr{B} \tag{2.4}$$

The equations vary depending on how the admittance matrix, voltage phasors and power injections are represented (rectangular or polar). The polar equations will be presented here, as these equations are used in the problems explained in later sections. The formulation is then:

$$p_i = |v_i| \sum_{k \in \mathscr{B}} |v_k| \left(\mathbf{G}_{ik} \cos(\theta_i - \theta_k) + \mathbf{B}_{ik} \sin(\theta_i - \theta_k)\right) \qquad \forall\, i \in \mathscr{B} \tag{2.5a}$$

$$q_i = |v_i| \sum_{k \in \mathscr{B}} |v_k| \left(\mathbf{G}_{ik} \sin(\theta_i - \theta_k) - \mathbf{B}_{ik} \cos(\theta_i - \theta_k)\right) \qquad \forall\, i \in \mathscr{B} \tag{2.5b}$$

Where $|v_i|$ and θ_i refer to the voltage magnitude and voltage angle at Bus i, i.e. $v_i = |v_i| e^{j\theta_i}$.

These two equivalent formulations (I-V and PQV) are both full (non-convex) AC-OPF formulations, and they represent the true physics of power flow in terms of the power injections. It is important to mention that the formulations presented in this section are general formulations; the explicit constraints of nodal load balance (i.e. a differentiation between load, line import, line exports, generation and shunt charging) can be represented following the principles of the equations written in this section. The complete problem statements will not be presented here as formulations presented in Chapters 5 and 6 will be explicit.

Table 2.1 shows the characteristics of the rectangular IV, rectangular PQV and polar PQV formulations [1].

Formulation	Network Constraints	Voltage Angle Difference	Bus Constraints
Rectangular IV	$2\|\mathscr{L}\|$ linear equalities	Linear	Locally quad.
Rectangular PQV	$2\|\mathscr{L}\|$ quadratic equalities	Non-convex (arctan)	Non-convex quad. ineq.
Polar PQV	$2\|\mathscr{L}\|$ non-linear equality constraints with quad. terms and sine/cosine	Linear	Linear

Table 2.1: Characteristics of different formulations for AC-OPF, adapted from [1]

2.2 Branch Flow Model

Baran and Wu presented in [31] a "simpler" power flow representation called *DistFlow* (also known as Branch Flow Model or BFM). This was developed for radial networks, for which it had numerical computational advantages over the Bus Injection Models. The BFM equations focus on the quantities flowing in the lines (as opposed to those reaching the bus as in the previous models).

The BFM assumes a directed graph with an arbitrary direction with the sending-end power flows, i.e. $i \to k$, from i to k. The BFM model is:

$$p_{ik} = R_{ik}\ell_{ik} - p_k + \sum_{m:k\to m} p_{km} \qquad \forall (i,k) \in \mathscr{L} \qquad (2.6a)$$

$$q_{ik} = X_{ik}\ell_{ik} - q_k + \sum_{m:k\to m} q_{km} \qquad \forall (i,k) \in \mathscr{L} \qquad (2.6b)$$

$$u_k = u_i - 2\left(R_{ik}\,p_{ik} + X_{ik}\,q_{ik} + \ell_{ik}\,[R_{ik}^2 + X_{ik}^2]\right) \qquad \forall (i,k) \in \mathscr{L} \qquad (2.6c)$$

$$\ell_{ik}\,u_i = p_{ik}^2 + q_{ik}^2 \qquad \forall (i,k) \in \mathscr{L} \qquad (2.6d)$$

Where R and X are the series impedance (resistance and reactance, respectively), ℓ the squared magnitude of the current ($|I|^2$) and u the squared magnitude of the voltage ($|v|^2$). The BFM model is exact for radial networks but only a relaxation in meshed networks, as it does not guarantee the consistency of voltage angles. The radial model also eliminates the phase angles of voltage and currents. Figure 2.1 shows the variables and parameters for the line (i,k).

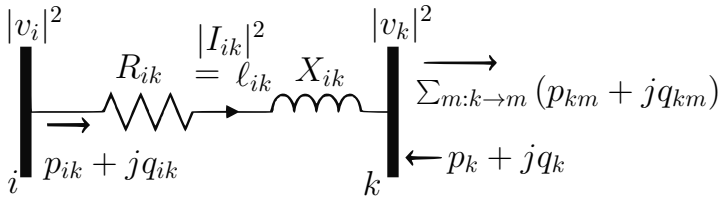


Figure 2.1: Branch Flow variables for line $(i,k) \in \mathscr{L}$. Taken from [29]

2.3 Relaxations and approximations

The exact representations of power flow are non-convex for general networks. However, there are relaxations and approximations which are useful depending on the

practical application. The difference between the exact non-convex space and the relaxations and approximations is illustrated in Figure 2.2.

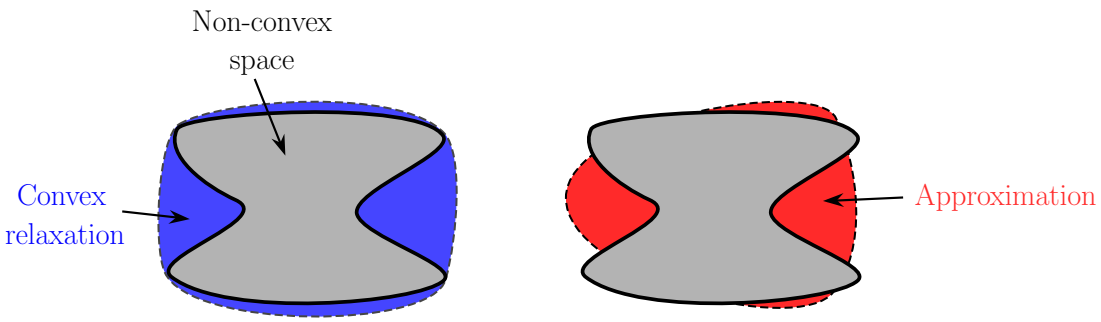


Figure 2.2: Conceptual differences between non-convex space, convex relaxation and approximation

Convex relaxations enclose, in a convex space, the non-convex feasible space of the power flow equations. Their solution provides bounds for optimization problems (lower bounds when minimising); they are also helpful because they can give an infeasibility certificate for the original problem if they are infeasible.

Another exciting feature of some relaxations is that they are exact for specific problems, which can be exploited by using convex optimization solvers.

In [29] there is a review of more than 20 relaxations and approximations, comparing them in terms of tightness, complexity and limitations. Many of the relaxations are focused on finding a global optimum for the problems, which is not the focus of the problem tackled in this thesis; the interest aims to find local optima and near-optimal feasible solutions.

Jabr's SOCP relaxation

One common convex relaxation for the OPF problem is Jabr's SOCP relaxation [32]. This relaxation is based on the bus injection model presented previously (eqs. 2.5), but it is modified by introducing the following:

$$c_{ii} = |v_i|^2 \tag{2.7a}$$

$$c_{ik} = |v_i||v_k|\cos(\theta_i - \theta_k) \tag{2.7b}$$

$$s_{ik} = -|v_i||v_k|\sin(\theta_i - \theta_k) \tag{2.7c}$$

Giving the following equations:

$$p_i = \mathbf{G}_{ii} c_{ii} + \sum_{k \in \mathcal{N} | k \neq i} \mathbf{G}_{ik} c_{ik} - \mathbf{B}_{ik} s_{ik} \qquad \forall i \in \mathcal{B} \qquad (2.8a)$$

$$p_i = -\mathbf{B}_{ii} c_{ii} + \sum_{k \in \mathcal{N} | k \neq i} -\mathbf{B}_{ik} c_{ik} - \mathbf{G}_{ik} s_{ik} \qquad \forall i \in \mathcal{B} \qquad (2.8b)$$

$$c_{ik} = c_{ki} \qquad \forall (i,k) \in \mathcal{L} \qquad (2.8c)$$

$$s_{ik} = -s_{ki} \qquad \forall (i,k) \in \mathcal{L} \qquad (2.8d)$$

$$c_{ik}^2 + s_{ik}^2 = c_{ii} c_{kk} \qquad \forall (i,k) \in \mathcal{L} \qquad (2.8e)$$

Where c_{ik} and s_{ik} are defined as described in (2.8) and (2.8). Jabr's Relaxation is an exact representation for radial networks (under a mild condition [33]), but this is not the case with meshed networks as it does not ensure the ability to recover a set of voltages that sum to zero around each loop. If the constraint $\tan(\theta_k - \theta_i) = \frac{s_{ik}}{c_{ik}}$ is added, then the formulation is equivalent to the original Bus Injection power flow model. Even without this constraint the formulation is still non-convex due to eq. (2.8e). A convex relaxation is formed by replacing eq. (2.8e) with its relaxation (2.8f), which is convex.

$$c_{ik}^2 + s_{ik}^2 \leq c_{ii} c_{kk} \qquad \forall (i,k) \in \mathcal{L} \qquad (2.8f)$$

DistFlow relaxation

The Branch Flow model equations presented in Section 2.2 are an exact representation for radial power networks but a relaxation for meshed networks, as they don't ensure angle consistency. The Branch Flow formulation is still non-convex as it includes eq. 2.6d, the convex relaxation consists in replacing eq. 2.6d with eq. 2.9 (in a similar fashion to Jabr's Relaxation):

$$p_{ik}^2 + q_{ik}^2 \leq \ell_{ik} v_i \qquad \forall (i,k) \in \mathcal{L} \qquad (2.9)$$

DC Power Flow approximation

Finally, a common approximation widely used in many applications is the DC power flow approximation. The derivation is closely related to a Taylor's expansion of the power flow equation around the voltage profile $V_i = 1.0\angle 0^\circ$ in all buses, neglecting the line resistances R_{ik} and using the small-angle difference approximation $\sin(\theta_i - \theta_k) \approx \theta_i - \theta_k$. It also assumes near-nominal voltage magnitudes $|V| = 1$ and ignores reactive power and line losses. The model is reduced to:

$$p_i = \sum_{(i,k)\in\mathscr{L}} B_{ik}(\theta_i - \theta_k)$$

where B is the susceptance of the line connecting i and k, p is the active power and the variables θ_i and θ_k are the bus angles of i and k.

The DC approximation is well suited for applications where reactive power or voltage levels are not of interest (e.g. long-term planning problems or market-based studies). It is generally not suitable for detailed representations of distribution networks, as the voltage limits may be active constraints more often than line limits, and their lines have high resistance to reactance ratios. The benefit of this approximation is that LP techniques can solve it, and these can handle large-scale power system problems efficiently.

minimized

Chapter 3

Distributed methods applied to OPF-based problems

In the last years, many papers related to power network "spatial" decompositions have been published. In the introduction of [34], the author mentions numerous papers that deal with spatial decompositions for power systems. In line with his review, in this Thesis, the decomposition approaches for this type of problem are classified as:

1. Augmented Lagrangian based decompositions
2. Optimality condition decomposition
3. Benders decomposition

3.1 Augmented Lagrangian based methods

The first group is based on Lagrangian and augmented Lagrangian methods and includes Dual Decomposition and the Alternating Direction Method of Multipliers (ADMM). ADMM is a commonly used method in recent literature regarding spatial decompositions of power systems.

3.1.1 ADMM

The Alternating Direction Method of Multipliers (ADMM) has experienced a resurgence in recent years due to its robustness and simplicity, which has been well

received and widely used in the machine learning research community. Nonetheless, ADMM is an "old" concept that first originated in the 1970s by the works of Mercier and Gabay [35], Glowinski and Marocco [36], and others. Gabay [37], and Eckstein and Bertsekas [38] developed the proofs for the convergence properties of ADMM back in 1983 and 1992, respectively. Gabay also showed that ADMM is based on a generalised method developed in the 1950s called the Douglas-Rachford splitting method.

The idea of ADMM is the result of combining two previously developed algorithms: dual decomposition and the method of multipliers. ADMM is an algorithm intended to blend the decomposability of dual ascent with the convergence of the method of multipliers.

The augmented Lagrangian methods were developed to improve the robustness of the dual ascent methods and, to yield convergence without the need of strict convexity of the original problem [39]. Given the original optimization problem, similar to the one presented in (2.1):

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to} \quad & Ax - b = 0 \end{aligned}$$

its augmented Lagrangian is:

$$L_p(x, y) = f(x) + y^T (Ax - b) + (\rho/2) \| Ax - b \|_2^2$$

where $\rho > 0$ is the penalty parameter. The difficulty in using an augmented Lagrangian method is that it destroys the separability of the problem. To overcome this issue, the variables are split (x and z) and minimized in a sequential alternating fashion using a Gauss-Seidel approach.

ADMM can be used to solve problems of the form:

$$\begin{aligned} \min \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned}$$

by building the augmented Lagrangian for the problem:

$$L_p(x,z,y) = f(x) + g(z)\, y^T (Ax + Bz - c) + (\rho/2)\, \|Ax + Bz - c\|_2^2$$

with the updates every iteration:

$$\begin{aligned} x^{k+1} &:= \operatorname*{argmin}_x L_p(x, z^k, y^k) \\ z^{k+1} &:= \operatorname*{argmin}_z L_p(x^{k+1}, z, y^k) \\ y^{k+1} &:= y^k + \rho^k (Ax^{k+1} + Bz^{k+1} - c) \end{aligned}$$

If $f(x)$ and $g(x)$ are convex, the constraint residuals will converge to zero. According to [40], the iterations quickly converge to moderate accuracy but take many iterations to converge to high accuracy, and their convergence rate will depend on the choice of ρ.

The general description of the standard ADMM is shown in Algorithm 3.1.

Algorithm 3.1 Standard ADMM algorithm, adapted from [39]

procedure INITIALIZATION
  Require: $z^0, y^1, \rho, \epsilon^{pri}, \epsilon^{dual}$
  Set: $k := 1, \left\| r^1 \right\|_2 \leftarrow \infty, \left\| s^1 \right\|_2 \leftarrow \infty$
end procedure
procedure ITERATION
  while $\left\| r^k \right\|_2 \geq \epsilon^{pri}$ or $\left\| s^k \right\|_2 \geq \epsilon^{dual}$ **do**
    Compute primal variables:
    $x^{k+1} = \operatorname{argmin}_x L_p(x, z^k, y^k)$
    $z^{k+1} = \operatorname{argmin}_z L_p(x^{k+1}, z, y^k)$
    Compute dual variables:
    $y^{k+1} = y^k + \rho^k (Ax^{k+1} + Bz^{k+1} - c)$
    Compute the primal and dual residuals:
    $r^{k+1} = Ax^{k+1} + Bz^{k+1} - c$
    $s^{k+1} = \rho A^T B(z^{k+1} - z^k)$
    $k := k + 1$
  end while
end procedure

The algorithm terminates when the size of the primal residuals (r^k) and dual

residuals (s^k) is small enough. r^k is related to the constraint violation (the consensus constraints), and s^k is related to the "asynchrony" or change of decisions of one side of the problem between consecutive iterations (for details, see [39]). The definition of the residuals is shown in Lines 13-14 of Algorithm 3.1.

As ρ plays a fundamental role in the convergence speed of the algorithm, much effort has been directed to adjusting it dynamically; this reduces the impact of a poor initial value. In [39] one of the ideas is to adjust ρ depending on the relative sizes of the primal and dual residuals trying to balance them. The scheme presented is:

$$\rho^{k+1} := \begin{cases} \tau^{incr}\rho^k & \text{if } \left\|r^k\right\|_2 > \mu \left\|s^k\right\|_2 \\ \rho^k/\tau^{decr} & \text{if } \left\|s^k\right\|_2 > \mu \left\|r^k\right\|_2 \\ \tau^{incr} & \text{otherwise} \end{cases}$$

Where τ^{incr}, τ^{decr} and μ are all greater than 1. Another idea also presented in [39] is the selection of different ρ for every constraint that is being "relaxed" or coordinated, achieved by replacing the term $\left(\rho/2\right)\|r\|_2^2$ with $(1/2)\,r^T P r$, where P is a symmetric positive definite matrix.

A common formulation is shown in Chapter 7. The formulation is done by consensus; in OPF problems, the typical consensus variables consist of local copies of the bus voltages, the bus angles or the power transmitted through the tie-lines. An excellent explanation and diagram showing this process can be found in [41].

The ADMM algorithm can be applied to non-convex problems but is not guaranteed to converge. In [42] the author presented a variation that converges in non-convex problems, provided that the penalty parameters are bounded and an interior point method solver is used which always finds a local solution.

3.1.2 ALADIN

ALADIN (Augmented Lagrangian based on Alternating Direction Inexact Newton) is a recent algorithm that was developed by Houska *et al.* in [43]. It is based on augmented Lagrangian and closely related to ADMM.

ALADIN is designed for non-convex distributed optimization. It combines features of ADMM with the fast convergence of SQPs, and by using second-order infor-

mation, enables distributed consensus optimization problems to be solved in far fewer iterations.

For the proof of global convergence a suitable step size must be chosen and in [43] a subroutine is proposed to achieve this.

The method is beneficial for weakly coupled non-convex problems (when the coupling is a refinement more than a dominant feature). As a general description, the algorithm consists of five steps:

1. Solve the decoupled NLPs (in parallel).
2. Check if solution satisfies convergence criteria. If so, stop.
3. Compute gradients, hessian (or hessian approximations) and constraint Jacobians, and detect the active set.
4. Construct consensus QP based on local sensitivities and active sets
5. Apply step-update strategies

One important drawback of ALADIN is that unlike ADMM, it requires a global dual update (i.e. a central coordinating entity). Reference [44] includes the full OPF formulation and an example of the algorithm for a regional decomposition of a power transmission network.

In [45] the authors make a comparison between ADMM and ALADIN for different IEEE test cases where the advantages in convergence speed are shown but also the drawbacks of the increased communication requirements.

3.2 Optimality Condition Decomposition

Optimality Condition Decomposition (OCD) is based on the distributed solution of the KKT conditions and has been used to address power system problems.

Conejo et al. [46] presented OCD and defined it as a particular implementation of the Lagrangian Relaxation procedure, motivated by a natural decomposition of the optimality conditions of the original problem.

In the decomposition, a problem is divided into blocks of variables to be optimized with their corresponding “inner" constraints and selected complicating constraints (each block resulting in a subproblem). Complicating constraints refers

to those constraints that if relaxed, then the problem can be easily separated. The complicating constraints involving variables from different blocks drastically complicate the solution of the problem and prevent its solution by blocks [47]. By fixing the value of the variables not present in the block, each subproblem can then be optimized independently, and once the solutions of the subproblems have been computed, the multipliers of the complicating constraints can be updated.

In OCD there is no need for any procedure to jointly update the multipliers as the complicating constraints are kept in the block subproblem. OCD has the advantage that the convergence properties do not require attaining the optimal solution of the subproblems at each algorithm iteration. It is enough to perform a single optimization step (e.g. a single interior point ieration) for each subproblem and update the variable values. As a result, this method can significantly reduce computation time. In order to terminate the algorithm it is necessary to verify that the solutions of the block variables and the multipliers are optimal in the original problem. In [47] the authors present an illustrative equality constrained case where a single newton step is performed in each subproblem per iteration and compared in convergence speed against traditional Lagrangian relaxation decompositions and show that OCD converges faster than an augmented Lagragangian and also faster than a Lagrangian relaxation approach. In [48], the authors present OCD applied to an OPF problem also showing a faster convergence when compared to traditional Lagrangian relaxations and augmented Lagrangian methods.

3.3 Benders decomposition

Benders algorithm decomposes the problem into a master problem and sub-problems, where a master problem is augmented with approximating functions α of the subproblems. In general, Benders decomposition can solve linear programming problems in a distributed manner. Only a general overview will be given here; the algorithm and the implications of non-convexity for the problem of interest will be explained in Chapter 6.

As an overview, the steps followed by the algorithm are:

1. Approximate $\alpha(x_1, ..., x_n)$ from below using hyperplanes.
2. Solve the master problem using these approximations
3. Improve the approximations using additional hyperplanes, generated by solving the subproblems with derivative information.
4. If the approximation is good enough, stop.

Benders decomposition is used for problems with complicating variables, i.e. variables that, if fixed or known, the resulting problem would become separable and so easier. Benders decomposition deals with problems of the form:

$$\min_{x_1,...,x_n\,;\,y_1,...,y_m} \quad \sum_{i=1}^{n} c_i x_i + \sum_{j=1}^{m} d_j y_j \tag{3.1}$$

subject to:

$$\sum_{i=1}^{n} a_{li} x_i + \sum_{j=1}^{m} e_{lj} y_j \le b_l\,; \quad l = 1, \ldots, q \tag{3.1a}$$

$$0 \le x_i \le \bar{X}_i\,; \quad i = 1, \ldots, n \tag{3.1b}$$

$$0 \le y_j \le \bar{Y}_j\,; \quad j = 1, \ldots, m \tag{3.1c}$$

In Problem 3.1, we choose x as the variable in the Master Problem. Problem 3.1 can be alternatively written as:

$$\min_{x_1,...,x_n} \quad \sum_{i=1}^{n} c_i x_i + \alpha(x_1, \ldots, x_n) \tag{3.2}$$

subject to:

$$0 \le x_i \le \bar{X}_i\,; \quad i = 1, \ldots, n \tag{3.2a}$$

where:

$$\alpha(x_1, \ldots, x_n) = \min_{y_1,...,y_m} \quad \sum_{j=1}^{m} d_j y_j \tag{3.2b}$$

subject to:

$$\sum_{j=1}^{m} e_{lj} y_j \leq b_l - \sum_{i=1}^{n} a_{li} x_i \,; \quad l = 1, \ldots, q \tag{3.2c}$$

$$0 \leq y_j \leq \bar{Y}_j \,; \qquad j = 1, \ldots, m \tag{3.2d}$$

where $\alpha(x_1, \ldots, x_n)$ is the function that provides the optimal objective function for the problem 3.2b for given values of $(x_1, \ldots, x_n)$.

For Benders decomposition, the problem structure will include a Master Problem (3.3) and a subproblem or subproblems (3.4). The form of the Master problem and the subproblems is the following:

Master Problem:

$$\min_{x_1, \ldots, x_n, \alpha} \quad \sum_{i=1}^{n} c_i x_i + \alpha \tag{3.3}$$

subject to:

$$\sum_{j=1}^{m} d_j y_j^{(k)} + \sum_{i=1}^{n} \lambda_i^{(k)} (x_i - x_i^{(k)}) \leq \alpha \,; \quad k = 1, \ldots, v \tag{3.3a}$$

$$0 \leq x_i \leq \bar{X}_i \,; \quad i = 1, \ldots, n \tag{3.3b}$$

$$\underline{\alpha} \leq \alpha \tag{3.3c}$$

where v is the current Benders iteration.

Subproblem at iteration k:

$$\min_{y_1, \ldots, y_m} \quad \sum_{j=1}^{m} d_j y_j$$

subject to:

$$\sum_{i=1}^{n} a_{li} x_i + \sum_{j=1}^{m} e_{lj} y_j \leq b_l \,; \qquad l = 1, \ldots, q \tag{3.3d}$$

$$0 \leq y_j \leq \bar{Y}_j \,; \qquad j = 1, \ldots, m \tag{3.3e}$$

$$x_i = x_i^{(k)} : \lambda_i \,; \quad i = 1, \ldots, n \tag{3.3f}$$

In (3.3) α is the approximation to the function $\alpha(x_1, \ldots, x_n)$ in 3.2b, the value surface. In Benders, the function α is defined as approximating a piecewise linear function that is progressively constructed.The approximating function is built with information gathered by sampling the subproblem. Each time the subproblem is solved the master problem is informed of its height ($\sum_j^m = d_j y_j$), and gradient (λ_i) with respect to the fixed variables ($x_i^{(k)}$) and the MP will construct a first-order approximation (3.3a) to the subproblem at the sampled point k. Each sampled point can potentially refine the precision of the approximation either by excluding a sampling area (feasibility cut) or better describing the objective within the feasible region (optimality cut).

In standard Benders, where the master problem and the subproblems are linear, the algorithm will converge to the optimal after a finite number of iterations, by adding a finite number of locally improving cuts enriching the approximation of the subproblems in the master problem [49].

In the non-linear side, using a Benders-like algorithm for non-linear problems is not a new concept, Generalized Benders Decomposition (GBD) was introduced by Geoffrion in 1972 [50]. GBD is a procedure to solve a broader class of problems where the subproblem no longer needs to be a linear program, and nonlinear convex duality theory is employed to derive the families of cuts corresponding to those in the Benders cases. The GBD is a Benders-based algorithm that is useful for nonlinear problems under some convex and regularity assumptions [50].

Later, the GBD method was extended to solving problems that do not meet convexity requirements [51], although these implementations may only provide local optima, whereas in other cases not even convergence can be achieved [52].

OPF-based problems solved with GBD can be found for example in [53], and more recently in [54]. In all of these cases, the non-convex OPF subproblem has been altered either by convexification or linearization iteratively to allow the use of GBD with algorithmic convergence guarantees. In [55] the authors restrict the domain of the subproblem and modify the OPF so it is only mildly non-convex but without guarantees.

Chapter 4

Test Cases

The hierarchical decompositions proposed in Chapters 5 and 6 exploit the structure of the transmission-distribution power system and its topology. To avoid having a vastly detailed model of the elements in each network, the test cases focus on the general structure of these problems, so there is no specific mention of modelling tap-changers, specific capacities of transformers, AC three-phase balance, and others. All the elements which are included in the models are mentioned in their specific chapters and vary depending on the type of OPF being solved. The test cases are constituted by a single network topology (described below) with several different time-horizons modelled both with DC and AC-OPF formulations, depending the type of problem to be solved.

The methodology proposed in this thesis can be applied to general power network topologies, taking into account the modifications required depending on the elements to be coordinated (see Section 5.3.7), regardless if the networks are weakly or strongly linked. However, there is a balance between the usefulness of the information exploited by the decomposition and the interface's size, which can affect the solution speed (against the undecomposed version).

The network designed for the test cases involves relatively weakly-linked networks with low dimensional Benders cuts, where we believe the balance of useful information and level of decomposition is convenient speed-wise. Additionally, the network topology tries to represent TN-DNs, where the bulk of generation occurs at the TN level, and the power demand at the DN level, with relatively simple links

between them. The choice of a unique network obeys the fact that we prioritise diversity in OPF model types tested rather than exhibiting different networks, which can be decomposed in line with what is presented in Section 5.3.7.

If we wanted to stress-test by modifying the network topology, the first step would include having multiple links between a DN and the TN, and adding links DN-DN; these modifications would require coordination of voltage angles (in the AC-OPF cases) and would increase the dimension of the interface. Then the next level of stress would be considering any standard meshed network topology, e.g. IEEE or Pegase cases, with an arbitrary spatial decomposition.

4.1 Networks

The problems targeted are multi-period OPF with several distribution networks, each treated in the decomposition as a single-period subproblem. A synthetic instance that gives a simplified representation of a large meshed transmission network is called *Reduced GB Network* specified in [56]. This network includes 29 buses, 66 generators and 99 lines. Its topology is shown in Figure 4.1 over-imposed on a Great Britain map to give a sense of scale and physical counterpart. While the map shows the general area of the aggregated buses is not supposed to be precise; however the topology used for the test cases comes directly from the information found in the source. This network is a highly meshed network composed of many duplicated lines with several links, in particular in the south of England and the Midlands.

Each distribution network is modelled as a 33-bus radial (tree) network. The topology of the distribution network was taken from a standard IEEE case known as *IEEE33BW* was presented in [57] and its topology is shown in Figure 4.2.

In order to create a large-scale network with a transmission-distribution topology, each bus in the transmission network has been connected to Bus 1 in the distribution network via a line with small impedance and resistance, representing a nearly ideal transformer. The connection is illustrated in Figure 4.3; the variables and constraints needed to couple the networks will be described in the decomposition chapters.

Once the network is put together, there are 1056 lines and 986 buses, corresponding to 29 buses in the transmission network and 33 buses in each distribution

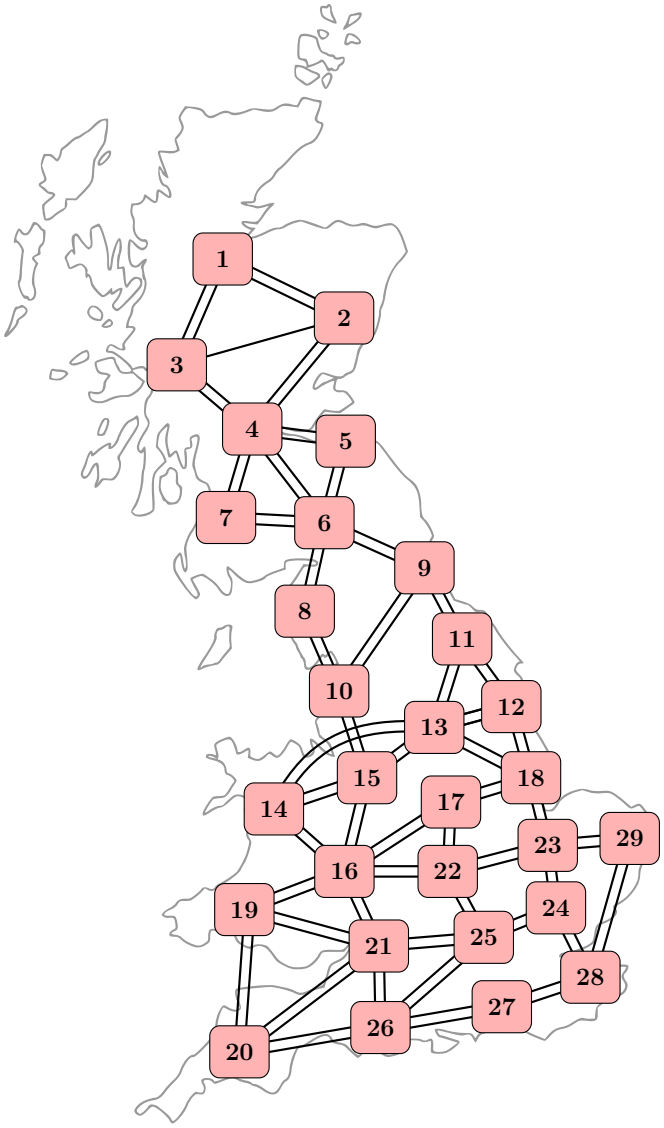


Figure 4.1: Reduced GB, adapted from [56]

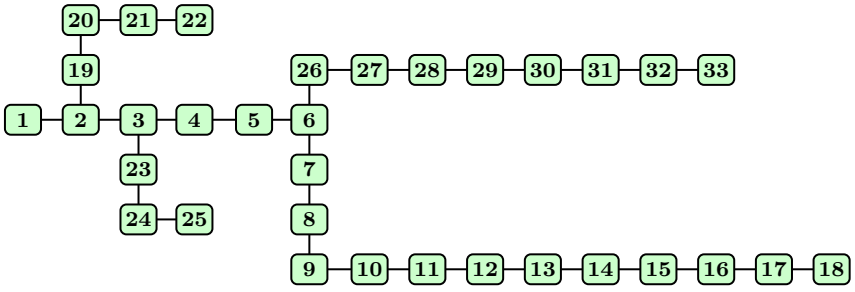


Figure 4.2: IEEE33BW, adapted from [57]

network.

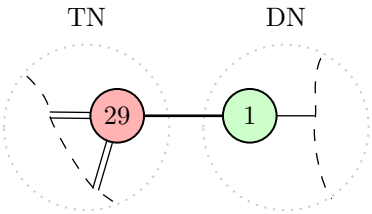


Figure 4.3: TN-DN connected by low resistance/reactance element

4.2 Generation

The ReducedGB network used for the transmission network has 66 generators and its data file gives their installed capacity, location and also their unit cost; however, the data file specifies that the cost for the generators is made up. Upon further inspection, there are only two types of costs: a group with almost zero cost and non-zero costs for the other group.

The installed capacities of the zero cost group are close to the historical total installed capacity of wind and solar renewables in 2015, so we assume these zero cost generators are wind and solar in the test case, and split them between solar and wind so to match the historical installed capacity per technology given in [58].

The non-zero cost generators were divided to match the thermal generation capacities by type in 2015, and the costs imposed come from [59]. Two large pumped hydro stores are connected to Buses 19 and 23 with their sizes also taken from [59].

In each distribution network, there is one thermal generator, two batteries and several reactive support devices (from 2 to 5 depending on the network) in diverse locations. Note that reactive support elements appear only in Chapter 6 which is where the AC power flow representation is used.

When put together, the total number of generators is 221, composed of 81 thermal generators, 28 renewable generators, 60 stores and 52 reactive support devices. In terms of storage, the total power discharge capacity is 11.5GW, including 8.7GW located in the distribution networks, and a total storage capacity of 103.2 GWh with 69.6 GWh at the distribution level.

4.3 Demand profiles

In the models presented in this thesis, the demands are based on two parameters: a base load and a profile level. The base loads represent the base (in power units) which is scaled according to a profile level (time series). The relationships between demand, profile and bus are clarified in a simple example in Appendix B.

This section describes the method for generating synthetic demands loosely based on the historical GB demands, but divided in 957 buses and 87 profiles (3 for each distribution network) for 8760 periods. A demand base load for real and reactive powers is needed in order to get a time-series (in power units) for each demand.

4.3.1 Base load

To define the demand base loads in the distribution buses a top-down approach was used. First, a calculation for a sensible base load was done at transmission level and then split into the buses of the distribution networks.

In the matpower data file of the Reduced GB Network [56], there are real and reactive demand loads specified for each one of the 29 transmission buses for a single period. The assumption in the test case is that they represent the proportional split of load in GB, i.e. if the total real power demand in the case is 56.7GW and Bus 25 (around London) is 9.7GW, then approximately 17% of the load will be assumed when creating synthetic time-series for each one of the 29 buses.

The first step was to use the proportional split at the transmission level mentioned in the previous paragraphs. The base was defined for the TN-Buses to have different profile combinations; for the residential part, proportions from 50% to 82% were used, for the commercial part between 10% and 38% and finally for the industrial part between 2% and 27%. For example, Bus 17 (around Birmingham) has proportions of 50%, 23% and 27% for the residential, commercial and industrial part respectively, and Bus 27 (around Bristol) has 77%, 15% and 8%. At the end of this process a base load (for each profile) was allocated to each TN bus; for example, from the 9.7GW of real power in Bus 25, 7.6GW (78%) correspond to residential, 1.5GW (15%) to commercial and 0.7GW (7%) to industrial base loads.

Once that the base load data had been calculated (for the 29 transmission buses),

it was transferred to the distribution networks. In the matpower data file of IEEE33BW there are real and reactive loads associated to each bus for a single period. A proportional split of the total load in IEEE33BW was used to distribute the base load calculated at the transmission level. The proportional split of those loads (inside the DN) was used for the test cases, i.e. in some distribution buses, e.g. Bus 24, get 11% of the load whereas others, e.g. Bus 11, only get a 1% load. Finally, once the total load per bus was defined for each distribution network bus, it was split again with different proportions by profile.

After this process, each demand was associated with a base load for real and reactive power for the 957 distribution buses.

4.3.2 Profile level

The demands in the test case are defined by combining three different profiles levels that change differently in time. The three profiles considered are: residential, commercial and industrial.

In Figure 4.4 the 3 profiles are shown (for 48 half-hours in a weekday), these profiles were constructed to account for variability of load type during the day, and they are loosely based on comments from [60]. In each graph, the y-axis has been normalised to the relative level of the profile with respect to its highest level on the day, i.e. the period with the highest demand will attain a value of 100%, for each of the 3 profiles.

Once the base loads had been calculated (as explained in the previous section), the 3 demand profiles were generated to have a characteristic pattern representing weekly demand. When all the base loads are multiplied by their respective profile level for a given day, they were calibrated to roughly match the daily energy demand in [61]. The division between different days was smoothen to avoid unrealistic change-of-day jumps in demand.

The time series used for the reactive loads are the same as those for real loads. The difference is found in the base loads. The reactive base loads were calculated using the methodology described at the beginning of this section and come from the *ReducedGB* test case. For reference, the magnitude of the reactive demands is

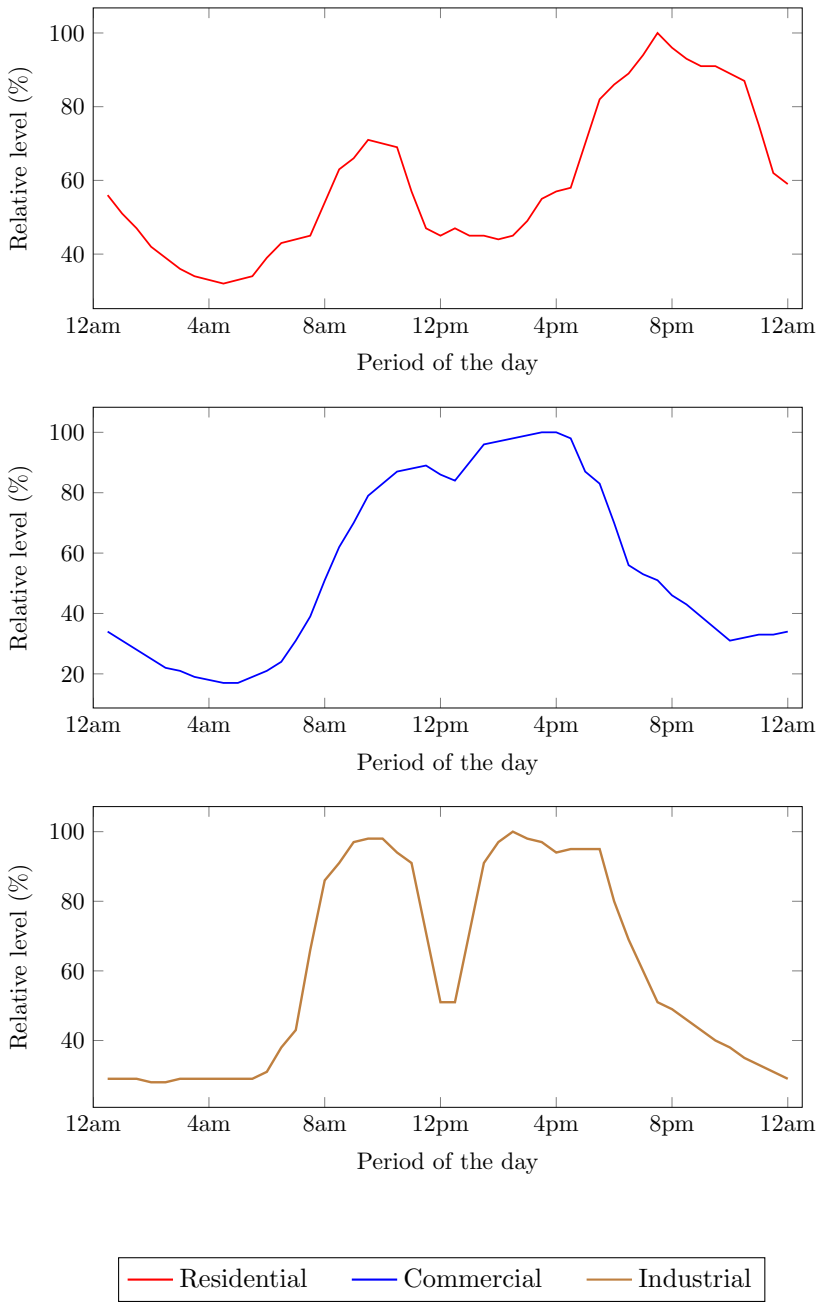


Figure 4.4: Demand Profiles

around 20-30% of that of the real demand.

After the process, the real and reactive demands were defined for 8760 periods for 87 profiles (3 per distribution network) that cover the 957 distribution buses. There are no demands in the transmission level in the test case, as they were all moved to the distribution level. For reference, the maximum and minimum real power demand in the time series corresponds to 54.8GW and 19.8GW, respectively, with a total yearly energy demand of 292.4TWh.

Similar to what was done with generation costs, the disconnection penalties (Value of lost load) are also taken from [59]. They are 40-100 times larger than the conventional generation costs, i.e. 3000 GBP/MWh; the same penalty was applied to reactive loads 3000 GBP/MVAr.

In terms of the renewable generators, in the models they are handled as curtailable negative demands; in the test cases, they are only located in the transmission level, and only a single profile was considered based on historic half-hourly wind generation in GB in 2015 [61], scaled to the relative sizes of the installed capacities. The same process as the wind was performed for the solar generation using a different historical half-hourly profile. In the test cases, the cost of curtailment was set to 0.

4.4 Discussion

The algorithm's test case focuses on the composition of a large scale problem with distributed resources, and it loosely represents the topology of a national level demand with regional networks. While the data sources come from official information of Great Britain, the cases do not pretend to be realistic. The development of this test case is focused on the algorithmic solution, but it still keeps sensible dimensions for load demands and costs.

Note that the specific values of generation costs, curtailment and lost load (shedding) should not affect or modify the proposed methodology from Chapters 5 and 6

Chapter 5

DC OPF-Based Problem

5.1 Motivation

As described in Chapter 2, the DC linear approximation is widely used for modelling transmission network problems, particularly in problems involving power market cost optimization and other problems that use binary or integer variables, such as unit commitment models, where the main requirement is to have fast and good enough approximation of the network.

In the standard DC approximation, the line losses are not considered; neither are the reactive powers nor the voltage magnitudes, which in real distribution networks are fundamental to understanding the network's power flow and keeping the system secure.

To simplify the network description, it is commonly assumed that the attached distribution networks can be represented as single point fixed loads at the connection point. This assumption makes it impossible to co-optimize the responsive elements (generators and stores) inside the distribution network with those at the transmission level.

Historically, this situation was not a concern due to the absence of responsive elements in the distribution networks, but now, that premise is no longer valid. Year after year, there is an increase of distributed generation and storage [62], and the trend is accelerating. The benefit of taking into account these elements begins to be significant, and they should be included to give better estimates of the optimal total

cost of generation towards a more realistic whole system optimization.

Based on the previous justification, on one side, there is a potentially strong case for a better (linear) representation of the network by taking the distribution networks and their elements into account (generators and stores), which will make use of the distributed generation and storage to decrease the total cost of the network. The big drawback comes from the explosion in the model's size and the need to collect the complete information of every distribution network.

Apart from the expensive computations from a detail-rich model, there is still the privacy issue, where every distribution network operator should give out the complete information about their network: topology, demand profile distribution (and disconnection costs), line limits and internal generator and stores characteristics (location, costs and power generation limits). This issue arises because there is potentially a different operating agent in every distribution network and a different one for the transmission network.

This section aims to solve a problem in a way that tackles the issues here presented. The proposed methodology has the following targets:

1. To have a more detailed distribution network representation optimizing the responsive elements in the distribution networks.
2. To solve the problem faster than an single undecomposed model while giving near-optimal or optimal solutions.
3. To reduce the amount of data exchanged, limiting what needs to be communicated, trying to preserve the privacy of the distribution network agents.
4. To be able to solve larger instances than a single undecomposed model.

With these objectives in mind, the original problem needs to be reformulated to separate the distribution networks from the transmission network. The steps of this process and the subsequent steps of solving the modified problem will be presented in this chapter. First, the original problem will be presented.

5.2 Undecomposed formulation

This section describes the optimization model using the DC power flow approximation. The objective is to minimize the total generation cost for a given demand

schedule, expressed as an average hourly operation cost, using optimally the generators and stores found in every part of the network. The decomposed formulation introduced later will be equivalent to the undecomposed formulation. The constraints and objective are presented in this secition, but the the notation can be found at the start of the thesis.

Constraints

$$-2\pi \leq \theta_{bst} \leq 2\pi \qquad \forall\, b \in \mathscr{B}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1a)$$

$$\underline{P}^G_g \leq p^G_{gst} \leq \bar{P}^G_g \qquad \forall\, g \in \mathscr{G}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1b)$$

$$0 \leq p^{Curt}_{gst} \leq p^G_{gst} \qquad \forall\, g \in \mathscr{G}^R, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1c)$$

$$0 \leq p^{in}_{gst} \leq \bar{P}^G_g \qquad \forall\, g \in \mathscr{G}^S, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1d)$$

$$0 \leq p^{Shed}_{dst} \leq P^D_{dst} \qquad \forall\, d \in \mathscr{D}^R, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1e)$$

$$0 \leq e^G_{gst} \leq \bar{E}^G_g \qquad \forall\, g \in \mathscr{G}^S, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1f)$$

$$-S^L_l \leq p^L_{lst} \leq S^L_l \qquad \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.1g)$$

$$\theta_{bst} = 0 \qquad b \in \mathscr{B}^{ref}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.2a)$$

$$e^G_{gs(t+1)} = e^G_{gst} + \Delta t \left(\eta_g\, p^{in}_{gst} - p^G_{gst}\right) \qquad \forall\, g \in \mathscr{G}^S, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.2b)$$

$$\frac{\epsilon^P_{bst}\,\beta}{|\mathscr{T}_s|} = 0 \qquad \forall\, b \in \mathscr{B}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.2c)$$

$$p^G_{gst} = D^P_{o(g)st} \qquad o : g \mapsto p,\ g \in \mathscr{G}^R, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.3a)$$

$$p^L_{lst} = \frac{\theta_{b'st} - \theta_{bst}}{X_l} \qquad b \in \mathscr{B}^f_l,\ b' \in \mathscr{B}^t_l\ \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.3b)$$

$$\sum_{g \in \mathscr{G}_b} p^G_{gst} \;-\; \sum_{l \in \mathscr{L}^f_b} p^L_{lst} \;+\; \sum_{l \in \mathscr{L}^t_b} p^L_{lst} =$$
$$\sum_{d \in \mathscr{D}^R_b} (P^D_{dst} - p^{Shed}_{dst}) \;+\; \sum_{g \in \mathscr{G}^R_b} p^{Curt}_{gst} + \sum_{g \in \mathscr{G}^S_b} p^{in}_{gst} \;+\; \epsilon^P_{bst}$$

$$\forall\, b \in \mathscr{B},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (5.3c)$$

Objective Function

$$z^* = \min \quad \sum_{s \in \mathscr{S}} \frac{W_s}{|\mathscr{T}_s|} \left(c_s^{Var} + c_s^{Shed} + c_s^{Curt} \right) \tag{5.4}$$

where:

$$c_s^{Var} = \sum_{t \in \mathscr{T}_s} \left(\sum_{g \in \mathscr{G}^T} \left[\beta\, p_{gst}^{G} \left(C_g^{L} + \frac{C_{n(g)}^{Fuel} + C^{CO_2} E_{n(g)}}{\eta_g} \right) + C_g^{Fix} \right] + \sum_{g \in \mathscr{G}^S} \left[\beta\, p_{gst}^{in} C_g^{L} + C_g^{Fix} \right] \right) \tag{5.4a}$$

$$c_s^{Shed} = \sum_{t \in \mathscr{T}_s} \sum_{d \in \mathscr{D}^R} \beta\, p_{dst}^{Shed}\, C_d^{Shed} \tag{5.4b}$$

$$c_s^{Curt} = \sum_{t \in \mathscr{T}_s} \sum_{g \in \mathscr{G}^R} \beta\, p_{gst}^{Curt}\, C_g^{Curt} \tag{5.4c}$$

$$n : g \mapsto f,\ \forall\, s \in \mathscr{S}$$

Here, Constraints (5.1) are two-sided inequality bounds on variables. Constraint (5.1a) sets the bounds for the voltage angle at buses, Constraint (5.1b) sets the bounds for real power generation and power discharge for generators and storages, respectively. Constraint (5.1c) sets the minimum and maximum curtailment of renewable generators to the power generated at that period.

Constraint (5.1d) sets the maximum charging input power to the stores, Constraint (5.1e) limits the minimum and maximum load shedding, depending on the real demand. Constraint (5.1f) sets the minimum and maximum energy in the stores, and finally, Constraint (5.1g) imposes line limits.

The next group corresponds to equality constraints (5.2). Constraint (5.2a) sets

the angle of the reference bus to an arbitrary value of 0, Constraint (5.2b) handles the energy level at the stores tracking how much energy enters or exits the store at any period. As a trick to easily query the scaled dual values of the load balance constraint (5.3c), the variable ϵ has been added to the constraint, and then in Constraint (5.2c), the scaled value of ϵ is fixed to zero, leaving the load balance constraint unaffected; querying directly from the solver the dual of this constraint will correspond to the (scaled) Locational Marginal Prices (LMPs). LMPs are commonly retrieved by querying the dual variables of the power balance constraint. However, these values have to be rescaled in this formulation since the objective function (5.4) has a scaling factor to report the objective value as the weighted hourly cost normalised by the system base β. Then constraint (5.2c) takes into account the scaling, making its dual value to correspond to the absolute increase in cost by one additional unit of demand at a specific bus.

The LMPs correspond to the marginal value of providing one additional unit of energy to a specific bus. These values may be relevant in frameworks such as power markets, where the market structure charges rates per location. The LMPs can reflect power transfer active limits that prevent the cheapest generator from providing energy to a bus, and they may also reflect the increase in price given the losses in the network. In the test cases explored in this thesis, the LMPs are not used, but they may be relevant in other use cases.

Constraint (5.3a) fixes the renewable generators to an associated negative demand and Constraint (5.3b) represents the DC power flow approximation in the network lines. Finally, Constraint (5.3c) handles the load balance at every bus in every period.

The objective function (5.4) is a minimization objective of the weighted average hourly costs and is composed of 3 parts: (5.4a) gathers generation costs, i.e. the sum of the power generation costs incurred by generators and the sum of the costs of charging the stores in all periods including fix cost; (5.4b) the penalty costs for unserved demand (disconnecting loads); and (5.4c) the penalties paid by renewable generation curtailment. The optimal objective value of the problem is named z^*.

In addition to the elements described previously, there are auxiliary mapping functions and auxiliary subsets for the creation of constraints. The definition P^D_{dst}

uses the mapping function m, which maps from demand d to profile p. In equation (5.4a) and the mapping function n is used to map from generator g to fuel f. Function o used in equation (5.3a) maps from a renewable generator g to a negative demand profile.

The auxiliary subsets used in this model are $\mathscr{B}^{Ref}$ which only contains the reference bus, $\mathscr{B}_l^f$ contains the bus where line l starts; similarly, $\mathscr{B}_l^t$ contains the bus where the line ends. $\mathscr{L}_b^f$ is the subset of all the lines that are coming out of bus b, and $\mathscr{L}_b^t$ is the subset of lines going out of bus b.

$\mathscr{D}_b^R$ is the subset of real demands incident to bus b, $\mathscr{G}_b^R$ is the set of incident renewable generators to bus b, and finally, $\mathscr{B}_b^S$ is the set of stores incident to bus b.

The problem defined in this section is a linear programming problem (LP), i.e. it only has linear constraints and linear objective function. While there are different ways of writing the constraints using auxiliary mapping functions and subsets differently, the presented formulation aims at clarity. There are redundant constraints and variables in the formulation, which it is assumed will be quickly and effortlessly detected and removed in the pre-solve phase of the solving process. A clear example of this is the variables p^G for renewable generators and the constraints that fix their values (5.3a). Based on the variables and constraints here presented, the size of the problem is shown in Table 5.1.

Table 5.1: Number of elements in the DC model formulation

Variables	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(2\lvert\mathscr{B}\rvert + \lvert\mathscr{G}\rvert + 2\lvert\mathscr{G}^S\rvert + \lvert\mathscr{G}^R\rvert + \lvert\mathscr{D}^R\rvert + \lvert\mathscr{L}\rvert\right)$
Constraints	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(3\lvert\mathscr{B}\rvert + \lvert\mathscr{G}\rvert + 3\lvert\mathscr{G}^S\rvert + 2\lvert\mathscr{G}^R\rvert + \lvert\mathscr{D}^R\rvert + \lvert\mathscr{L}\rvert + 1\right)$
Variables on the Obj.	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(\lvert\mathscr{G}\rvert + \lvert\mathscr{G}^S\rvert + \lvert\mathscr{G}^R\rvert + \lvert\mathscr{D}^R\rvert\right)$

Putting aside the privacy issues previously discussed, a more detailed system including transmission and distribution networks could have thousands of buses, lines, demands and generators, if that is multiplied by hundreds or thousands of periods where there is storage connecting them, the problem scales up to millions or hundreds of millions of variables and constraints. As a single undecomposed problem, solving a problem of this scale will be expensive in terms of time and memory requirements.

From both perspectives, privacy and problem size, the problem can benefit from

a reformulation allowing decomposition in space for the privacy argument; and time and space for the problem size argument.

5.3 Decomposition

With the targets introduced at the beginning of this chapter (Page 42), the problem is then reformulated to enable spatial and temporal decomposition. The spatial decomposition allows for creating separate regions, where most of the region information is contained and used inside the region's boundaries, and a temporal decomposition, reformulating the problem allowing for each region to solve only single-period problems.

These follow the standard Benders Decomposition approach (described in Section 3.3). The changes needed to achieve spatial and temporal decomposability are now presented.

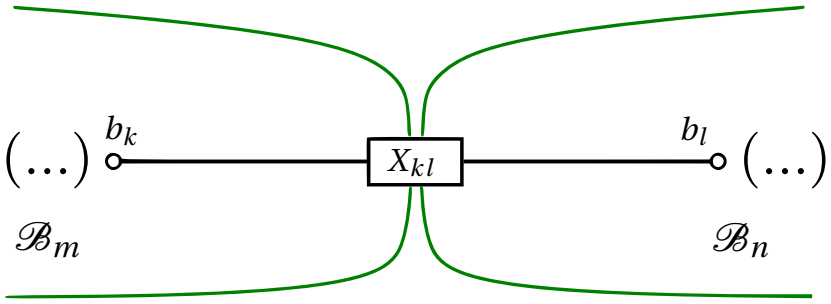
5.3.1 Spatial decomposition

In general, a network can be partitioned in r regions, where the elements belonging to that region are considered internal, and the elements crossing regional divisions are considered interregional elements. If a coordinating problem ensures that the interregional elements attain the same value in both regions, the resulting problem will be equivalent to the original problem.

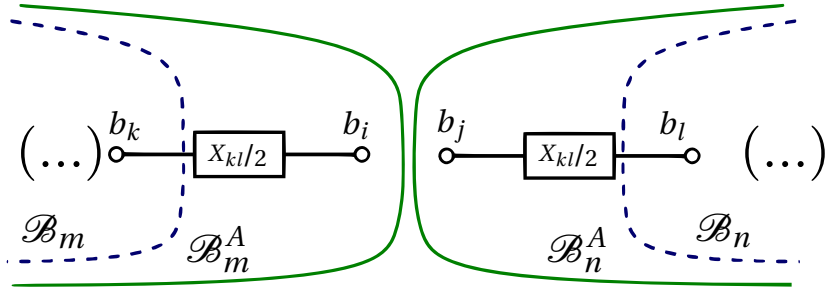
The coordination requirements or conditions can be progressively enforced, such as in Augmented Lagrangian approaches (e.g. ADMM, describe in sec. 3.1.1), or explicitly set, as in Benders decomposition. For the instances presented in Chapter 4, the division into regions is natural given the transmission-distribution structure of the power network.

There are different ways of splitting a network; one example is duplicating "real" buses and having a local copy of it on one side of the decomposition; this is not the approach in this thesis. The approach of this thesis is in line with [63], where the elements joining two regions (inter-regional lines) need to be split and then coordinated, and this is achieved by introducing fictitious buses and generators, as

will be explained later. While analogous to [63], this approach has the drawback of introducing more additional variables and constraints than a duplication of buses approach. These additional elements could produce degeneracy and complicate the problem. The effect on the test cases may be small and can be overlooked; in other cases, there are potentially better ways of splitting the network.



(a) Original model



(b) Modifications for separability

Figure 5.1: Spatial Decomposition. Separation in regions

Figure 5.1 shows the modification used for separating the networks into 2 different regions. In Figure 5.1a there is a unique line connecting buses b_k and b_l, with one end of the line located in region m and the other in region n, with the sets of buses of $\mathscr{B}_m$ and $\mathscr{B}_n$ for regions m and n, respectively. An interregional line $e : b_k \rightarrow b_l$ has a reactance X_{kl}.

As explained earlier, to achieve decomposability (Figure 5.1b), a pair of auxiliary buses is created at the midpoint of the interregional line. Bus b_i belongs to the augmented bus set $\mathscr{B}^A_m$ and bus b_j is part of set $\mathscr{B}^A_n$, and each region of the network gets one half of the line with the adjusted reactance of $1/2$ the original reactance. In strict terms, there is no need to split the line and adjust the reactances; the same effect can be achieved by duplicating the bus at the end of the line keeping the original bus in one region and the copy in the other.

The model can then be rewritten into an equivalent model, which considers the regions as separated entities with additional constraints. Given that this is a standard DC model without line losses, the interface variables are reduced to only the real power crossing the line and the voltage angles at its midpoint. The notation, for the modifications to the model found in this section, is found at the start of the thesis.

Changes to the objective function and constraints

$$z^* = \min \quad \sum_{s \in \mathscr{S}} \frac{W_s}{|\mathscr{T}_s|} \sum_{r \in \mathscr{R}} \left(c_r^{Var} + c_r^{Shed} + c_r^{Curt} \right) \tag{5.5a}$$

subject to:

$$p_l^L = p_m^L \qquad l \in \mathscr{L}_{rr'}^T,\ m \in \mathscr{L}_{r'r}^T\ \forall\, r \in \mathscr{R},\ \forall\, r' \in \mathscr{R},\ r \neq r' \tag{5.5b}$$

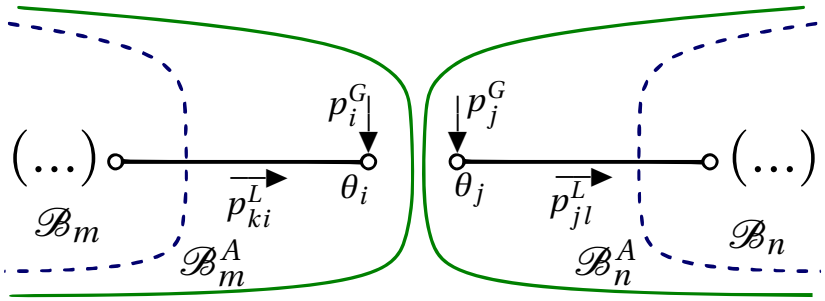
$$\theta_b = \theta_c \qquad b \in \{\hat{b}_r\},\ c \in \{\hat{b}_{r'}\}\ \forall\, r \in \mathscr{R},\ \forall\, r' \in \mathscr{R},\ r \neq r' \tag{5.5c}$$

$$\text{and the other constraints in Section 5.2 separated by regions.} \tag{5.5d}$$

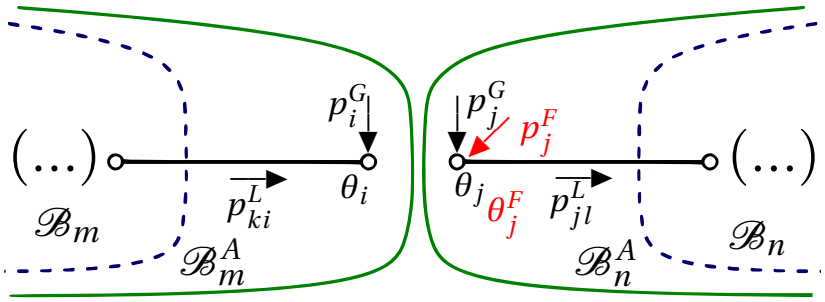
For clarity, slice and period indices (st) have been drop from expressions (5.5a)-(5.6). $\hat{b}_r$ refers to the augmented bus. Using standard elements in the model presented in the previous sections, the interface buses b_i and b_j have new "exchange" generators attached to them, and these generators have zero cost.

Depending on the method used to coordinate the exchange generators, more elements could be required to keep the separated problems feasible. In the case of a flexible link (e.g. ADMM), no additional elements are needed as the consensus variables (exchange generators on both sides) do not have to agree initially, and they will agree when the price signals are correct. With an "inflexible" link (fixed request), the interface needs to be further augmented to keep the problems feasible to any request.

In Figure 5.2a, there are additional power exchange generators, and to have a decomposed model equivalent to the original, power entering or exiting the auxiliary bus needs to be equal on both regions, and so has to be the voltage angle. This interface is enough for methodologies where both sides can disagree, such as ADMM.



(a) Basic auxiliary elements



(b) Augmented interface with flexibility elements

Figure 5.2: Auxiliary elements according to solution methodology

For the decomposition here proposed, using Benders algorithm, initially, the regions are partitioned into a transmission network (or central region) which will be present in the Master Problem, and in coordinated regions (subproblems, distribution networks).

In Benders, there is a direct assignment from one region (in the Master problem) to the other (subproblem), when the master problem region does not know the internal constraints of the subproblem, it is possible to assign values that would make the subproblem infeasible. In Benders, feasibility cuts can handle this situation, but only optimality cuts are considered in the approach presented here. To cope with an infeasible request, Figure 5.2b shows the elements needed to "flexibilise" the master problem requests by using a feasibility generator that carries a hefty penalty when used.

The agreement in voltage angle needs to be considered only when a subproblem (distribution region) has more than one inter-regional line to the master problem (transmission region), or in general, when the links back to the transmission network form a cycle, then there needs to be a variable θ in the interface, with a corresponding θ^F for request flexibility. θ^F will act as a phase angle regulator with a significant

penalty for deviating from the request.

In summary, the process for creating the spatial decomposed equivalent network is as follows:

1. **Detect tie-lines.** Check which lines are joining two buses in different regions ($b \in \mathscr{B}_r$, $b \in \mathscr{B}_{r'}$; r, $r' \in \mathscr{R}$ and $r \neq r'$).

2. **Create and connect auxiliary buses.** Create a local copy on both sides of an "auxiliary" bus at the middle of the tie-line ($\hat{b}_i$ and $\hat{b}_j$ where i, $j \in \mathscr{R}$) for every tie-line. The newly created auxiliary buses are then the end-point of the tie-lines for each side of the network, by adjusting the characteristics of the tie-line, i.e. dividing the reactance of the original line (X_l).

3. **Augment sets.** Enlarge the set $\mathscr{B}_r$ with the newly created bus $\hat{b}$ forming $\mathscr{B}_r^A$ for every region of interest. Also, create auxiliary exchange generators on the auxiliary bus that allows the "consensus power" (representing the power in the middle of the line) to be transferred, and augment the set of generators in the region with the new generator i.e. $\mathscr{G}_r' = \mathscr{G}_r \cup \{\hat{g}\}$. If flexibility to the requests is needed, create flexibility generators with the same process.

In the test cases introduced in Chapter 4, only one link between the transmission network and distribution network is present; and in the same way as the angle of the reference bus was set arbitrarily in the original model, every distribution region will set a bus angle value. For those cases, the variable θ^F is not needed, and it is only needed in more general networks that need to ensure angle consistency back to the master problem.

5.3.2 Temporal decomposition of the distribution networks

If constraints (5.2b) were not present in the problem, the original DC model (Section 5.2) would be decomposable in time. The existence of storage complicates the problem, and the optimization has to be done over the whole time horizon. With the regional decomposition (Subsection 5.3.1), the benefit relies on only solving a section of the complete network, but there is still time-linking in the subproblem.

The elements found in distribution networks are small fast response generators, renewable generators and small-sized storage. It is not unlikely to assume that the fast response generators are not subjected to ramping constraints (opposed to large thermal generators in transmission networks), and only the storage is linked in time.

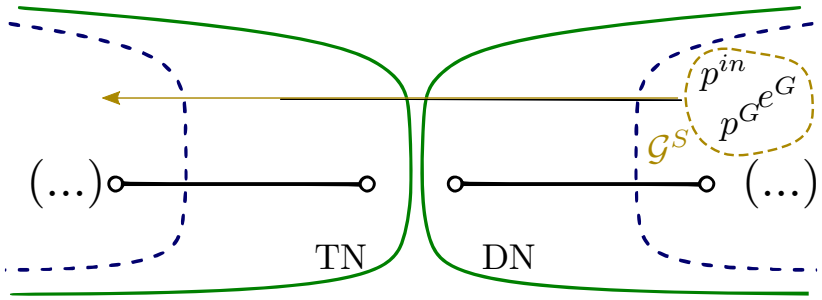
In the test cases presented in this thesis, the only time-linking constraints in the distribution network are the ones related to storage, and they are used as an example of this type of constraint. In general, with the temporal decomposition presented in this section, any time-linking constraint can be dealt with after performing the steps mentioned at the end of this section. This process enables the temporal decomposition, regardless of any other time-linking constraints, e.g. ramp rates.

If storage is taken as the example for the time-linking constraints, the temporal information about distributed storage is moved from the distribution networks to the master problem to achieve time-independence in the subproblems. Due to large-scale storage and generation, the transmission network already deals with temporal information.

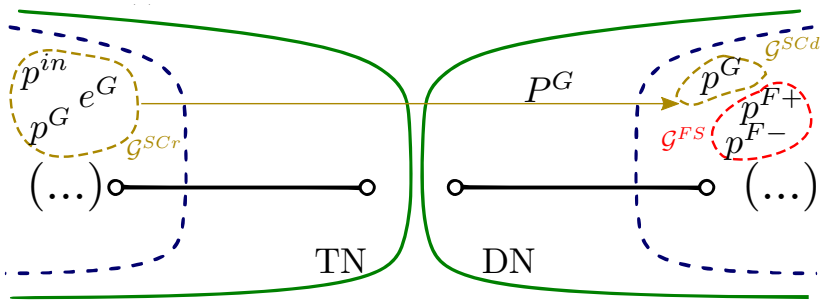
If the master problem has access to the information about the distributed storage power injection bounds, discharge bounds and energy bounds, then it will not need to learn these limits and will be able to propose power injection/discharge values respecting the bounds in the subproblems. Even if these parameters are unknown to the master problem, in Benders, it will learn them through cuts at the expense of iteration. If the energy limits are not known to the master problem, an additional fixing variable will need to cross the interface, and the sensitivity about the proposed energy level by the master to the subproblem needs to be communicated back.

Once the storage is removed from the subproblems and the master problem manages the charge/discharge variables, all the subproblems become time-independent and can be solved as single-period problems.

Figure 5.3 represents the modifications to the regions, where the region on the left-hand side corresponds to the transmission (coordinating) region, and the region on the right-hand side is the distribution (coordinated) region. The stores and their characteristics are taken to the master problem and replaced by a single positive/negative power generator. The combination of two storage variables, p^G and p^{in} (i.e. charge and discharge) into a single parameter P^G, is done to reduce the



(a) Migration of storage variables to the master problem



(b) Replacement of variables in subproblem by a single parameter

Figure 5.3: Temporal decomposition of the subproblems

information passed through the interface. After that, the parameter P^G will fix a single variable p^G in the distribution network side. The separation of p^G and p^{in} happens in the master problem, as it will be shown in subsection 5.3.3.

Analogously to creating flexibility variables on the tie-lines, two flexibility variables are needed for power assignments flexibility in stores (one to allow for positive deviations to the request and the other for negative deviations). The flexibility variables here will keep the problem feasible to requests that would break internal subproblem constraints, such as power limits on lines close to the coordinated store.

The variables and elements are grouped in new subsets, $\mathcal{G}^{SCr}$, $\mathcal{G}^{SCd}$ and $\mathcal{G}^{FS}$,that will be defined later in this section. Figures 5.3a and 5.3b illustrate the process of variable transfer to the master problem and subproblems.

In summary, the process for the temporal decomposition of the distribution networks is as follows:

1. **Detect time-linking elements in distribution.** Check what elements are subjected to time-linking constraints in the distribution network, e.g. energy stores.

2. **Transfer elements to the transmission network.** Pass the information about the elements to the transmission network, creating the necessary variables and constraints.

3. **Replace elements in the distribution network and create sets.** Create the sets with the coordinated and coordinator elements, and add the required coordinated variables in the distribution networks.

5.3.3 Modifications to the model

As explained in previous sections, the original DC model needs to be modified to decompose it spatially and temporally. The model, after the separation, will be solved using Benders decomposition.

Based on the original formulation (Section 5.2), only the changes required to the model will be presented, with a description at the end of the section. It will also be pointed out whether these are new variables and constraints or replace existing ones.

In the following formulation, only one link between transmission and each distribution network is considered (consistent with the test cases presented in Chapter 4); therefore, the coordination of variables θ has been dropped. The relevant notation is found at the start of the Thesis.

Master problem: objective function

$$z^{MP*} = \min \left[\sum_{s\in\mathscr{S}} \frac{W_s}{|\mathscr{T}_s|} \sum_{r\in\mathscr{R}^{TN}} \left(c^{Var}_{rs} + c^{Shed}_{rs} + c^{Curt}_{rs} \right) + \sum_{s\in\mathscr{S}} \sum_{t\in\mathscr{T}_s} \sum_{r\in\mathscr{R}^{DN}} \alpha_{rst} \right] \quad (5.6a)$$

Master problem: constraints

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|} \left[z^{SP*}_{rst} + \lambda^{x\,\mathsf{T}}_{rst} \left(x_{rst} - X^{*i}_{rst} \right) \right] \qquad \forall\, r \in \mathscr{R},\, \forall\, s \in \mathscr{S},\, \forall\, t \in \mathscr{T}_s \quad (5.6b)$$

$$e^{G}_{gs(t+1)} = e^{G}_{gst} + \Delta t \left(\eta_g\, p^{in}_{gst} - p^{G}_{gst} \right) \qquad \forall\, g \in \mathscr{G}^{SCr},\, \forall\, s \in \mathscr{S},\, \forall\, t \in \mathscr{T}_s \quad (5.6c)$$

$$p^{Net}_{gst} = p^{G}_{gst} - p^{in}_{gst} \qquad \forall\, g \in \mathscr{G}^{SCr},\, \forall\, s \in \mathscr{S},\, \forall\, t \in \mathscr{T}_s \quad (5.6d)$$

$$\underline{E}^G_g \leq e^G_{gst} \leq \bar{E}^G_g \qquad \forall\, g \in \mathscr{G}^{SCr}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.6e)$$

$$0 \leq p^G_{gst} \leq \bar{P}^G_g \qquad \forall\, g \in \mathscr{G}^{SCr}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.6f)$$

$$0 \leq p^{in}_{gst} \leq \bar{P}^G_g \qquad \forall\, g \in \mathscr{G}^{SCr}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.6g)$$

Subproblem: additional variables

Additional variables need to be introduced for the flexibility elements in the distribution side. Note that the variables in the problem are also indexed by region, for clarity of notation, these indices are not mentioned in this section.

p^{F+}_{gst}	real power positive flexibility deviation from requests g; $g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, s \in \mathscr{S}, t \in \mathscr{T}_s$
p^{F-}_{gst}	real power negative flexibility deviation from requests g; $g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, s \in \mathscr{S}, t \in \mathscr{T}_s$
x_{rst}	variable concatenation vector, according to the interface in region r (Subsec. 5.3.5)
λ^p_{gst}	dual variable of the variable fixing constraint for real power; $p \in \mathscr{G}^F$
λ^{pS}_{gst}	dual variable of the variable fixing constraint for coord. storage; $p \in \mathscr{G}^{FS}$
λ^D_{pst}	dual variable of the variable fixing constraint for demand profile levels; $p \in \mathscr{P}$
λ^x_{rst}	dual variable concatenation vector, according to the interface in region r (Subsec. 5.3.5)

Subproblem: objective

$$\begin{aligned} z^{SP*}_{rst} = \min \quad & \Bigg[\sum_{g \in \mathscr{G}^T} \left[\beta\, p^G_{gst} \left(C^L_g + \frac{C^{Fuel}_{n(g)} + C^{CO_2} E_{n(g)}}{\eta_g} \right) + C^{Fix}_g \right] \\ & + \sum_{g \in \mathscr{G}^{SCd}} \left[\beta\, p^G_{gst} C^L_g + C^{Fix}_g \right] + \sum_{d \in \mathscr{D}^R} \beta\, p^{Shed}_{dst} C^{Shed}_d + \sum_{g \in \mathscr{G}^R} \beta\, p^{Curt}_{gst} C^{Curt}_g \\ & + \sum_{g \in \left(\mathscr{G}^F \cup \mathscr{G}^{FS}\right)} \beta \left(p^{F+}_{gst} + p^{F-}_{gst} \right) C^{Flex}_g \Bigg] \end{aligned} \quad (5.7a)$$

Subproblem: constraints

$$x_{rst} := X^*_{rst} \quad : \lambda^x_{rst} \qquad \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (5.7b)$$

$$p^{F+}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (5.7c)$$

$$p^{F-}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (5.7d)$$

$$\sum_{g \in \mathscr{G}'_b} p^G_{gst} \quad - \sum_{l \in \mathscr{L}'^{f}_b} p^L_{lst} \quad + \sum_{l \in \mathscr{L}'^{t}_b} p^L_{lst} =$$
$$\sum_{d \in \mathscr{D}^R_b} (p^D_{dst} - p^{Shed}_{dst}) \quad + \sum_{g \in \mathscr{G}^R_b} p^{Curt}_{gst} + \sum_{g \in \mathscr{G}^F_b \cup \mathscr{G}^{SCd}_b} \left(p^{F-}_{gst} - p^{F+}_{gst}\right) + \epsilon^P_{bst}$$
$$\forall b \in \mathscr{B}^A_r, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (5.7e)$$

The first part of the objective function of the master problem (5.6a) is similar to the one in the undecomposed original DC model; the difference is that it only includes the elements contained in the transmission network region (large generators, transmission line limits, etcetera). This first part of the objective function calculates the average hourly price of the elements found in the transmission network. The second part of the function includes the sum over time and regions of the variables α, which are approximations of a single-period distribution network problem.

Constraint (5.6b) corresponds to the Benders optimality cuts, which are linear approximations of the subproblem value surface at the query point. In the constraint, there is a scaling to match the hourly nature of approximating function α in the master problem objective. While the scaling can be done directly in the objective function instead of the cut, the numeric performance was better in the test cases, so the cuts are scaled as shown by the constraint.

Constraint (5.6c) handles the storage level of the coordinated stores. Constraints (5.6e), (5.6f), (5.6g) impose bounds on the coordinated constraints in the master problem. While these constraints could be kept in the distribution region and learned through cuts, the practical decision was to integrate them into the master problem as they are inexpensive constraints that will produce better initial proposals and cuts. In case of privacy conflicts, they can be removed from the master problem and kept in the subproblems; the master problem will learn them eventually.

Finally, for the master problem, constraint (5.6d) ensures that the sign of the requested power is correct, and when passed to the subproblems, it allows to model both the charge and discharge decisions on the coordinated store.

The objective function of the subproblem (5.7a) is a single-period objective,

where costs of thermal generation, load shedding and generation curtailment are considered. The previously existing variables for storage have been replaced by parameters (fixing variable p^G) coming from the master problem. The last part of the objective function contains the flexibility penalties.

In the subproblems, the constraint (5.7b) fixes the request of the master problem by setting it equal to a variable in the subproblem. The auxiliary variables x exist in the subproblem to facilitate the retrieval of the gradient by querying the dual value at the optimal solution of these constraints. The definition of the variable x will depend on the characteristic of the problem and instance; in subsection 5.3.5 the interface for this problem will be specified.

Constraints (5.7c) and (5.7d) ensure the non-negativity of the flexibility variables and constraint (5.7e) is a modified version the load balance constraint presented previously, but considering the additional elements including coordinated stores and flexibility variables.

In Constraint (5.7e), the indices have been modified for brevity, but as also mentioned earlier, the subindices b refer to the subset for elements that are incident to that bus, e.g. $\mathscr{G}'_b$ refers to the augmented subset of generators that are incident to bus b. Subsets $\mathscr{L}'^t_b$ and $\mathscr{L}'^f_b$ appear in the same constraint, and they refer to the subsets of incident "to" and "from" ends to bus b of the augmented set of lines ($\mathscr{L}'_r$ from the notation) in a region.

5.3.4 Cut shareability

The demands in the distribution network can be parametrised in groups, called demand profiles, where a subset of buses will move their demand together proportionally, according to a base load (B^P_d) of a demand incident to a bus and a time-series for the profile D^P_{pst}. Each bus can be associated with an arbitrary number of demands.

As presented in Chapter 4, the profiles are classified as: residential, commercial and industrial. In this approach, each bus can have a combination of residential, commercial and industrial loads, and the total load to which the bus is subjected is the sum of all the incident loads at each period; different distribution networks can

have different base loads and different profiles.

As it is shown in the original DC model, the demands at the buses are set by a profile D^P_{pst} and a base load for the demand B^P_d, which defines the demand at every time-period $P^D_{dst} = B^P_d \, D^P_{m(d)st}$, where m is a mapping function from d to p.

If a cut has the form (5.8a), where x includes only the previously described variables for spatial and temporal decomposition (power at the middle of the tie-line and storage generation), then every cut can only be used in a specific period, for a fixed demand.

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|} \left[z^{SP*}_{rst} + \lambda^{x\,\mathsf{T}}_{rst} \left(x_{rst} - X^{*i}_{rst} \right) \right] \qquad \forall\, r \in \mathscr{R},\, \forall\, s \in \mathscr{S},\, \forall\, t \in \mathscr{T}_s \quad (5.8a)$$

Using the fact that the model uses demand profiles (parametric demands), if then the group of variables x is augmented to include information about the profiles, i.e. profile level and sensitivities, referred here as $\hat{x}$ and dual variables $\hat{\lambda}^x$, the cuts will be shareable in time.

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|} \left[z^{SP*}_{rst} + \lambda^{\hat{x}\,\mathsf{T}}_{rst} \left(\hat{x}_{rst} - \hat{X}^{*i}_{rst} \right) \right] \qquad \forall\, r \in \mathscr{R},\, \forall\, s \in \mathscr{S},\, \forall\, t \in \mathscr{T}_s \quad (5.8b)$$

where $\hat{x}$, $\hat{X}$ and $\lambda^{\hat{x}}$ include the extra elements with information about demand profiles

The demand profiles have the following benefits:

1. The amount of data that has to be specified to define the problem is much less (because the size of P^D_{dst} is vastly bigger than the sum of the sizes of B^P_d and D_{pst}).

2. When sharing cuts, the dimension of the demand part of the cut is just the number of profiles, not the number of demands, resulting in smaller dimensional value functions, which is likely to improve the convergence of the decomposition.

Note that the use of demand profiles does not limit the modelling capabilities; for example, if a particular demand is significant and does not move with the others, then this demand can be attached to a single bus and move with a separate profile; the interface would then need to take into account this additional dimension.

5.3.5 Transmission-distribution interface description

The type of OPF model and the decomposition structure will define the interface between the problems. For the spatial decomposition presented in Section 5.3.1 using the DC OPF approximation, the fixed variables at the interface will only have p^G and θ.

Using Figure 5.2 (Page 50) as a reference, the variables p_i^G and p_j^G represent the values of the power crossing the interface at the midpoint of the tie-line, if:

$$p_i^G = -p_j^G$$
$$\theta_i = \theta_j$$
$$p_j^F = 0$$
$$\theta^F = 0$$

Both the undecomposed model and the decomposition will be equivalent. Furthermore, if there is only one connection between the transmission network and a distribution network, the variable θ is not required for coordination.

Concerning the temporal decomposition of the subproblems, only p^G of the storage is needed, and the interface will contain information of each store being coordinated by the master problem. With the constraints presented in the previous section, the power balance is modified to keep the model consistent.

The cut shareability can be achieved by passing information for each profile acting on each distribution network. The information about the profile level at each period and its gradient need to be communicated.

Other required information that should cross the interface is the true objective value of the subproblem for the sampled point and, to avoid duplicated cuts, the active set at the solution of the subproblem. Figure 5.4 illustrates which information is crossing the interface and in which direction.

The dimension of the cuts for region r is:

$$dim(\alpha_{rst}) = 2|\mathcal{N}_r| + |\mathcal{G}_r^{SCd}| + |\mathcal{P}_r| \qquad \text{(Multi-link)}$$

$$dim(\alpha_{rst}) = 1 + |\mathcal{G}_r^{SCd}| + |\mathcal{P}_r| \qquad \text{(Single link)}$$

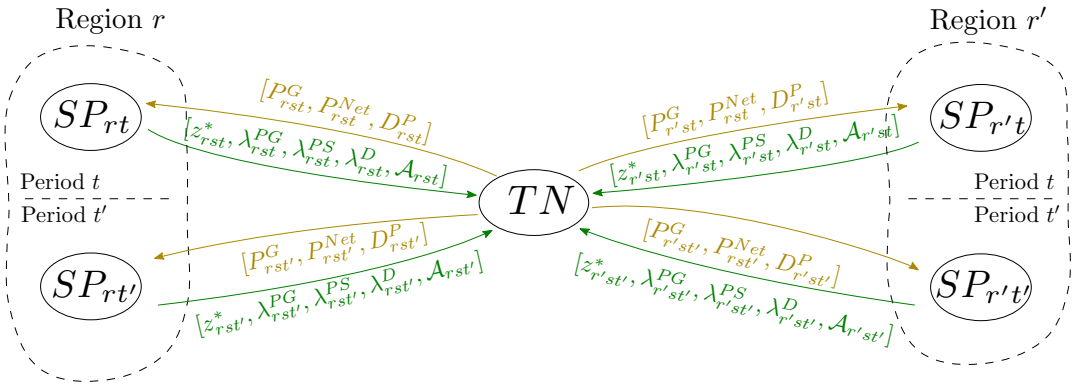


Figure 5.4: Interface for a problem, with 2 DN regions and 2 periods

where $\mathcal{N}_r$ is the set of links between distribution network region r and the transmission network.

The profile level is a fixed parameter in both the master problem and the subproblems, but when the information of the gradient with respect to the level is gathered, it can be used to adjust the cut height.

Figure 5.4 presents an example of a network with a central transmission network and two different single-linked distribution networks. The interface corresponds to the information that must be passed at every iteration of Benders. This figure considers only one coordinated store and one demand profile.

Figure 5.4 also shows that only a single-period subproblem is needed per region; the difference in requests proposed by the master will explore different parts of the same subproblem. Capital letters show that the information passed from the master problem is fixed in the subproblems. In here, P^G, P^{Net} and D^P are the powers fixed at the interface, in/out of the coordinated store and the profile level, respectively.

The information passed from each subproblem to the master problem z^*, λ^{PG}, λ^{PS} and λ^D_{rst} are the optimal objective value of the subproblem, the dual values of the fixed power and the interface and coordinated storage, and the dual with respect to the profile level, respectively. Finally, $\mathscr{A}$ corresponds to the active set of the solution; this piece of information is used to categorise the cuts that will be added to the master problem.

The figure does not explicitly show that the cuts formed from with the proposed and returned information can be shared between periods, however as mentioned earlier, the sensitivity about the profile levels will enable the cut to be adjusted to the

correct height for the given profile level.

5.3.6 Algorithms

Standard Benders Decomposition

In general terms, a standard Benders algorithm applied to a linear problem will be exact and converge to the optimum by adding cuts that improve the definition of the master problem [47].

The following algorithm can be used with the DC-OPF problem, provided the interface has been augmented to avoid infeasibility in the master problem requests to the subproblem. If the feasibility penalties are large enough not to be active in the solution, the problem will converge to the same objective as the undecomposed problem. If the original problem is infeasible, the decomposition could be feasible in some cases. However, the spuriosity of the solution can be observed by a non-zero value in the feasibility variables.

A standard version of the Benders algorithm can be applied directly to the reformulated problem with the changes explained in the previous sections. The methodology used for this problem is presented in Algorithm 5.1.

The first step is to solve the master problem (Line 6), whose objective value corresponds to the lower bound of the problem (Line 7). Line 9 ensures that only points that have not been explored before (set $\mathscr{X}_r$) are explored, skipping duplicated subproblems and saving solving time. If the point has not been seen, then master problem request is fixed in the subproblem (Line 11) and then solved.

When the subproblem is solved, its information is stored in a sample dictionary called $\mathscr{A}_r$. The information kept in the dictionary contains the optimal objective value of the subproblem, point explored, duals of fixing constraints and active set. If the active set a_{rst} has been seen earlier, only the height and points are stored in the corresponding active set group; this information is enough to characterise the cut in every point where the constraint set is active, given the fact that this a linear problem. It is assumed that the gradient is exactly the same at every point in the active set.

Also, Line 15 expands the list of visited points, and if the same point is seen in the future, it will be skipped. When a subproblem is skipped, the historic objective value

at that point is retrieved from the sample dictionary (Line 17).

In Line 21, the absolute and relative gaps are calculated. When the required tightness of the problem is achieved, the algorithm is stopped. On the other hand, if that is not the case, cuts coming from newly explored active sets found at the current iteration are added to the problem at every period.

Unlike Algorithm 6.1, where stabilisation is used, in Algorithm 5.1, stabilisation was not required to achieve fast convergence. When tested, it slowed down the process by increasing the number of Benders iterations for the same convergence and it also slowed down the MP solution by making it a QP (when using the same stabilisation methodology as in the AC OPF-based problems).

The notation for the algorithms is found at the end of the notation section.

Algorithm 5.1 Benders algorithm for the DC OPF spatial and temporal TN-DN decomposition

Separate: the undecomposed problem into regions (Section 5.3).
Map: the links between the master problem and the subproblems.
Define: ϵ^{rel}, $\bar{I}$
Set: $i \leftarrow 1$, $\bar{Z} \leftarrow \infty$, $\underline{Z} \leftarrow -\infty$, $G^{Rel} \leftarrow 1$, $\mathscr{X}_r \leftarrow \emptyset$, $\mathscr{A}_r \leftarrow \emptyset$
while $i \leq \bar{I}$ and $G^{Rel} > \epsilon^{Rel}$ **do**
    Solve: MP_i
    Set: $\underline{Z} \leftarrow z_i^{MP*}$
    Collect: Interface variables values x^*_{rst}
    for all $r \in \mathscr{R}^{\mathscr{D}\mathscr{N}}$, $s \in \mathscr{S}$, $t \in \mathscr{T}_s$ **do**
        if $x^*_{rst} \notin \mathscr{X}_r$ **then**
            Fix: variables at the interface $x_{rst} \leftarrow x^*_{rst}$ in SP_{rst}
            Solve: SP_{rst}
            Classify: active set of the solution depending on its active inequalities
            $a_{rst} \leftarrow$ **call** Alg. 6.3 $\left(\lambda^{I*}_{rst}\right)$
            Collect: $\chi_{rst} \leftarrow [z^{SP*}_{rst}, \lambda^{SP*}_{rst}, x^*_{rst}]$
            Augment: the information on the active set a_{rst}
            $\mathscr{X}_{ra} \leftarrow \mathscr{X}_{ra} \cup \{\chi_{rst}\}$
        else
            Set: $z^{SP*}_{rst} \leftarrow z^{SP*}_{rn}$; where n is the visited repeated point.
        end if
    end for
    Calculate: valid upper bound at the current point and relative gap

$$\hat{G} \leftarrow \sum_{r\in\mathscr{R}^{SP}} \sum_{s\in\mathscr{S}} \sum_{t\in\mathscr{T}_f} \left(\frac{W_s}{|\mathscr{T}_s|} z^{SP*}_{rst} - \alpha^*_{rst} \right)$$
$$\bar{Z} \leftarrow \underline{Z} + \hat{G}$$
$$G^{Rel} \leftarrow \hat{G} / \left(\underline{Z} + \hat{G}\right)$$

    Define: next iteration solution strategy: Primal Simplex, Dual Simplex or Barrier
    Add: Benders optimality cuts for newly detected AS to every period :

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|} \left[z^{SP*}_{ra} + \lambda^*_{ra} \left(x_{rst} - x^*_{ra} \right) \right] \; ; \qquad \forall r \in \mathscr{R}^{SP}, \forall a \in (\mathscr{A}^i_r \cap \mathscr{A}^{i-1}_r), \; \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s$$

    $i \leftarrow i + 1$
end while

Progressive Exploration

Solving the multi-period problem even with decomposition can be expensive. Solving one complete iteration over a set $\mathscr{T}$ of periods (including all the periods in the slices) requires solving one master problem and $|\mathscr{R}||\mathscr{T}|$ subproblems. Even if each subproblem can be solved quickly, the number of subproblems is large, and the information returned by many samples may not be beneficial. In the first iterations, the master problem has a few cuts and does not have a good idea of the behaviour of the subproblems; consequently, it will propose uninformed points to sample for different periods.

When the cuts are shareable in time, there are different options for solving a multi-period problem by exploiting the fact that a cut generated in a period will be valid for other periods (due to the sensitivity of profile levels). The convergence time of a multi-period problem can be reduced by using cuts learned from problems with a subset of those periods.

A progressive temporal exploration is proposed as an efficient and inexpensive method of generating cuts. In the proposed methodology, a set of periods from a multi-period problem $\mathscr{T}$ is divided into subsets of periods that are used for "auxiliary" problems. These auxiliary problems are problems with fewer periods that are solved to gather valid cuts to be used in the target multi-period problem.

Let $\mathscr{E}$ be an exploration strategy, composed of tuples (l, s) representing the length of exploration (number of periods in the auxiliary instance) and number of samples for that length, respectively; ordered by increasing l i.e.

$$\mathscr{E} = \{(l_1, s_1), (l_2, s_2), \ldots, (l_n, s_n)\},$$

where $l_1 \leq l_2 \leq \ldots < l_n$; with $l_n = |\mathscr{T}|$ and $s_n = 1$. Then the strategy consists in creating a set of reduced length auxiliary problems, where l will determine the number of periods in the multi-period problem and s the number of samples for that l.

A clear example of the methodology can be seen in the results shown in Table 5.12 (84), in the table, eight auxiliary problems are solved before solving the 1440-period problem (target problem). There, auxiliary problems 1-5 are single-period problems, auxiliary problems 6 and 7 are 48-period problems, and auxiliary problem 8 is a 336-period problem. The table shows that the number of active sets starts from the

previous problem each time, so problems are solved serially. The exploration strategy for this example is:

$$\mathscr{E}_i = \{(1,1),(1,1),(1,1),(1,1),(1,1),(48,1),(48,1),(336,1),(1440,1)\}$$

The exploration strategy could have been different in a parallel framework if some auxiliary problems had used the same level of information before being solved. In that case, the strategy could have been:

$$\mathscr{E}_i = \{(1,5),(48,2),(336,1),(1440,1)\}$$

Figure 5.5 shows a serial progressive exploration case similar to the one applied in Table 5.12 . In the figure, the process starts at the top left, where horizontal movements correspond to samples of the same problem size l; every sampling instance will potentially detect and augment new active sets, and cuts are transferred to the next auxiliary instance. The rows in the figure represent different lengths of exploration. Algorithm 5.2 describes the progressive temporal exploration.

As it will be shown in results, the progressive exploration process converges faster than a single decomposed $|\mathscr{T}|$-periods problem. If the exploration strategy is chosen to represent patterns, it could help it to converge even faster; for instance, solving a single-period problem would provide information in the form of cuts about the feasibility of the requests at the interface for a fixed storage power charge or discharge. Suppose the next sampling length is a day. In that case, solving the auxiliary problem will likely provide information about daily patterns and how to use storage daily; longer instances will provide information about weekly, monthly and seasonal patterns and storage use.

The periods solved in Table 5.12 were chosen randomly.

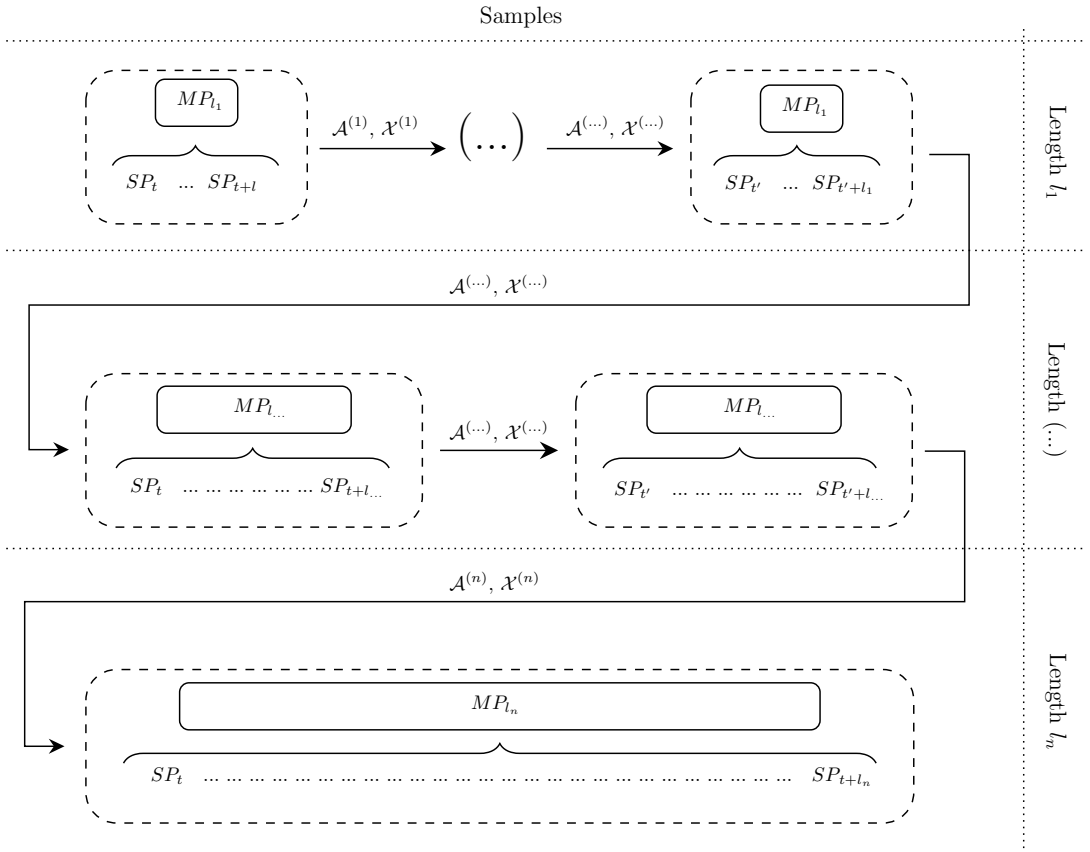


Figure 5.5: (Serial) Progressive temporal exploration

Algorithm 5.2 Progressive temporal exploration

Set: $\mathscr{X}_r \leftarrow \emptyset$, $\mathscr{A}_r \leftarrow \emptyset$
Require: set of pairs $\mathscr{E} := (l, s)$
for all $e \in \mathscr{E}$ **do**
    Form: a MP problem of size e_l
    for all $s \in 1 : e_s$ **do**
        Adjust: the data for MP and SPs for the sampled periods
        Import: $\mathscr{A}_r$, $\mathscr{X}_r$ $\quad \forall r \in \mathscr{R}$
        Solve: auxiliary problem of length e_l using Benders (Alg.5.1)
        Augment: with the newly learned information: $\mathscr{A}_r$, $\mathscr{X}_r$ $\quad \forall r \in \mathscr{R}$
    end for
end for

5.3.7 Generalisation of the interface

The interface presented in Subsection 5.3.5 assumes that there is an explicit representation of the transmission network in the Master Problem. In that interface, one side

of it belongs to the master problem with an explicit representation of the network, and the other side is handled by Benders approximations.

There is no mention of connecting two distribution regions in the interface discussed earlier, which represents an important limitation for general networks. The Master problem needs to handle tie-lines where both sides of the interface are Benders approximations to generalise the interface. Figure 5.6 shows a case where both sides of the linking interface are approximated by Benders.

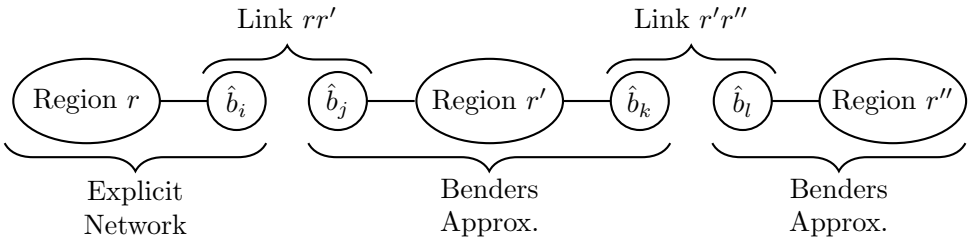


Figure 5.6: Benders approximation on both sides of the interface (Link $r'r''$)

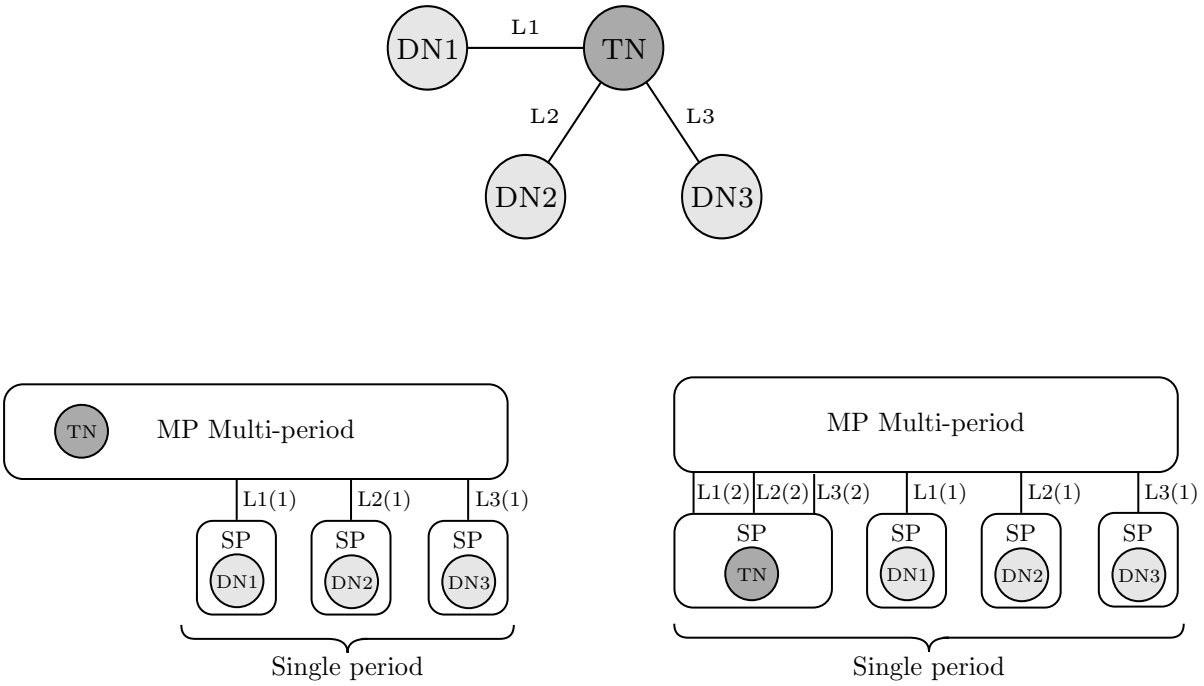
In the Figure, the explicit network (the network from region r) is included in the master problem, and according to the previous formulation, there will be a variable in the master problem that contains the local copy of the auxiliary bus at the middle of the tie-lines in the Link rr'. The situation changes when there is no direct link from the explicit network to other external links, such as the Link $r'r''$. If the master problem wants to fix the decisions on those links, it should include the linking variables of the external tie-lines as well.

The process to generalise the spatial decomposition, in addition to what was previously presented, are:

1. **Detect external tie-lines.** Check which lines are joining two buses in two different regions, and these regions do not contain the explicit network included in the master problem.

2. **Create variables in the master problem.** From the external tie-lines, create the decision variables on both sides of the interface e.g. p^G and θ that will be determined by the cuts which include the gradient in the external links.

3. **Link by constraints both sides of the interface.** ITo modify the models previously presented and help with simple implementation, the two groups of



(a) TN embedded into the MP

(b) No embedded networks in the MP

Figure 5.7: Interfaces for simple links

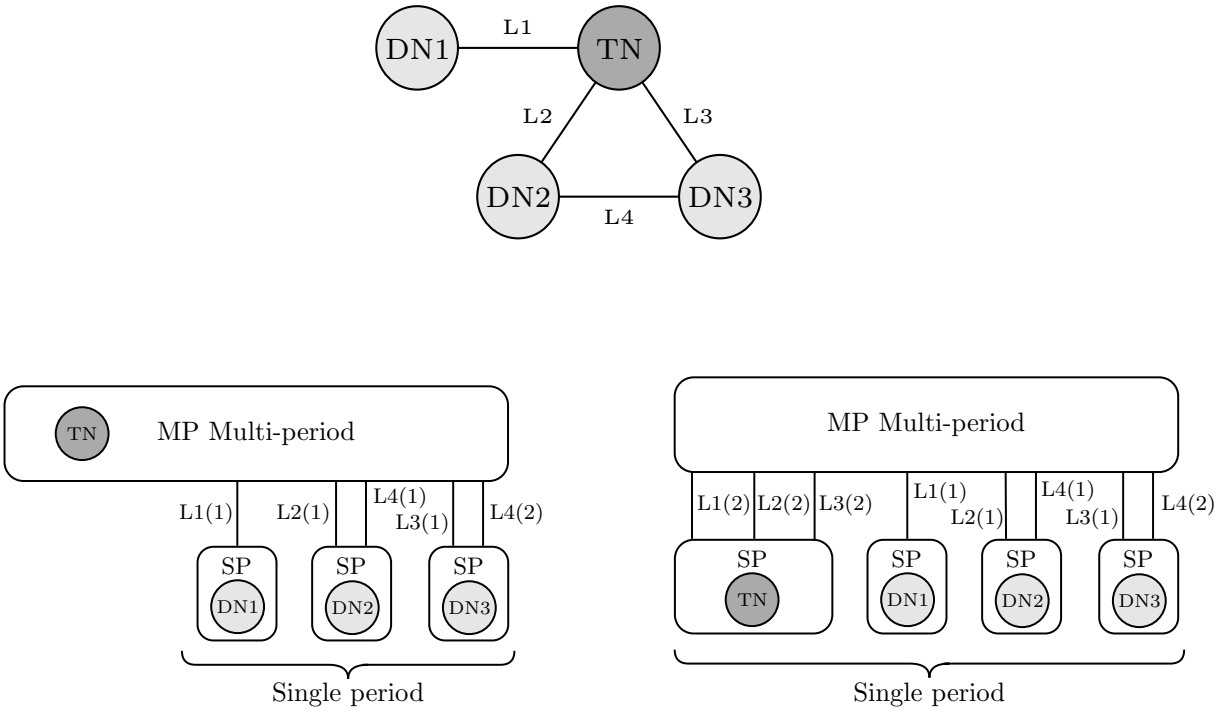
variables will exist on the master problem, and an additional constraint is added to the master to make them equal.

With this process, it is possible to separate the transmission network from the master problem by having Benders approximations on both sides of the tie-line (such as in the Link $r'r''$ from Figure 5.6).

Figure 5.7 illustrates the interfaces required for a simple linked interface, depending on if the transmission network is embedded in the master problem or not. The dimension of the cuts and the number of variables fixed (only for the tie-line part, i.e. not including coordinated storage or profile levels) will depend on the number of links connecting each problem.

In Figure 5.7a there are only three tie-lines (one per subproblem), so each subproblem will be approximated in the master problem with cuts only with one exchange generator variable.

In 5.7b the distribution networks are unaffected, but when the transmission network is not embedded into the master problem, the cuts for it will have higher



(a) TN embedded into the MP

(b) No embedded networks in the MP

Figure 5.8: Interfaces for meshed links

dimension, as they need to include three exchange generator variables.

The numbers in parenthesis next to the tie-line number indicate which models are linked together with equality constraints in the master problem, according to step 3 in the process described in this section. Furthermore, no angle coordination is needed for this decomposition as the bus angles do not need to match in the link.

Figure 5.8 illustrates the interfaces required for a meshed interface, depending on if the Transmission network is embedded in the master problem or not.

An essential difference to the network in Figure 5.7 is that for this case, the bus angle variables need to be included in the interface for all links except for *DN1*.

5.3.8 Temporal decomposition of the transmission network

The temporal decomposition presented earlier included only the decomposition of the distribution networks, as they were represented by Benders approximations in the master and the problem gathered gradients with respect to demand profiles.

The temporal decomposition of the transmission network was not possible in the framework where a central region is explicitly represented as part of the master problem.

By using a "general" interface (Subsection 5.3.7), it is possible to treat the transmission network as another decomposed region, transferring its interface elements to the master problem, which will not contain any explicit network. The transmission network can be solved as a single-period problem instead of a multi-period at the expense of increasing the dimensions of the cuts generated from the transmission network.

In the transmission network, the tie-line power, storage, and demand profiles will be fixed by the master problem. If the transmission network is connected to $|\mathscr{R}|$ regions, the interface will have $n|\mathscr{R}|$ variables in the cuts for the tie-line coordination, where n is the number of variables needed to coordinate, plus all the coordinated stores and demand profiles. Furthermore, large thermal generators in the transmission network are likely constrained by ramp rates; in that case, these variables and time-linking constraints will be in the master problem (and cuts). Without explicit networks in the master problem, the objective function will only contain Benders approximations as shown:

Master problem (no explicit networks): objective function

$$z^{MP*} = \min \sum_{s\in\mathscr{S}} \sum_{t\in\mathscr{T}_s} \sum_{r\in\mathscr{R}} \alpha_{rst}$$

The effectivity of this decomposition will considerably depend on how the network is connected and what constraints and variables are time-linked and must be transferred to the master problem.

5.4 Results

The algorithms here presented had been tested and benchmarked using the test case presented in Chapter 4. In summary, the test case corresponds to a network with 986 buses, 1056 lines and over 150 generators including 58 distributed stores.

Problems with a different number of periods had been solved to show how the

decomposition behaves compared to the original undecomposed model. Furthermore, the methodology has been tested with different price and generation capacities combinations. The two most divergent combinations will be presented and referred to in this section as “Cases”.

In Case A, the generator prices are in a normal range (in line with [59]), and there is a high cost for CO_2; the numerous wind and solar generators are located in various places of the transmission network, and their generation is considered negative load that can be curtailed if needed. Line limits in the transmission network are binding for several periods, but the network can satisfy loads without disconnection (load shedding).

Case B deals with an uncommon state of the network, but it is thought to represent cases that could occur in an exploratory setting, such as an investment planning problem on top of the operation. The generators’ capacities have been reduced, and renewable generators have been replaced by committable inexpensive generation; the line limits in the transmission networks have been reduced. There are periods with high load shedding in various locations in this case.

The models have been coded in Julia 1.5.3 [64] and JuMP 0.21.5 [65] as the programming modelling package, the solver used for this section is Gurobi 9.1 [66] with different optimization algorithms, which will be specified in the results. All the tested instances have the same convergence tolerances and are run in the same computer i7-9700K with 64GB of DDR4-2666 memory.

In this section, the term “Decomposition” refers to the case when the transmission network is part of the master problem, and the term “Generalised decomposition” refers to the case when a master problem contains no explicit network, and the transmission network is decomposed in time.

5.4.1 Problem size

As explained earlier, in terms of variables and constraints, the size of the problem depends on the network: number of buses, lines and generators all multiplied by the number of periods.

Table 5.2 shows the number of constraints, variables and matrix non-zeros in

the problem depending on the number of periods of the case. The table only shows the values for Case A, Case B is very similar because most of the changes are just modifications of cost and coefficients. On average, with the decomposition (keeping the transmission network in the mmaster problem), there is a reduction in the size of at least 12 times for variables and non-zeros, and 13 times in the number of constraints.

In Benders decomposition, the master problem will grow in number of constraints with the addition of cuts each iteration, and the table refers to the size of the master problem at the solution, including all the necessary cuts for convergence.

Table 5.2: Number of elements for Case A, depending on the number of periods (before pre-solve)

Periods	Undecomposed (TN+DNs)			Decomposition (MP at solution)		
	Variables	Constraints	Non-zeros	Variables	Constraints	Non-zeros
1	1.1×10^4	1.8×10^4	3.6×10^4	8.3×10^2	1.5×10^3	3.0×10^3
48	5.3×10^5	8.4×10^5	1.7×10^6	4.0×10^4	7.1×10^4	1.5×10^5
336	3.7×10^6	5.9×10^6	1.2×10^7	2.8×10^5	5.0×10^5	1.1×10^6
1440	1.6×10^7	2.5×10^7	5.2×10^7	1.2×10^6	2.1×10^6	4.6×10^6
2880	3.2×10^7	5.1×10^7	1.0×10^8	2.4×10^6	4.3×10^6	9.4×10^6

An internal pre-solve routine will be called when the model is passed to the solver. Pre-solve transforms the original model m into an equivalent "pre-solved" model p that theoretically has the following properties:

- Problem p is infeasible if and only if m is infeasible.
- Problem p is unbounded if and only if m is unbounded.
- If p has an optimal solution, then its objective optimal value will be identical to the one of m.
- Any feasible (or optimal) solution of p can be mapped to a feasible (or optimal) solution of m.

And the goal of pre-solve is that the pre-solved model is smaller and easier to solve than the original model. When the pre-solve option is activated, the solver will create an "uncrush" map u such as $u : p \mapsto m$, with the objective of mapping the solution of the pre-solved model s' back to the solution of the original model s. And then, it will solve the pre-solved model.

For numerical reasons, the properties above stated are not strictly true, which means that even if the solution for p is optimal, the recovered solution s can be slightly infeasible or suboptimal. Depending on the algorithm, there are additional steps that the solver takes into removing any infeasibility or suboptimality in the solution of model m [66].

Table 5.3: Number of elements for Case A, according to number of periods (after pre-solve)

Periods	Undecomposed (TN+DNs)			Decomposition (MP at solution)		
	Variables	Constraints	Non-zeros	Variables	Constraints	Non-zeros
1	2.9×10^3	8.9×10^2	6.2×10^3	1.8×10^2	1.5×10^2	8.1×10^2
48	1.5×10^5	4.8×10^4	3.2×10^5	1.6×10^4	1.1×10^4	6.7×10^4
336	1.0×10^6	3.4×10^5	2.2×10^6	1.1×10^5	7.6×10^4	4.7×10^5
1440	4.4×10^6	1.4×10^6	9.6×10^6	4.9×10^5	3.3×10^5	2.1×10^6
2880	8.9×10^6	2.9×10^6	1.9×10^7	9.9×10^5	6.9×10^5	4.2×10^6

Pre-solve reduces the size of the undecomposed and decomposed problems; after pre-solve the size differences become smaller, 4.5 times on average for variables and non-zeros in the matrix and 9 times for constraints. The pre-solve routine is more successful in the case of the undecomposed problem where it managed to reduce the size a factor of 3, 17 and 5.5 when comparing before and after pre-solve, for variables, constraints and non-zeros, respectively. In the decomposition, the pre-solve routine manages to reduce the size of the problem only by 2.4, 6.5 and 2.2 times for variables, constraints and non-zeros. The numbers presented in this section refer to the decomposition where the master problem includes the transmission network.

Explaining the difference in pre-solving performance is not straightforward, as these models have too many variables to be explored by hand, and more importantly, many pre-solving rules are kept confidential in the solver software.

The intuition around why this is the case is that the DC lossless problem is relatively simple to solve (at least when compared to the AC OPF models). When the network is not constrained by acting line limits, there is a natural order of preference of generation variables (merit order). By fixing some variables and solving sets of equations, the solver can deduce the values of different variables and eliminate constraints that must be inactive. With the undecomposed model and a clear view of what happens in the distribution networks, the solver can exploit the structure better

in the undecomposed model than in the decomposed one.

However, as explained in earlier paragraphs, even if the pre-solve is supposed to be theoretically equivalent, it is not in practice; numerical implications will affect how feasible or optimal the recovered full solution is. In many cases, this is not significant and can be accepted or dealt with later; there are other cases where the recovered solution's error is unacceptable. An example of this situation is shown in Table 5.6.

5.4.2 Solution time

One of the core objectives of the decomposition is to speed up the solution time of a problem compared to an undecomposed version. This section presents the solution times with different exploration variants, for both Cases (A and B), in a range of problem lengths. The elapsed time reported corresponds only to the time spent by the solver in the optimization routine. The solver will be considered in this section as a black box. Other processes will affect the total elapsed time of solving a problem, but they will be reported in Section 5.4.5.

In this section, three different exploration variants of the decomposition will be presented: a) progressive exploration with cut sharing, b) full model solving with cut sharing, and c) full model without cut sharing. Full model refers to solving by decomposition the target $|\mathcal{T}|$-period without the use of auxiliary problems. Unless otherwise stated, the results mentioned as "Decomposition" will employ the first strategy, "Decomposition" includes the transmission network as a part of the master problem. A different version called "Generalised decomposition" will be tested only with the progressive exploration strategy and, unlike "Decomposition", it will not consider any explicit network as part of the master problem. For benchmarking purposes, each undecomposed instance is also solved by barrier method, barrier method with crossover, primal Simplex and dual Simplex.

As explained earlier, Case A represents a network's normal working condition, where load shedding if present is not significant. The solving times of the algorithms for problems with a different number of periods can be seen in Table 5.4, and the relative time of the algorithms in Figure 5.9.

Table 5.4: Solving time (in seconds) for **Case A** with different methods

Periods		Decomposition		Undecomposed			
		General	TN in MP	Barrier	Barrier, Cr	Pr. Simplex	D. Simplex
1	(Single)	0.07	0.12	**0.02**	0.02	0.03	0.03
48	(Day)	**0.76**	1.25	4.88 [1]	4.08	4.63	6.36
336	(Week)	10.91	**9.65**	48.03 [1]	146.36	157.41	133.95
1440	(Month)	**66.58**	82.58	241.64 [1]	4371.50	2594.07	2086.57
2880	(2 Months)	**117.76**	201.16	627.36 [1]	12 976.38	11 782.39	7255.08
5760	(Season)	**312.46**	600.05	— [2]	— [2]	— [2]	— [2]
8760	(Half year)	**480.14**	1592.50	— [2]	— [2]	— [2]	— [2]

[1] Reported as suboptimal by Gurobi.
[2] It cannot be solved due to memory limitations.

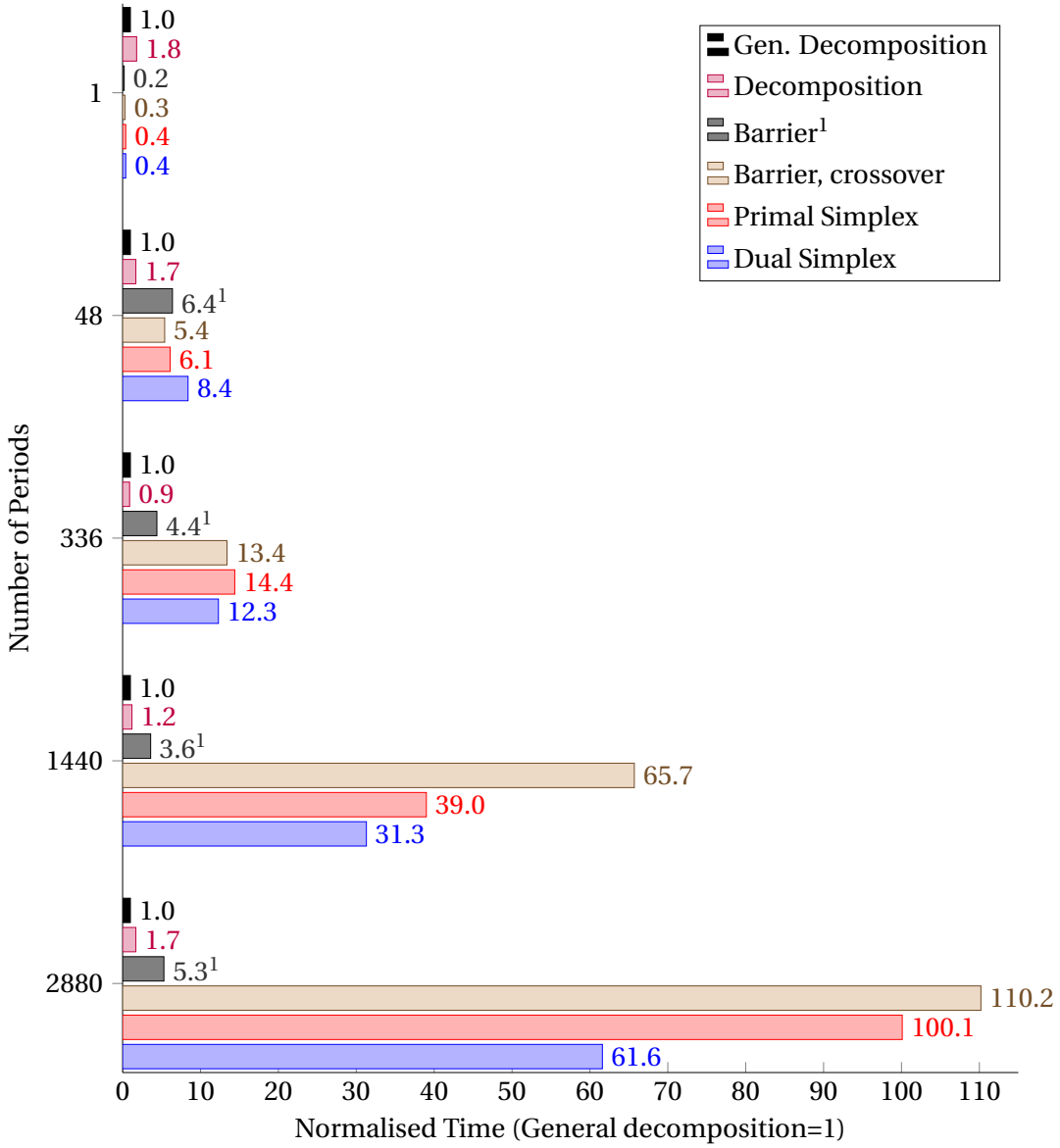


Figure 5.9: Solving time from Table 5.4, normalised time, **Case A**

For Case A and B, solving a single-period problem with decomposition is slower than solving the undecomposed problem; in each iteration, the algorithm needs to solve 29 subproblems (or 30, in the generalised decomposition case) and add cuts from them to the master problem and then moving to the next iteration. In these single-period problems, the temporal decomposition does not apply, and with the problem being easy to solve, we found no benefits over the undecomposed problem.

The multi-period decomposition framework becomes faster than for undecomposed problems for multi-period problems as the number of periods grows. For Case A, even with a relatively small instance (48 periods), the problem can be solved by "Decomposition" at least 3 times faster than the original problem and 1.2 times faster for Case B. Tables 5.4 and 5.7, and Figures 5.9 and 5.10 show the performance of the decompositions against the undecomposed problems for Case A and B.

The fastest algorithm in Case B for the undecomposed version was the Barrier method, but the solver reported suboptimal termination in most instances. Tables 5.5 and 5.6 show the degree of suboptimality in the solutions found by the solver for cases A and B, respectively.

Table 5.5: Suboptimal solutions with the Barrier method, **Case A**

Periods	True optimal objective	Error	
		Abs.	Rel.
48	2 326 397.1	−10.546	−0.0004%
336	4 229 850.6	−14.500	−0.0003%
1440	4 789 043.0	−56.186	−0.0012%
2880	4 981 788.6	−28.222	−0.0005%

Table 5.6: Suboptimal solutions with the Barrier method, **Case B**

Periods	True optimal objective	Error		
		Abs.	Rel.	
48	856 882.3	−0.002	0.00	%
336	1 407 696.7	913.532	0.06	%
1440	1 749 772.4	21 345.146	1.21	%
2880	1 764 183.8	28 309.570	1.58	%

Interestingly, for every problem size, the primal or dual violations were above the required convergence criteria, but the result of these violations has different impacts

on the objective value. While in Case A, the impact is minimal with objective values slightly below the true objective (presumably primal infeasible solution), in Case B the impact is very significant and higher than what would be expected for a linear problem, e.g. in the problem with 2880 periods, the error accounts for 1.58%, which is far above any sensible tolerance for a continuous linear problem.

The intuition behind these divergent results between the cases is related to the mode of operation, i.e. in Case A the network behaves normally without disconnecting loads, while in Case B the load shedding penalties dominate the objective. It could be that the variable value errors are similar for both cases, but their consequences are different because Case B is in an area of high shedding costs.

Unlike the Barrier method that follows an internal path to the solution, the Simplex methods start at a vertex and move along the edges of the polytope to the neighbouring vertices until it reaches optimality. As the problems from this section are linear, the optimal objective will not be found at the interior of the polytope, so Simplex, when converged, will give a highly accurate optimal solution. If a crossover routine is imposed after a Barrier method algorithm has been performed, the result will also have high accuracy. The Crossover consists of three main phases: a primal push phase, where primal variables are pushed to bounds, a dual push phase, where dual variables are pushed to bounds, and a cleanup phase, where Simplex is used to remove any primal or dual infeasibilities that remain after the push phases are complete.

The decompositions perform well when the algorithm starts by solving the master problem with the Barrier method with crossover, and then it is switched to Primal or Dual Simplex when the master problem approximation is close enough to the real objective value. Using Simplex at every iteration (at the finishing stage of the crossover or as a master problem solution method) will guarantee a highly accurate solution that cannot be directly compared to the barrier without crossover. Crossover is in situations, such as Case B, where the potential imprecision of the barrier algorithm has a significant toll on the quality of the result.

The optimal objective values with the decomposition are the same as the results obtained by using Simplex on the original undecomposed problem (up to a relative difference of 1×10^{-15}). According to what has been explained in the last paragraphs

Table 5.7: Solving time (in seconds) for **Case B** with different methods

Periods		Decomposition		Undecomposed			
		General	TN in MP	Barrier	Barrier, Cr	Pr. Simplex	D. Simplex
1	(Single)	0.06	0.12	**0.02**	0.03	0.03	0.03
48	(Day)	1.33	**1.23**	1.44 [1]	3.62	5.72	8.56
336	(Week)	**8.24**	12.79	18.52 [1]	71.56	103.37	145.16
1440	(Month)	**44.91**	56.04	234.76 [1]	992.49	1662.69	2036.07
2880	(2 Months)	**81.06**	123.11	465.86 [1]	3898.94	8201.79	8766.80
5760	(Season)	**186.96**	368.96	— [2]	— [2]	— [2]	— [2]
8760	(Half year)	**291.91**	721.78	— [2]	— [2]	— [2]	— [2]

[1] Reported as suboptimal by Gurobi.
[2] It cannot be solved due to memory limitations.

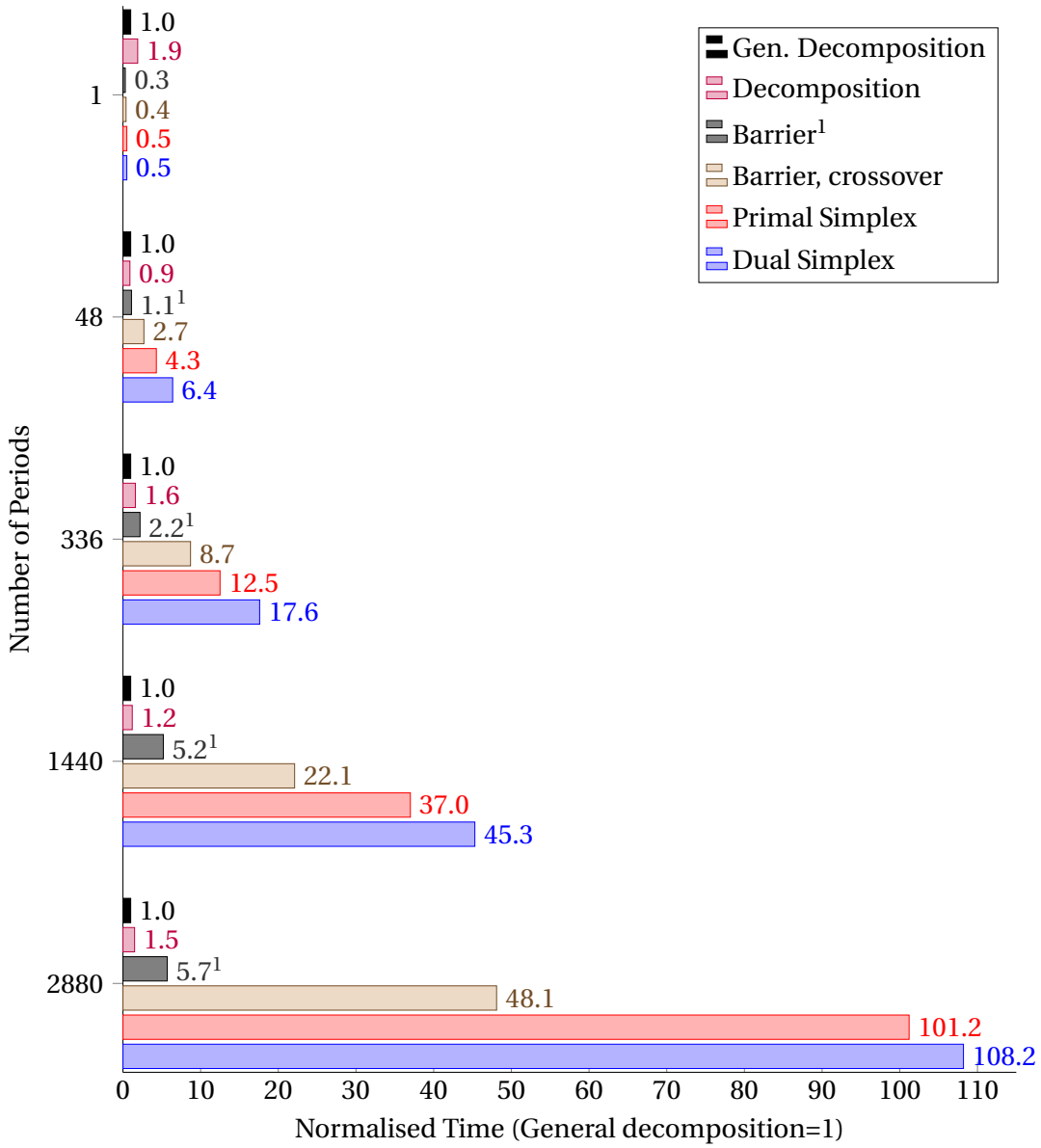


Figure 5.10: Solving time from Table 5.7, normalised time, **Case B**

about the inaccuracy of the barrier method when used alone, the proper comparison to the decomposition is the solving times for the barrier with crossover, primal Simplex and dual Simplex.

The decomposition outperformed all the other methods in every multi-period case. Even if not directly comparable, in large instances, the decompositions outperformed the barrier (suboptimal) method by a factor of 3 to 5 for both Case A and Case B. Compared to the other highly accurate methods, the decomposition shines, solving the 2880-period problem instance 30 to 50 times faster than without decomposition.

The decomposition handles smaller models with fewer variables (in total) than the undecomposed problem, enabling it to solve larger instances. In Tables 5.4 and 5.7 the solving times for 5760 and 8760-period problems are shown for Case A and B, respectively.

In the "Decomposition" (TN in MP), going from a 5760 to a 8760-period problem in Case A had a significant impact on the solution time that did not scale linearly with the problem size, where it did in Case B. The "Generalised decomposition" performed much better, especially with larger instances when the solving time grew linearly with problem size.

5.4.3 Decomposition variants

The results presented earlier (Figures 5.9, 5.10 and Tables 5.4, 5.7), come from one possible variant of the "Decomposition" structure (TN in MP) and "Generalised decomposition" structure (No network in MP), the *Progressive exploration* variant. That is the one that provided the best results in the comparison with the undecomposed model approach.

Two alternative variantes are described in this subsection and compared with the Progressive Exploration variant. The three variants are:

1. **No-cut sharing**. Temporal and spatial decomposition without sharing cuts between periods.

2. **Cut sharing**. Temporal and spatial decomposition with cut sharing.

3. **Progressive exploration**. Temporal and spatial decompostion by progressive exploration (Section 5.3.6).

In the No-cut sharing variant, the active sets explored when sampling a specific period will generate a cut only applicable to the same period. In contrast, the Cut sharing variant uses the sensitivity gathered from the profile levels and adjusts the cut height for different demands; as a result, when a new active set is explored in a period, a cut is generated for every other period, and these are all added to the master problem. The progressive exploration variant is a modification of the Cut sharing variant, which works by exploring a sequence of reduced sized problems to find good cuts inexpensively and then apply them to the full target problem.

Table 5.8: Results by decomposition variant, **Case A**

Periods	No-cut sharing			Cut sharing			Progr. expl.	
	Time (s)	Iter.	SPs	Time (s)	Iter.	SPs	Time (s)	SPs
1	**0.1**	7	**135**	**0.1**	7	**135**	**0.1**	**135**
48	7.5	12	8721	5.0	7	6432	**1.3**	**1656**
336	61.0	13	55 301	34.1	6	39 988	**9.7**	**11 158**
1440	428.3	14	232 138	283.5	7	171 542	**82.6**	**51 831**
2880	1164.6	14	467 444	752.1	7	347 226	**201.2**	**96 064**

Table 5.9: Results by decomposition variant, **Case B**

Periods	No-cut sharing			Cut sharing			Progr. expl.	
	Time (s)	Iter.	SPs	Time (s)	Iter.	SPs	Time (s)	SPs
1	**0.1**	10	**129**	**0.1**	10	**129**	**0.1**	**129**
48	7.9	11	9634	5.9	9	7683	**1.2**	**1797**
336	62.8	12	70 352	37.5	7	49 120	**12.8**	**12 501**
1440	322.4	13	298 649	222.5	7	203 477	**56.0**	**57 303**
2880	979.3	14	606 973	736.6	10	452 708	**123.1**	**102 173**

In Tables 5.8 and 5.9, the summary of results applying different variants is shown for the decomposition with the transmission network in the master problem. When comparing No-cut sharing vs Cut sharing, apart from the single-period case, where they are the same, in every case, the Cut sharing variant converges in fewer iterations and in less time. A reduced number of iterations means solving the master problem fewer times and potentially solving fewer subproblems.

In the presented cases, Cut sharing outperforms the No-cut sharing variant by up to 50% fewer iterations, having solved up to 33% fewer subproblems and speeding up the process by 33%.

In both cases (A and B), the Progressive Exploration variant outperforms the other alternatives by a wide margin. It manages to solve the largest instance much faster, up to 5 times compared to the no-cut sharing and more than 3 times compared to standard cut sharing. In progressive exploration, the number of subproblems solved is also smaller.

In the 2880-period instance of Case A, if a naive implementation of the method is used that does not take into account skipping the already sampled points, it would solve a total of 1 169 280 subproblems, i.e. 29 subproblems for 2880 periods and 14 master problem iterations. By avoiding solving the already sampled points, the number goes down by 60% with respect to the No-cut sharing variant. Instead, by combining the effect of avoiding duplicates, cut sharing and progressive sampling as is done in the Progressive Exploration variant, the number of subproblems solved is only 96 064, which is an effective reduction of 92% when compared to the naive implementation.

Tables 5.10 and 5.11 show the details of the solution for Case A; in the tables, the total solving time can be observed, as well as the time spent in the master problem and subproblems, the gap between lower and upper bound, the number of active sets found. The column called “Alg” refers to the algorithm used to solve the master problem, where *BC* stands for Barrier, with crossover and *PS* means Primal Simplex.

In the tables, there is a significant difference in the number of active sets detected. The No-cut sharing variant explored more active sets of the suproblems. The intuition behind this result, is the fact that with cut sharing the cutting plane representation of the subproblems in the master problem becomes more accurate in earlier iterations as a consequence the master problem visits a less diverse set of points and coverges in a smaller number of iterations. The decrease of the gap for these variants can be seen in Figure 5.11.

In the Progressive Exploration variant, an exploration strategy is required to start the process (Algorithm 5.2). The exploration strategy will define a set of auxiliary problems over shorter time horizons than the target problem, which are much faster

Table 5.10: Decomposition **without** cut sharing, detail for the 1440 period problem, **Case A**

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
1	15.07	2.96	12.11	41 760	45	100.00%	BC
2	19.42	6.27	13.15	41 760	180	100.00%	BC
3	48.21	34.40	13.81	41 760	323	99.99%	BC
4	71.25	59.02	12.23	40 574	345	85.32%	BC
5	63.10	54.50	8.60	29 548	367	93.77%	BC
6	70.12	65.89	4.23	14 203	388	98.58%	BC
7	67.68	64.81	2.86	9659	406	91.27%	BC
8	62.22	59.82	2.41	8114	411	2.92%	BC
9	3.36	2.82	0.53	1595	413	0.86%	PS
10	2.25	1.89	0.37	1123	415	0.04%	PS
11	1.40	1.22	0.17	529	415	0.00%	PS
12	1.50	1.27	0.23	720	415	0.00%	PS
13	1.38	1.23	0.16	493	415	0.00%	PS
14	1.32	1.22	0.10	300	415	0.00%	PS
TOTAL	**428.30**	**357.33**	**70.70**	**232 138**	**415**	**0.00%**	

Table 5.11: Decomposition **with** cut sharing, detail for the 1440 period problem, **Case A**

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
1	14.73	2.88	11.85	41 760	45	100.00%	BC
2	42.16	28.85	13.31	41 760	184	100.00%	BC
3	40.14	28.65	11.49	41 203	211	93.92%	BC
4	79.89	71.18	8.71	31 232	233	97.31%	BC
5	101.50	97.62	3.88	13 713	253	0.50%	BC
6	3.65	3.15	0.50	1512	256	0.00%	PS
7	1.43	1.31	0.12	362	256	0.00%	PS
TOTAL	**283.50**	**233.63**	**49.86**	**171 542**	**256**	**0.00%**	

to solve, to generate cuts that will be useful when solving over larger time horizons of the target problem.

The detailed process of solving a 1440-period example with the progressive exploration variant is shown in detail in Table 5.12. An analogous table with the results of Case B can be seen in the Appendix C.

In Table 5.12, the exploration strategy is $L_e = [1,1,1,1,1,48,48,36,1440]$ with $S_e = [1,1,1,1,1,1,1,1,1]$, which can be read as solving serially 5 randomly selected 1-period problems, 2 randomly selected 48-period problems, 1 problem for 336 periods and the target period range of $1:1440$. In the smaller time-horizon problems, some

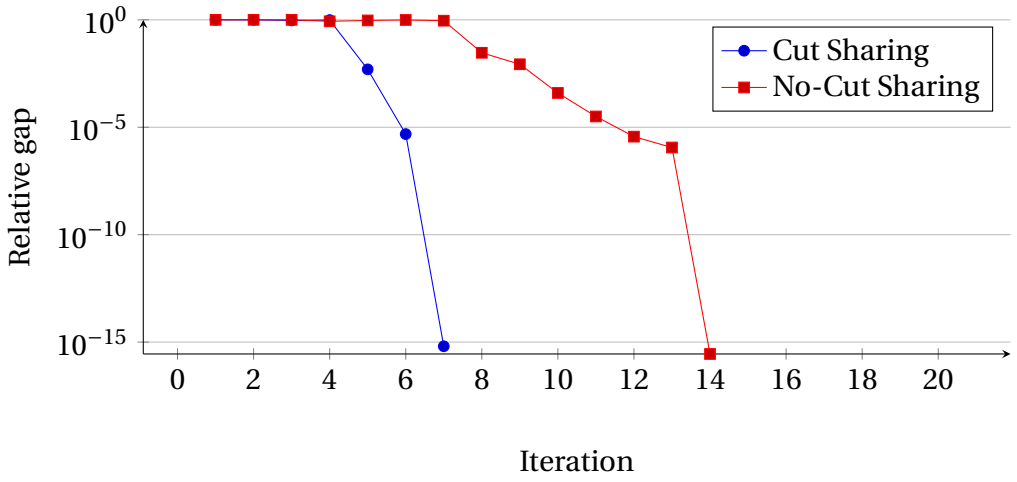


Figure 5.11: Relative gap comparison with and without cut sharing, 1440 periods, **Case A**

general features can be learned, such as the feasibility limits of the requests and in the 48-period problems how to use storage for a daily pattern, etcetera.

In this example, the first period in each auxiliary problem is chosen at random. However, given that the sampling aims to learn the usage patterns from the auxiliary problems, this is not ideal. If relevant information about load profiles is known, it can be used to sample, for instance, a weekday and a weekend day, hoping to learn the different intra-day storage management patterns for these days.

Selecting the start of the ranges of the auxiliary problems at random is fast and straightforward but sometimes leads to poor choices and wasted effort; an example of this can be seen in Table 5.12, where the auxiliary problems 6 and 7 sampled a very similar range (the range explored is shown in brackets); this made Auxiliary Problem 7 redundant, and as it can be seen from the number of visited active sets, where no new information was learned, but it still took 1.83 seconds to explore the problem.

The power of the Progressive Exploration variant is readily observable by looking at the number of detected active sets found by each problem. For the solution of Case A, with the decomposition (including TN in MP), only 123 different active sets in total were detected, and 99 of them were found in 0.11 seconds by Auxiliary Problem 1. In Case B (Appendix), this fact is still present, but there is a clear need for more extended period exploration, as 116 active sets were found by auxiliary problem 1, but this was out of a total of 240.

Table 5.12: **Progressive exploration**, Decomposition (TN in MP), **Case A**, 1440 periods

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
***Auxiliary Problem 1**, 1-period [1384]*							
1	0.02	0.00	0.02	29	29	100.00%	BC
2	0.02	0.00	0.02	29	56	100.00%	BC
3	0.02	0.00	0.02	29	80	98.11%	BC
4	0.02	0.00	0.01	28	93	99.01%	BC
5	0.01	0.00	0.01	18	96	29.85%	BC
6	0.01	0.00	0.00	9	97	12.99%	BC
7	0.01	0.00	0.00	12	99	0.19%	BC
8	0.00	0.00	0.00	5	99	0.00%	PS
Subtotal	*0.11*	*0.02*	*0.09*	*159*			
***Auxiliary Problem 2**, 1-period [1324]*							
1	0.01	0.00	0.01	29	101	3.88%	BC
2	0.01	0.00	0.00	3	102	19.03%	BC
3	0.00	0.00	0.00	1	103	12.14%	BC
4	0.00	0.00	0.00	1	104	1.30%	BC
5	0.00	0.00	0.00	1	104	0.00%	PS
Subtotal	*0.03*	*0.02*	*0.01*	*35*			
***Auxiliary Problem 3**, 1-period [816]*							
1	0.01	0.00	0.01	29	104	0.00%	BC
Subtotal	*0.01*	*0.00*	*0.01*	*29*			
***Auxiliary Problem 4**, 1-period [416]*							
1	0.01	0.00	0.01	29	105	0.58%	BC
2	0.00	0.00	0.00	1	105	0.00%	PS
Subtotal	*0.01*	*0.00*	*0.01*	*30*			
***Auxiliary Problem 5**, 1-period [360]*							
1	0.01	0.00	0.01	29	105	0.00%	BC
Subtotal	*0.01*	*0.00*	*0.01*	*29*			
***Auxiliary Problem 6**, 48-period [610:657]*							
1	0.93	0.55	0.39	1385	112	0.03%	BC
2	0.04	0.03	0.01	43	115	0.00%	PS
3	0.03	0.03	0.01	18	116	0.00%	PS
Subtotal	*1.01*	*0.60*	*0.40*	*1446*			
***Auxiliary Problem 7**, 48-period [618:665]*							
1	1.83	1.67	0.16	558	116	0.00%	BC
Subtotal	*1.83*	*1.67*	*0.16*	*558*			
***Auxiliary Problem 8**, 336-period [1356:251]*							
1	8.34	5.57	2.77	9734	121	0.04%	BC
2	0.31	0.25	0.05	167	121	0.00%	PS
Subtotal	*8.65*	*5.83*	*2.82*	*9901*			
***Target Problem**, 1440-period [1:1440]*							
1	69.38	58.35	11.03	39 008	123	0.00%	BC
2	1.55	1.34	0.21	636	123	0.00%	PS
Subtotal	*70.93*	*59.69*	*11.23*	*39 644*			
TOTAL	**82.58**	**67.79**	**14.79**	**51 831**	**123**	**0.00%**	

Similar to Table 5.12, Table 5.13 shows the progress of the exploration method (with the same exploration periods selected) but using the generalised decomposition.

5.4.4 Re-use of cuts

The results in the previous subsection show that the progressive exploration variant solves a new problem several times faster than the other two variants considered. In these examples, problems were solved without using prior historical information.

An additional benefit of cut shareability is that repetitive problems, where only the demands are changing, can use the information gathered in earlier problems using information stored in sample dictionaries. Then instead of building a progressive exploration from scratch, the cuts can be directly imported from the sample dictionary without auxiliary problems, and the target problem can be solved even faster. This situation could arise from solving similar problems daily, which is usual with an operational problem.

Even if a subset of subproblems changes its characteristics, e.g. new generators are added, new lines are built, or generation costs change, the cuts from the other problems that were not modified are still valid and can be used to help solve the new problem.

Table 5.14 shows the solving time for 1440-period problems with and without sample dictionaries. Instance “a” (Periods 1:1440) is solved by Progressive Exploration without any dictionary of samples, but Instances “b”, “c” and “d” are solved using sample dictionaries. If a good dictionary of samples is available, there is no need to do progressive exploration, and the target problem can be solved directly. In the table, the column “Imp. Cuts.” refers to the starting dictionary or dictionaries used to solve the problem.

The decomposition becomes even more powerful when the successive problems have numerous periods in common, such as problems with a rolling horizon structure, where a problem with the next 30 days is solved every day.

The next time the problem is solved in a rolling horizon framework, the immediate periods have occurred, and storage levels or generator set-points may have

Table 5.13: **Progressive exploration**, General decomposition, **Case A**, 1440 periods

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
***Auxiliary Problem 1**, 1-period [1384]*							
1	0.02	0.00	0.02	30	30	100.00%	BC
2	0.02	0.00	0.02	30	60	100.00%	BC
3	0.01	0.00	0.01	30	89	18.20%	BC
4	0.01	0.00	0.01	29	90	0.69%	BC
5	0.00	0.00	0.00	2	90	0.00%	PS
Subtotal	*0.07*	*0.00*	*0.06*	*121*			
***Auxiliary Problem 2**, 1-period [1324]*							
1	0.01	0.00	0.01	30	93	10.36%	BC
2	0.00	0.00	0.00	5	94	19.03%	BC
3	0.00	0.00	0.00	2	95	12.14%	BC
4	0.00	0.00	0.00	2	96	1.30%	BC
5	0.00	0.00	0.00	2	96	0.00%	PS
Subtotal	*0.02*	*0.01*	*0.01*	*41*			
***Auxiliary Problem 3**, 1-period [816]*							
1	0.01	0.00	0.01	30	96	0.00%	BC
Subtotal	*0.01*	*0.00*	*0.01*	*30*			
***Auxiliary Problem 4**, 1-period [416]*							
1	0.01	0.00	0.01	30	97	0.58%	BC
2	0.00	0.00	0.00	2	97	0.00%	PS
Subtotal	*0.01*	*0.00*	*0.01*	*32*			
***Auxiliary Problem 5**, 1-period [360]*							
1	0.01	0.00	0.01	30	97	0.00%	BC
Subtotal	*0.01*	*0.00*	*0.01*	*30*			
***Auxiliary Problem 6**, 48-period [610:657]*							
1	0.66	0.25	0.41	1440	107	0.19%	BC
2	0.07	0.02	0.04	130	111	0.03%	PS
3	0.02	0.01	0.01	20	113	0.00%	PS
4	0.02	0.02	0.00	6	113	0.00%	PS
Subtotal	*0.77*	*0.30*	*0.47*	*1596*			
***Auxiliary Problem 7**, 48-period [618:665]*							
1	0.61	0.42	0.19	631	114	0.04%	BC
2	0.12	0.07	0.05	139	117	0.01%	PS
3	0.08	0.05	0.04	103	117	0.00%	PS
Subtotal	*0.81*	*0.53*	*0.27*	*873*			
***Auxiliary Problem 8**, 336-period [1356:251]*							
1	5.55	2.71	2.85	10 080	124	0.06%	BC
2	0.99	0.80	0.19	539	126	0.01%	PS
3	0.25	0.14	0.12	289	126	0.00%	PS
Subtotal	*6.80*	*3.64*	*3.16*	*10 908*			
***Target Problem**, 1440-period [1:1440]*							
1	49.71	38.26	11.45	40 303	138	0.01%	BC
2	6.34	5.83	0.51	1411	142	0.00%	PS
3	1.23	0.73	0.50	1216	143	0.00%	PS
4	0.81	0.74	0.07	177	143	0.00%	PS
Subtotal	*58.09*	*45.55*	*12.53*	*43 107*			
TOTAL	**66.58**	**50.05**	**16.54**	**56 738**	**143**	**0.00%**	

Table 5.14: Re-use of 1440-period problem cuts

Probl.	Periods	Imp. Cuts	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets
a	1:1440	None	66.58	50.05	16.54	56 738	143
b	1441:2880	a	48.59	34.98	13.61	44 724	144
c	2881:4320	a,b	50.55	37.77	12.78	43 200	144
d	4321:5760	a,b,c	56.20	43.21	12.99	44 068	145

changed; many of the distant periods could remain static. The benefit of this structure is that the master problem is likely to propose subproblem points that have already been explored before, especially for the distant horizon, so the number of subproblems to solve will be reduced with it, the solving time.

Three consecutive problems were solved to show the consequences of a rolling horizon structure: first a problem for days 1 to 30 (periods 1:1440), then days 2 to 31 and finally days 3 to 32. The problems were solved by barrier with crossover, and the cuts were imported in the following problem. The new master problem suggested repeated points and managed to reduce the number of subproblem points explored by nearly 65%, which is a relatively small decrease given that the demands in 29 out of 30 days were not modified in the following problem. A reason for this is that our implementation cannot use and modify the optimal basis found in the previous problem when solving the following problem because, in the code implementation, a new model object (in the programming language) is created with each problem.

Another example that would greatly benefit from the decomposition is an operational problem that has been solved, and then a new, more precise short term forecast for demands and renewable generation has been made available for the same time interval. With the existing cuts and existing exploration points, the master problem is likely to suggest already visited points, reducing the number of subproblems to be solved in the new slightly modified problem. In this case, the benefit is not only to reduce the number of subproblems solved, but if the optimal basis of the previous master problem was stored, this slightly modified version (short term forecast updates) is likely solved in a low number of Simplex iterations, accelerating the process even more.

Table 5.15 shows the results of solving this situation. In the example, first, the 1440-period problem from Case A is solved with the generalised decomposition

and progressive exploration (see Table 5.13), and then when new information about the renewable generation and load profiles is available, the first 48 periods (the immediate day after the present period) are modified. A random perturbation for every period up to ±10% is introduced in every demand profile and renewable generation level.

Table 5.15: 48-period perturbations in a 1440-period problem, Case A

Problem	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Alg.
Generalised decomposition						
Original	66.58	50.05	16.54	56 738	143	Various
Perturbed	7.22	6.51	0.71	2256	143	Pr. Simplex
Undecomposed problem						
Original	2086.57	-	-	-	-	Dual Simplex
Perturbed	243.08	-	-	-	-	Pr. Simplex

As the perturbation induced was minor and the previous basis stored, a primal Simplex method can be highly effective in solving the master problem in the new perturbed problem. As it can be seen in Table 5.15, it only took 7.22 seconds to find the optimal solution and to evaluate the new proposed subproblem points. Internally, only 2535 primal Simplex iterations were performed, and the master problem needed additional information at only 2256 points in the subproblems. After the perturbation, it would be needed to solve at least a further 1440 subproblem points (30 subproblems, 48 periods). However, due to the suboptimality of the starting solution, additional 816 points needed to be evaluated, the other 40944 points from the unperturbed days (30 subproblems, 1440 periods minus 816) were previously seen.

Note that the benefit of storing the master problem basis in the framework of this example is not exclusive to the decomposition, and a similar process could be implemented in the undecomposed version. However, the basis size is considerably larger than in the decomposition, and the expense in evaluations and Simplex iterations is considerably higher. As it can be seen in Table 5.15, the undecomposed model required 2203 primal Simplex iterations and took 243.08 seconds, which is more than 30 times as long as the decomposed solve.

5.4.5 Model building time and memory usage

Another benefit of the decompositions is their ability to handle larger models. As it is shown for this instance in Subsection 5.4.1, the size of the decomposed model can be significantly smaller than the one in the undecomposed case. Fewer variables and constraints, assuming the same sparsity, mean that less memory is needed. While there are different methods of building a model, in JuMP at least 2 full copies of the model need to be kept (of the non pre-solved version), one internally in JuMP and one communicated to the solver. The solver itself will need additional space for the pre-solved model, the mapping functions between this and the non pre-solved version and additional memory to perform optimization computations.

The memory usage and building time are heavily affected by the modelling language, the model building routines, coding style, and the solver options. In the results presented in this section, the same language, modelling routines and coding style are applied to both the undecomposed and decomposed models. In Table 5.16 and Figure 5.12, a speed-up factor of more than 20 times is achieved with the decomposition (TN in MP) for larger instances. In terms of model building, the fact that there are smaller groups of elements in the decomposition (regions) has a significant benefit when constraints are built, as the size of the tables that link the variables are much smaller.

A group of supporting subsets was created to avoid extensive conditional table searches, e.g. checking if a line is connected to a bus when building a power balance constraint. The supporting subsets are fast to generate and only need to be generated once instead of having a condition in a constraint building routine that repeats the check for every period in the problem. The results are shown in Table 5.16 and already include supporting elements to speed up the model building process.

After the model is built, it is passed to the solver, where large instances will take time to build the problem internally; the time spent by the solver creating the model is reported as “Launching”. In the case of the decompositions, additional routines consume time; the main additional routines are cut building and duplicated point checks. The cut building routine puts together the information about the new cuts and modifies the model object by adding the necessary constraints. The duplicated

point check ensures that the same problem is not solved more than once.

Table 5.16: Model building and solver launching time (in seconds) for **Case A**, Decomposition vs. Undecomposed

Periods		Undecomposed			Decomposition			
		Building	Launching	Total	Building	Launching	Other[1]	Total
1	(Single)	0.0	0.1	**0.1**	0.2	0.1	0.0	0.4
48	(Day)	3.4	1.5	4.9	0.4	3.1	0.1	**3.6**
336	(Week)	52.3	40.0	92.3	2.5	20.3	0.8	**23.6**
1440	(Month)	378.7	591.8	970.5	11.6	74.0	5.5	**91.0**
2880	(2 Months)	2183.5	2218.0	4401.5	31.7	138.3	14.8	**184.9**

[1] Includes cut addition routine and repeated points check.

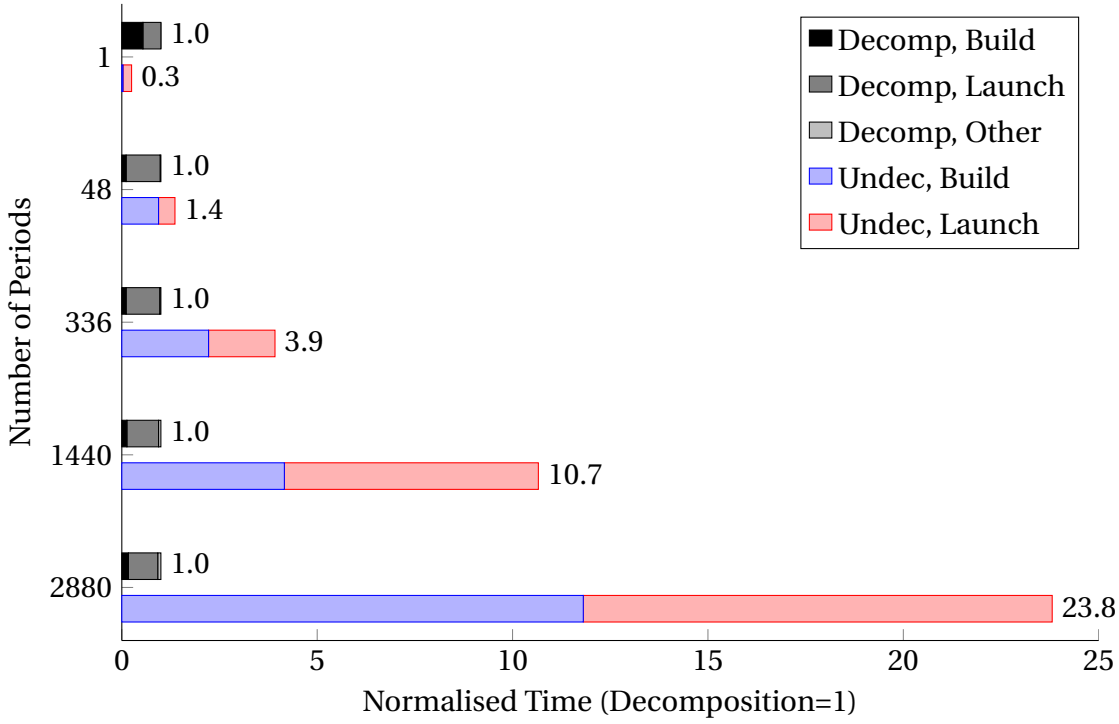


Figure 5.12: Model building and solver launching time from Table 5.16, normalised (Decomposition=1), **Case A**

In small models, the decomposition has a considerable time penalty because it needs to build $|\mathscr{R}|$ models (one per distribution region plus the transmission network), but when there are more periods, the building time proportionally stops being significant. Most of the non-solving time in the decomposition is spent in the Launch phase. Even if most of the subproblems are practically instantaneous to solve (0.0002s), launching the solver consumes five times longer (0.001s), with the fact that

there are at most $|\mathscr{R}||\mathscr{T}|$ subproblems to solve per iteration, this penalty becomes significant.

With the undecomposed model, the time spent in non-solving operations is balanced between model building and solver launching. As noted earlier, the non-solving times are not included in the algorithm time, as they are heavily language and code dependent, and the solver is considered a black box in this thesis.

Another benefit, that has not been tested here, is the ability to parallelise the solution process for the subproblems; in the examples presented in this section, there would be a small benefit in a parallel implementation, as most of the time is spent in the master problem.

5.5 Discussion

At the beginning of the chapter, the proposed methodology had the objective of having a more detailed representation of the distribution networks that could be optimized together with the transmission network in a fast and private way and that was able to cope with large scale problems.

The decompositions here presented successfully achieve the goal by solving the problems up to 50 times faster than the undecomposed problem with high accuracy (See Fig. 5.10, Case B, 2880-periods). They are faster, whether they are solving a new problem or a problem with modifications, by generating, using and re-using the information learned in the previous explorations. The decompositions are also robust; when a distribution network installs a new technology, the rest of the problem will still be valid, and the cuts can be used to solve the problem efficiently.

The decompositions are memory friendly; the reduced number of elements decreases the amount of memory needed by the models and allows them to solver larger instances. Due to the same reason, the models are also faster to build and launch. The decompositions are also general (in the generalised case), provided that each bus is assigned to a solution region; they can be split arbitrarily, with the problem and model changes presented in this chapter.

Even if privacy is not achieved in a strict sense (the coordinating agent could reverse-engineer some of the features of the regional networks), the amount of com-

municated information is limited, especially in situations where no explicit network is contained in the master problem. In this framework, a third-party operator without detailed knowledge of any of the networks could act as the manager of the master problem and allow only to receive and propose points and solutions. Depending on the level of information desired to be communicated, explicit constraints in the master problem could be left intact in the subproblems but still learned by the master problem at the expense of additional iterations.

The decomposition would also have advantages when used in stochastic investment planning problems, for instance, in cases where an uncertain price of uranium or gas can impact whether to build or not a nuclear plant. In a pre-existing network, those regions where the investment is not considered are not affected by the price uncertainty, and their cuts and information gathered can be valid for different nodes in the tree.

In summary, the proposed methodology is faster, more resilient and limits the explicit information needed of other regions when compared to the undecomposed version, and it needs to be tested further to better understand its behaviour , expanding its use, capabilities and power; opening the way for further exciting developments.

Chapter 6

AC OPF-Based Problem

6.1 Motivation

This chapter deals with a similar motivation of that of Chapter 5 but in a full AC OPF framework. A full AC OPF model takes into account the true physics of the system, including voltage limits, reactive power and line losses, and they are commonly used for modelling distribution networks where many of these issues may be binding.

The move from a linear power flow model (DC-OPF) to a general non-convex model presents huge challenges of solution complexity and solution time. Even a general local solver is several times slower to converge (if it even manages to converge) than a mature linear solver for an equivalent LP problem. This fact, coupled with the natural growth of the model size, with the addition of reactive power variables, voltage magnitude variables and other network or bus parameters, makes it a challenge to solve large networks where a feasible real problem solution is needed.

The methodology presented in this chapter aims to tackle the following challenges in a full AC-OPF framework:

1. Have a detailed AC-OPF model for both transmission and distribution networks, optimizing over variables on both levels.
2. Get a close-to-optimal feasible solution.
3. Reduce the amount of data exchanged between parts, preserving privacy.
4. Allow connecting different model types easily.

The methodology here presented addresses these objectives by exploiting the structure of the problem in regions and multi-period frameworks, by building cutting plane approximations to the value surfaces of the subproblems that can be adjusted to new circumstances (new power demand levels, for example). The methodology is similar to the one presented in Chapter 5 (for the linear case) but has a fundamental difference in the sense that the actual value surfaces may be non-convex. So cuts cannot be assumed to be globally valid; the consequence is that they need to be updated and validated for the convergence point.

The models presented in this section are non-convex. As such, they can have multiple optima as shown in [67]. The multiple local optima can arise from problems with disconnected feasible regions, loop flow, excess of real or reactive power, negative costs, and significant voltage angle differences.

The interest in the proposed methodology is to find near-optimal feasible solutions. Whether the test cases have local optima is not of particular interest, but as seen later in the results section, all the decompositions converged to the same objective value (to the specified tolerance) as the undecomposed problem. The decompositions never found a different optimum, which does not necessarily rule out other optima and does not mean that the decomposition's generation schedule solution is close to the undecomposed solution, as the problem may have flat areas or have degeneracy.

Unlike practical GBD approaches mentioned in the introduction, the non-convex subproblems solved in this chapter have not been altered (apart from feasibility considerations) and they are solved as they are presented. The proposed methodology while does not ensure optimality is used as a heuristic to provide fast and good feasible solutions.

The models that will be solved are presented in the following section for both an undecomposed formulation and the decomposition, with separate transmission network and distribution network models.

6.2 Undecomposed formulation

This section describes the undecomposed optimization model, with an AC-OPF power flow model (Polar PQV in Chapter 2), that will be used as the benchmark and the starting point which will be then decomposed into a multi-period transmission network and several single-period distribution network problems. The model i.e. constraints and objective are presented.

Constraints

$$\underline{V}_b \leq v_{bst} \leq \bar{V}_b \qquad \forall b \in \mathscr{B}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1a)$$

$$-2\pi \leq \theta_{bst} \leq 2\pi \qquad \forall b \in \mathscr{B}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1b)$$

$$\underline{P}^G_g \leq p^G_{gst} \leq \bar{P}^G_g \qquad \forall g \in \mathscr{G}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1c)$$

$$\underline{Q}^G_g \leq q^G_{gst} \leq \bar{Q}^G_g \qquad \forall g \in \mathscr{G}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1d)$$

$$0 \leq p^{Curt}_{gst} \leq p^G_{gst} \qquad \forall g \in \mathscr{G}^R, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1e)$$

$$0 \leq p^{in}_{gst} \leq \bar{P}^G_g \qquad \forall g \in \mathscr{G}^S, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1f)$$

$$0 \leq e^G_{gst} \leq \bar{E}^G_g \qquad \forall g \in \mathscr{G}^S, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1g)$$

$$0 \leq l^{Shed}_{pst} \leq 1 \qquad \forall p \in \mathscr{P}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.1h)$$

$$\left(p^{L^f}_{lst}\right)^2 + \left(q^{L^f}_{lst}\right)^2 \leq \bar{S}^2_l \qquad \forall l \in \mathscr{L}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.2a)$$

$$\left(p^{L^t}_{lst}\right)^2 + \left(q^{L^t}_{lst}\right)^2 \leq \bar{S}^2_l \qquad \forall l \in \mathscr{L}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.2b)$$

$$\theta_{bst} = 0 \qquad b \in \mathscr{B}^{ref}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.3a)$$

$$e^G_{gs(t+1)} = e^G_{gst} + \Delta t \left(\eta_g \, p^{in}_{gst} - p^G_{gst}\right) \qquad \forall g \in \mathscr{G}^S, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.3b)$$

$$p^G_{gst} = D_{o(g)st} \qquad o : g \mapsto p,\ g \in \mathscr{G}^R, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.3c)$$

$$\sum_{g \in \mathscr{G}_b} p^G_{gst} - \sum_{l \in \mathscr{L}^f_b} p^{L_f}_{lst} - \sum_{l \in \mathscr{L}^t_b} p^{L_t}_{lst} =$$
$$\sum_{d \in \mathscr{D}^R_b} (P^D_{dst} - B_d D^P_{m(d)st} l^{Shed}_{m(d)st}) + \sum_{g \in \mathscr{G}^R_b} p^{Curt}_{gst} + \sum_{g \in \mathscr{G}^S_b} p^{in}_{gst} + v^2_{bst} G^B_b$$

$$\forall\, b \in \mathscr{B},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3d)$$

$$\sum_{g \in \mathscr{G}_b} q^{G}_{gst} \quad - \sum_{l \in \mathscr{L}^{f}_{b}} q^{L_f}_{lst} \quad + \sum_{l \in \mathscr{L}^{t}_{b}} q^{L_f}_{lst} =$$
$$\sum_{d \in \mathscr{D}^{R}_{b}} (Q^{D}_{dst} - B_d D^{Q}_{m(d)st} l^{Shed}_{m(d)st}) \quad - v^{2}_{bst} B^{Sh}_{b}$$

$$\forall\, b \in \mathscr{B},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3e)$$

$$p^{L_f}_{lst} = -\frac{v_{bst}\, v_{b'st}}{L^{Ratio}_{l}} \left(G_l \cos\left(\theta_{bst} - \theta_{b'st}\right) + B_l \sin\left(\theta_{bst} - \theta_{b'st}\right)\right) + \left(\frac{v_{bst}}{L^{Ratio}_{l}}\right)^{2} \left(G_l + {}^{1}\!/_{2}\, G^{Sh}_{l}\right)$$

$$b = f(l),\ b' = t(l); \quad \forall\, l \in \mathscr{L},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3f)$$

$$p^{L_t}_{lst} = -\frac{v_{b'st}\, v_{bst}}{L^{Ratio}_{l}} \left(G_l \cos\left(\theta_{b'st} - \theta_{bst}\right) + B_l \sin\left(\theta_{b'st} - \theta_{bst}\right)\right) + \left(\frac{v_{b'st}}{L^{Ratio}_{l}}\right)^{2} \left(G_l + {}^{1}\!/_{2}\, G^{Sh}_{l}\right)$$

$$b = f(l),\ b' = t(l); \quad \forall\, l \in \mathscr{L},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3g)$$

$$q^{L_f}_{lst} = \frac{v_{bst}\, v_{b'st}}{L^{Ratio}_{l}} \left(-G_l \sin\left(\theta_{bst} - \theta_{b'st}\right) + B_l \cos\left(\theta_{bst} - \theta_{b'st}\right)\right) - \left(\frac{v_{bst}}{L^{Ratio}_{l}}\right)^{2} \left(B_l + {}^{1}\!/_{2}\, B^{Sh}_{l}\right)$$

$$b = f(l),\ b' = t(l); \quad \forall\, l \in \mathscr{L},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3h)$$

$$q^{L_t}_{lst} = \frac{v_{b'st}\, v_{bst}}{L^{Ratio}_{l}} \left(-G_l \sin\left(\theta_{b'st} - \theta_{bst}\right) + B_l \cos\left(\theta_{b'st} - \theta_{bst}\right)\right) - \left(\frac{v_{b'st}}{L^{Ratio}_{l}}\right)^{2} \left(B_l + {}^{1}\!/_{2}\, B^{Sh}_{l}\right)$$

$$b = f(l),\ b' = t(l); \quad \forall\, l \in \mathscr{L},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.3i)$$

Objective Function

$$z^* = \min \quad \sum_{s \in \mathscr{S}} \frac{W_s}{|\mathscr{T}_s|} \left(c_s^{Var} + c_s^{Shed} + c_s^{Curt} \right) \tag{6.4}$$

where:

$$c_s^{Var} = \sum_{t \in \mathscr{T}_s} \left(\sum_{g \in \mathscr{G}^T} \left[\beta \, p_{gst}^G \left(C_g^L + \frac{C_{n(g)}^{Fuel} + C^{CO_2} E_{n(g)}}{\eta_g} \right) + C_g^{Fix} \right] + \sum_{g \in \mathscr{G}^S} \left[\beta \, p_{gst}^{in} C_g^L + C_g^{Fix} \right] \right) \tag{6.4a}$$

$$c_s^{Shed} = \sum_{t \in \mathscr{T}_s} \sum_{p \in \mathscr{P}} \sum_{d \in \mathscr{D}_p^R} \beta \, l_{pst}^{Shed} \, C_d^{Shed} \left(B_d^P + |B_d^Q| \right) \tag{6.4b}$$

$$c_s^{Curt} = \sum_{t \in \mathscr{T}_s} \sum_{g \in \mathscr{G}^R} \beta \, p_{gst}^{Curt} \, C_g^{Curt} \tag{6.4c}$$

Here, Constraints (6.1) are bounds on the variables. Constraints (6.1a) and (6.1b) set the bounds for the voltage magnitude and voltage angle at each bus, respectively.

Constraint (6.1c) sets the bounds for real power generation and power discharge for generators and stores. Constraint (6.1d) sets the maximum and minimum absorption/injection of reactive power for a generator or a reactive power support device. Constraint (6.1e) sets the minimum and maximum curtailment of renewable generators, setting a maximum limit of curtailment equal to the power generated at that period.

Constraint (6.1f) sets the maximum charging input power to the stores, Constraint (6.1g) sets the minimum and maximum energy in the stores.

Unlike the DC model presented earlier, in this approach, the power shedding is done by profile and constraint (6.1h) limits the minimum and maximum load shedding percentage.

Constraints (6.2a) and (6.2b) are the maximum thermal limits of lines. Depending on the line properties and the power flow, the limit can be binding in one side or the other, hence the two constraints representing both the “from” and “to” ends of a line.

The next group corresponds to equality constraints (6.3). Constraint (6.3a) sets the angle of the reference bus to an arbitrary value of 0, constraint (6.3b) handles the energy level at the stores tracking how much energy enters or exits the store at any period. Constraint (6.3c) fixes the renewable generation according to an exogenous value.

Constraints (6.3d) and (6.3e) correspond to the real and reactive power balance per bus, respectively. And constraints (6.3f) - (6.3i) correspond to the power flow model. These constraints include susceptance and conductance for the lines and shunt conductance and shunt susceptance at lines and buses. As explained in Chapter 4, the base networks are *ReducedGB* and *IEEE33BW*, which only consider line susceptance and line conductance, and in the case of ReducedGB it also considers line shunt susceptance. A similar situation happens with the tap-ratios L_l^{Ratio}, as the base networks have tap-ratios of 1 at each line. The practical reason for having these elements in the model is that this OPF model has been coded and validated against diverse MATPOWER cases [68], and in many of those cases, these elements are considered and affect the solution. For simplicity, in the modified models presented later in this Chapter, the elements which are not used will be dropped.

The objective function (6.4) is a minimization objective of the weighted average hourly costs and is composed of 3 parts: (6.4a) gathers generation costs i.e. the sum of the power generation costs incurred by generators and the sum of the costs of charging the stores in all periods including fix cost; (6.4b) gathers the penalty costs for unserved demand (disconnecting loads); and (6.4c) includes the penalties paid by renewable generation curtailment. The optimal objective value of the problem is named z^*.

In addition to the elements described previously, there are auxiliary mapping functions and auxiliary subsets for the creation of constraints. The definition P_{dst}^D uses the mapping function m, which maps from demand d to profile p. In equation (6.4a) and the mapping function n is used to map from generator g to fuel f. Function o used in equation (6.3c) maps from renewable generator g to a (negative demand) profile.

The auxiliary subsets used in this model are $\mathscr{B}^{Ref}$ which only contains the reference bus, $f(l)$ and $t(l)$ are mapping functions from line l to the start and end buses

of the line, respectively; and for clarity they are linked to b and b' descriptions in Constraints (6.3f) - (6.3i). $\mathscr{L}_b^f$ is the subset of all the lines that are coming out of bus b, and $\mathscr{L}_b^t$ is the subset of lines going out of bus b. $\mathscr{D}_p^R$ represents the real demands subjected to profile p.

$\mathscr{G}_b^R$ is the set of incident renewable generators to bus b, $\mathscr{B}_b^S$ is the set of stores incident to bus b, and finally, $\mathscr{D}_b^R$ is the short-hand for the subset of real demands incident to bus b.

There are no ramping constraints in this model, but they can be easily added by limiting the change (increase/decrease) in generation level in consecutive periods.

The problem defined in this section is a non-linear non-convex model, in the formulation the objective is presented as linear for compactness but in the test cases there are small quadratic cost coefficients for thermal generation costs (similar to Eq. (2.2a)). Constraints (6.2a) and (6.2b) are non-linear but convex, constraints (6.3d), (6.3e) are quadratic equalities (non-convex) and (6.3f)-(6.3i) are clearly non-convex.

Based on the variables and constraints here presented, the size of the problem is shown in Table 5.1

Table 6.1: Number of elements in the AC-OPF formulation model

Variables	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(2\lvert\mathscr{B}\rvert + 2\lvert\mathscr{G}\rvert + 2\lvert\mathscr{G}^S\rvert + \lvert\mathscr{G}^r\rvert + \lvert\mathscr{P}\rvert + 4\lvert\mathscr{L}\rvert\right)$
Constraints	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(6\lvert\mathscr{B}\rvert + 4\lvert\mathscr{G}\rvert + 5\lvert\mathscr{G}^S\rvert + 3\lvert\mathscr{G}^r\rvert + 2\lvert\mathscr{P}\rvert + 6\lvert\mathscr{L}\rvert + 1\right)$
Variables on the Obj.	$\sum_{s\in\mathscr{S}} \lvert\mathscr{T}_s\rvert \left(\lvert\mathscr{G}\rvert + \lvert\mathscr{G}^S\rvert + \lvert\mathscr{G}^R\rvert + \lvert\mathscr{P}\rvert\right)$

Comparing this table to Table 5.1 (Page 46) it can be seen that the model is larger, with more variables and constraints, many of which are non-linear. If the increased size is combined with slower non-linear solvers, the model is significantly slower to solve than the DC approximation, as shown later in this chapter. Unlike the previously presented linear model, where problems up to 2880 periods could be solved without decomposition, in the non-linear framework, 48 periods was the maximum length for an undecomposed problem that fitted into the same machine's memory, highlighting the need for an efficient decomposition method.

6.3 Decomposition

The methodology here proposed is similar to the one presented in Section 5.3, with modifications on the interface due to the additional variables in AC-OPF. The spatial decomposition separates the network into a Transmission Network and several distribution networks. The transmission network corresponds to a multi-period OPF, and the distribution networks will be treated as single-period OPF problems.

Unlike the decomposition for the DC approximation from Chapter 5, the numerical results only include the cases with the transmission network embedded into the master problem since the maximum number of periods tested is relatively small (96 periods). Also, if decomposed, the interface for the transmission network would be high dimensional with active sets likely to be more non-linear.

6.3.1 Spatial decomposition

The motivation behind the spatial decomposition has been explained in Subsection 5.3.1, and it will not be repeated in this section. The new sets, new expanded sets and changes are valid for the AC-OPF problem.

In the AC-OPF decomposition, there are more variables in the interface that require coordination than those previously presented in Figure 5.2. The voltage magnitudes need to be the same on both sides of the interface; additionally, there is reactive power to be coordinated.

Given that this methodology is based on Benders decomposition, the master problem will set a value for every variable coordinated in the subproblems. Because only optimality cuts are used, the subproblems need to be augmented with flexibility elements; these allow a mismatch in the coordinated variables but with high linear penalties, which provides the flexibility needed to ensure the subproblem is always feasible.

The resulting spatial decomposition interface is shown in Figure 6.1.

Starting from a link similar to the one shown in Figure 5.1 (for the DC case), Figure 6.1 shows the modified interface for the AC-OPF case.

The interface variables that need to match are the voltage magnitudes v, voltage angles θ, real power p and reactive power q. In terms of Figure 6.1 this is:

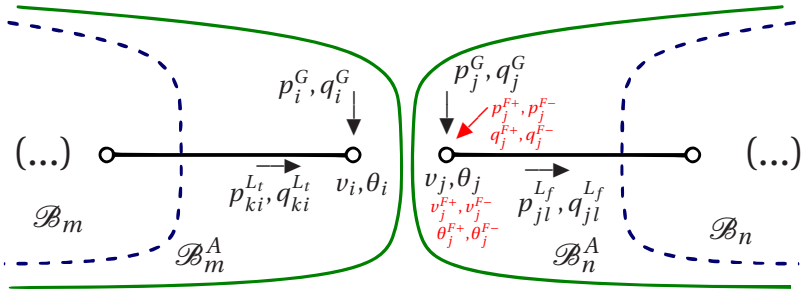


Figure 6.1: Flexible interface in the AC-OPF problem

$$p_i^G = -p_j^G \tag{6.5a}$$

$$q_i^G = -q_j^G \tag{6.5b}$$

$$v_i = v_j \tag{6.5c}$$

$$\theta_i = \theta_j \tag{6.5d}$$

The equations are written in the model in this form to keep a "generator" object on both sides of the interface as mentioned in Section 5.3.1. The negative signs on the fixed generation levels (equations 6.5a and 6.5b) mean that if the master problem is importing power (positive sign), then the subproblem will see this as an export (negative power).

On the right-hand side of the figure, the region needs to include feasibility elements to cope with requests that would otherwise be infeasible in the distribution networks.

A simple example of why these elements are needed can be easily seen when the master problem thinks that importing power is free. The master problem will fix the generation interface variable to take all the power it needs from the subproblem. If the subproblem does not have enough internal generation to generate as much as the master problem asks, the request will be infeasible. Something similar happens when the master problem wants to export a large amount of power to the subproblem; the subproblem can only reduce generation to the minimum, but if the power is larger than the power balance constraints allow, then the request will also be infeasible.

With flexibility variables, any mismatch between the requests and the subproblem

capabilities will be collected by these variables at a significant cost penalty, resulting in a high objective value of the subproblem and a high gradient for the request. The flexibility needs to be with both signs, positive and negative.

6.3.2 Temporal decomposition

The methodology to achieve temporal decomposition has been previously described in Subsection 5.3.2 for the DC case. In the AC-OPF problems, the elements involved in that link in time the subproblems are the same as in the DC case.

Similar to what was explained in the temporal decomposition for the DC OPF-based problems (Subsec. 5.3.2), the time decomposition is achieved by moving the time-linking constraints to the Master Problem and creating auxiliary elements to coordinate the relevant variables of the DNs. Only storage links the periods in the test cases presented in this thesis.

The generators in the distribution network subproblems are only fast response thermal generators, reactive support devices and small scale stores. The fast response generators (such as backup diesel generators) are unlikely to be time-linked because they can respond and reach maximum production capacity in less than one time step, so ramp limits do not effectively constrain them. The reactive support devices are also assumed not to be time-linked; only the storage technologies link the subproblems in time.

The ramping constraints are not included in the test cases for the distribution networks, as the temporal resolution of the examples is 30 minutes. The coarse temporal resolution may make these constraints not binding, but in cases where the temporal resolution is finer than in the presented test cases, these constraints may be binding and then they have to be handled as the other time-linking constraints (see Subsec. 5.3.2).

In the subproblems of the test cases, the variables related to stores need to be transferred to the master problem to achieve time decomposability; the master problem will propose generation/charging levels for each single-period subproblem and coordinate the energy levels of the stores in time. The detail of the modelling of this is the same as previously explained in Subsection 5.3.2.

The subproblems are also augmented with flexibility variables that allow the storage charging or discharging power to deviate from the value requested by the master problem, which allows the request to be feasible but at a high penalty.

6.3.3 Subproblem model flexibility

Another natural benefit of the decomposition is the ability to easily attach different model specifications in each subproblem and just agree on a common interface. This concept is also achievable in an undecomposed problem, but the spatial separability of the decomposed problem makes this situation natural, and if one of the subproblems has a different model because the agent owning the model (say a distribution network operator) decides so, all the changes happen within the sub-network, and there are no changes needed in the other distribution networks or the transmission network, i.e. the rest of the problem is unmodified.

The reason for having different problem specifications on the subproblems can be varied. In the test case from Chapter 4, each subproblem (distribution network) has a radial structure with only one connection point to the transmission network. A branch flow model can be then used.

In radial distribution networks, an alternative for voltage-based bus injection models are branch flow models (Subsection 2.2). In branch flow models voltage angles and magnitudes are replaced by squared voltage magnitude and uses squared current magnitudes. According to Baran and Wu [31], for radial networks this formulation has numeric benefits and it can reduce the solution time.

The Branch Flow model is a non-convex quadratically constrained quadratic problem whereas the voltage-based bus injection model which is general non-convex problem.

Suppose the voltage angles and voltage magnitudes are required. In that case, they can be calculated *a posteriori* from the optimal solution based on the real and reactive powers crossing the lines with a process like the one shown in [30].

6.3.4 Subproblem model description

As explained earlier, the spatial decomposition allows subproblems with different models to be easily connected. For the test cases described in Chapter 4, the subproblems are the distribution networks with a radial topology and with single connections to the transmission network; as a result, the branch flow model is a good fit.

The BFM model will be presented in this section.

Constraints

$$\underline{V}_b^2 \leq v_{bst} \leq \bar{V}_b^2 \qquad \forall\, b \in \mathscr{B}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6a)$$

$$\sum_{g \in \mathscr{G}_b} p^G_{gst} \quad + \sum_{l \in \mathscr{L}^t_b} (p^{L_f}_{lst} - R_l \ell_{lst}) =$$
$$\sum_{d \in \mathscr{D}^R_b} \left(P^D_{dst} - B_d D^P_{m(d)st} l^{Shed}_{m(d)st} \right) \quad + \sum_{g \in \mathscr{G}^R_b} p^{Curt}_{gst} \quad + \sum_{l \in \mathscr{L}^f_b} p^{L_f}_{lst} \quad + G^B_b v_{bst}$$

$$\forall\, b \in \mathscr{B}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6b)$$

$$\sum_{g \in \mathscr{G}_b} q^G_{gst} \quad + \sum_{l \in \mathscr{L}^t_b} (q^{L_f}_{lst} - X_l \ell_{lst}) =$$
$$\sum_{d \in \mathscr{D}^R_b} \left(Q^D_{dst} - B_d D^Q_{m(d)st} l^{Shed}_{m(d)st} \right) \quad + \sum_{l \in \mathscr{L}^f_b} q^{L_f}_{lst} \quad - B^B_b v_{bst}$$

$$\forall\, b \in \mathscr{B}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6c)$$

$$(p^{L_f}_{lst})^2 + (q^{L_f}_{lst})^2 \leq (\bar{S}_l)^2 \qquad \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6d)$$

$$(p^{L_f}_{lst} - R_l \ell_{lst})^2 + (q^{L_f}_{lst} - X_l \ell_{lst})^2 \leq (\bar{S}_l)^2 \qquad \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6e)$$

$$v_{l_t st} - v_{l_f st} = -2(R_l p^{L_f}_{lst} + X_l q^{L_f}_{lst}) + (R_l^2 + X_l^2)\, \ell_{lst} \qquad \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6f)$$

$$\ell_{lst} v_{l_f st} = (p^{L_f}_{lst})^2 + (q^{L_f}_{lst})^2 \qquad \forall\, l \in \mathscr{L}, \forall\, s \in \mathscr{S}, \forall\, t \in \mathscr{T}_s \quad (6.6g)$$

Constraint (6.6a) sets the bounds for the squared voltage, which replaces constraints (6.1a) and (6.1b). The power balance constraints (6.6b) and (6.6c) replace constraints (6.3d) and (6.3d); note that the power at the end of the lines is written as the power at the beginning of the line minus the change of power along the line depending on the square of the current and the impedance.

Constraints (6.6d) and (6.6e) are line limits similar to the ones presented earlier but written in terms of impedance and current squared. Constraint (6.6f) corresponds to Ohm's law, and finally, equation (6.6g) is the definition of power, where the expression on the left-hand side defines apparent power squared as the sum of real and reactive squared powers. The last two constraints together form the power flow part of the model.

6.3.5 Modifications to the model

An interface for information exchange has to be set up to decompose the model by region and time. The subproblems need to be augmented with feasibility variables that will make the subproblems always feasible.

In the following formulation, only one link between transmission and each distribution network is considered; the master problem (including the transmission network) is modelled using an AC-OPF Bus Injection polar formulation with the distribution networks using a Branch Flow formulation. The variables v (voltage magnitude) are used in the master problem and ν (squared voltage magnitude) in the subproblems, and these must be coordinated.

Master problem: objective function

$$z^{MP*} = \min \quad \left[\sum_{s \in \mathcal{S}} \frac{W_s}{|\mathcal{T}_s|} \sum_{r \in \mathcal{R}^{TN}} \left(c_{rs}^{Var} + c_{rs}^{Shed} + c_{rs}^{Curt} \right) + \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{T}_s} \sum_{r \in \mathcal{R}^{DN}} \alpha_{rst} \right] \tag{6.7a}$$

Master problem: constraints

$$\alpha_{rst} \geq \frac{W_s}{|\mathcal{T}_s|} \left[z_{rst}^{SP*} + \lambda_{rst}^{x\,\mathsf{T}} \left(x_{rst} - X_{rst}^{*i} \right) + \left(x_{rst} - X_{rst}^{*i} \right)^{\mathsf{T}} H \left(x_{rst} - X_{rst}^{*i} \right) \right]$$

$$\forall r \in \mathscr{R}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7b)$$

$$e^G_{gs(t+1)} = e^G_{gst} + \Delta t \left(\eta_g \, p^{in}_{gst} - p^G_{gst}\right) \qquad \forall g \in \mathscr{G}^{SCr}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7c)$$

$$p^{Net}_{gst} = p^G_{gst} - p^{in}_{gst} \qquad \forall g \in \mathscr{G}^{SCr}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7d)$$

$$\underline{E^G}_g \le e^G_{gst} \le \bar{E^G_g} \qquad \forall g \in \mathscr{G}^{SCr}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7e)$$

$$0 \le p^G_{gst} \le \bar{P^G_g} \qquad \forall g \in \mathscr{G}^{SCr}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7f)$$

$$0 \le p^{in}_{gst} \le \bar{P^G_g} \qquad \forall g \in \mathscr{G}^{SCr}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.7g)$$

Subproblem: objective

$$z^{SP*}_{rst} = \min \quad \left(c^{Var}_{rst} + c^{Shed}_{rst} + c^{Curt}_{rst} + c^{Feas}_{rst}\right) \quad (6.8)$$

where:

$$c^{Var}_{rst} = \sum_{g \in \mathscr{G}^T} \left[\beta \, p^G_{gst} \left(C^L_g + \frac{C^{Fuel}_{n(g)} + C^{CO_2} E_{n(g)}}{\eta_g}\right) + C^{Fix}_g\right] + \sum_{g \in \mathscr{G}^{SC}{}_d} \left[\beta \, p^G_{gst} C^L_g + C^{Fix}_g\right] \quad (6.8a)$$

$$c^{Shed}_{rst} = \sum_{p \in \mathscr{P}} \sum_{d \in \mathscr{D}^R_p} \beta \, l^{Shed}_{pst} \, C^{Shed}_d \left(B^P_d + |B^Q_d|\right) \quad (6.8b)$$

$$c^{Curt}_{rst} = \sum_{g \in \mathscr{G}^R} \beta \, p^{Curt}_{gst} \, C^{Curt}_g \quad (6.8c)$$

$$c^{Feas}_{rst} = \sum_{g \in \mathscr{G}^F} \beta \left(p^{F+}_{gst} + p^{F-}_{gst} + q^{F+}_{gst} + q^{F-}_{gst}\right) C^{FeasG}_g + \sum_{b \in \{\hat{b}_r\}} \left(v^{F+}_{bst} + v^{F-}_{bst}\right) C^{FeasV}_{bst} \quad (6.8d)$$

Subproblem: constraints

$$x_{rst} := X^*_{rst} \quad : \lambda^x_{rst} \qquad r \in \mathscr{R}^{DN}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9a)$$

$$p^{F+}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9b)$$

$$p^{F-}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F \cup \mathscr{G}^{FS}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9c)$$

$$q^{F+}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9d)$$

$$q^{F-}_{gst} \geq 0 \qquad \forall g \in \mathscr{G}^F, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9e)$$

$$v^{F+}_{bst} \geq 0 \qquad \forall b \in \{\hat{b}_r\}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9f)$$

$$v^{F-}_{bst} \geq 0 \qquad \forall b \in \{\hat{b}_r\}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9g)$$

$$v_{bst} = v^{Req}_{bst} + v^{F+}_{bst} - v^{F-}_{bst} \qquad \forall b \in \{\hat{b}_r\}, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9h)$$

$$\sum_{g \in \mathscr{G}_b} p^{G}_{gst} + \sum_{l \in \mathscr{L}^t_b} (p^{L_f}_{lst} - R_l \ell_{lst}) + \sum_{g \in \mathscr{G}^F_b \cup \mathscr{G}^{FS}_b} (p^{F+}_{gst} - p^{F-}_{gst}) =$$
$$\sum_{d \in \mathscr{D}^R_b} \left(P^D_{dst} - B_d D^P_{m(d)st} l^{Shed}_{m(d)st} \right) + \sum_{g \in \mathscr{G}^R_b} p^{Curt}_{gst} + \sum_{l \in \mathscr{L}^f_b} p^{L_f}_{lst} + G^B_b v_{bst}$$

$$\forall b \in \mathscr{B}^A_r, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9i)$$

$$\sum_{g \in \mathscr{G}_b} q^{G}_{gst} + \sum_{l \in \mathscr{L}^t_b} (q^{L_f}_{lst} - X_l \ell_{lst}) + \sum_{g \in \mathscr{G}^F_b \cup \mathscr{G}^{FS}_b} (q^{F+}_{gst} - q^{F-}_{gst}) =$$
$$\sum_{d \in \mathscr{D}^R_b} \left(Q^D_{dst} - B_d D^Q_{m(d)st} l^{Shed}_{m(d)st} \right) + \sum_{l \in \mathscr{L}^f_b} q^{L_f}_{lst} - B^B_b v_{bst}$$

$$\forall b \in \mathscr{B}^A_r, \forall s \in \mathscr{S}, \forall t \in \mathscr{T}_s \quad (6.9j)$$

The objective function in the master problem (6.7a) is the weighted cost of the variable generation costs, shedding and curtailment costs; it also considers the cost of the subproblems for every region and every period approximated by the variables α.

The constraints that are added to the master problem are the quadratic cuts (6.7b) above which the subproblem approximations need to be. The cuts have been weighted depending on the slices. Unlike the cuts added for the DC linear approximation, a quadratic term is added where a matrix H will be calculated according to

the algorithms presented later in this chapter. The constraint (6.7b) only shows the form of the cuts, proper indexing is shown and explained in Algorithm 6.1.

The constraints (6.7c) and (6.7d) handle the coordinated storage level and the net (positive or negative) request passed to the subproblem, respectively. Constraints (6.7e) - (6.7g) deal with the bounds for energy level, storage discharge and charging in the coordinated stores.

In terms of the subproblems, the objective function (6.8) considers only the costs in a region in a single-period; they have been augmented with feasibility penalties (6.8d). Flexibility in the requests is achieved by introducing non-negative variables with a high penalty cost to keep the subproblems feasible.

Constraint (6.9a) assigns the requests from the master problem (X^*) to a variable x to facility retrieving the gradient λ^x (dual value) with respect to these requests, and depending on the subproblem description, and the vector x can have different elements, e.g. power and voltages at the interface, coordinated storage, and different demand profiles.

Constraints (6.9b)-(6.9g) ensure the non-negativity of the flexibility variables. Finally, the same flexibility elements need to be added to the power balances in (6.9i) and (6.9j).

6.3.6 Cut shareability

Similar to the linear case (DC approximation), one of the main benefits of using a Benders-type algorithm, where a surrogate or approximate model is built, is that if the cuts are parametrised, they can be adapted or transferred to new values of those parameters. For example, if the demand level is parametrised, the cuts can be adjusted in height, gradient, and curvature for new demands. For other remarks on the use of demand profiles, see Sec. 5.3.4 on Page 57.

Unlike the linear case, where the parametrised cuts are always valid, in the non-convex case, they are not necessarily tight or even valid. Nonetheless, their information can be handy and accelerate convergence, especially when enough historical data is already gathered, such as problems that are solved every day.

As described earlier in this section, quadratic cuts are used for the AC OPF prob-

lem and the methodology here proposed. These cuts have the form shown in expression (6.10a).

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|}\left[z^{SP*}_{rst} + \lambda^{x\,\mathsf{T}}_{rst}\left(x_{rst} - X^{*i}_{rst}\right) + \left(x_{rst} - X^{*i}_{rst}\right)^{\mathsf{T}} H \left(x_{rst} - X^{*i}_{rst}\right)\right]$$

$$\forall\, r \in \mathscr{R},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.10a)$$

Suppose the variables x, which in this expression are assumed to contain only spatial information (i.e. the variables at the transmission-distribution interface and coordinated storage), are augmented with the levels of the demand profiles. In that case, they form the expanded variable $\hat{x}$, which also contains information for the exogenous profiles $\hat{X}$.

The new cut description includes the expanded $\hat{x}$, with the corresponding expanded sensitivity $\lambda^{\hat{x}}$ and expanded curvature matrix $\hat{H}$.

These simple changes coupled with the storage transfer to be coordinated in the master problem allows the master problem to estimate the subproblem response to different demand profile levels while only needing the subproblems to be solved as single-period problems.

The expression (6.10a) is then changed for expression (6.10b)

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|}\left[z^{SP*}_{rst} + \lambda^{\hat{x}\,\mathsf{T}}_{rst}\left(\hat{x}_{rst} - \hat{X}^{*i}_{rst}\right) + \left(\hat{x}_{rst} - \hat{X}^{*i}_{rst}\right)^{\mathsf{T}} \hat{H} \left(\hat{x}_{rst} - \hat{X}^{*i}_{rst}\right)\right]$$

$$\forall\, r \in \mathscr{R},\ \forall\, s \in \mathscr{S},\ \forall\, t \in \mathscr{T}_s \quad (6.10b)$$

For clarity, the hat operator ($\hat{\cdot}$) will be dropped, and it will be assumed that the profile level sensitivity will always be gathered. Note that the cuts have the form specified in this section, but the proper indexing is shown in Algorithm 6.1 (Page 114).

6.3.7 Transmission-distribution interface description

The model formulation and the type of OPF on both sides (transmission and distribution) will determine the interface characteristics. In a general AC OPF network, the variables at the interface will be p^G, q^G, v and θ.

Suppose there is only one connection point between the transmission region and a distribution region and that the distribution regions are not linked to each other. In that case, the θ variable can be dropped in the interface as the master problem will not need to enforce angle consistency other than in the internal elements of the master problem, such as in the test case presented in Chapter 4.

Concerning the temporal decomposition element information passed between the levels of the problem, the stores are only required to be fixed in terms of real power p^G as the assumption is that storage will not generate or supply reactive power.

The other elements needed for the temporal decomposition are the profile levels, which are exogenous parameters known by the master problem and determine the time-series demands.

As with cutting plane approximations, the other information that needs to cross between levels is the objective value of the subproblem and all the gradients with respect to the fixed variables.

In the proposed approach, the non-linear subproblems need to detect the active set to high precision at the solution for each sampled point since only one cut per active set of the subproblem is kept in the master problem. So failure to detect the active set correctly will either make the algorithm fail or take a longer time to converge.

Based on what was described in this subsection, the dimension of the cuts will be the following:

$$dim(\alpha_{rst}) = 4|\mathcal{N}_r| + |\mathcal{G}_r^{SCd}| + |\mathcal{P}_r| \qquad \text{(Multi-link)}$$

$$dim(\alpha_{rst}) = 3 + |\mathcal{G}_r^{SCd}| + |\mathcal{P}_r| \qquad \text{(Single link)}$$

where $\mathcal{N}_r$ is the set of links between the distribution network region r and the transmission network.

As the profiles are exogenous to the problem (on both levels) and are not opti-

mization variables, their value and sensitivity will only be used to adjust the predicted height for new profile levels. The true dimension of the cut is then lower than presented here and can be reduced by $|\mathscr{P}_r|$.

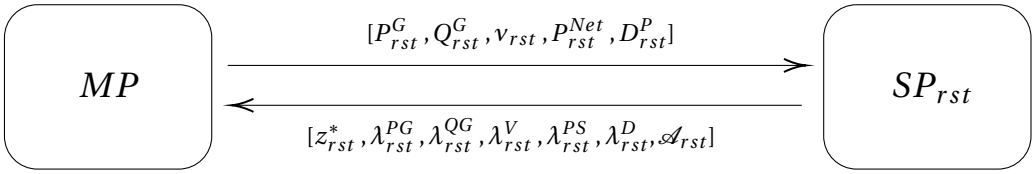


Figure 6.2: Data exchange between master problem and each single-period subproblem

In Figure 6.2 the data transferred between parts in each iteration is shown. The vector of requests needs to be passed to the subproblems. Note that P^{Net} and D^P will include as many elements as there are coordinated stores and demand profiles in the subproblem.

6.3.8 Algorithms

This section describes the algorithms used in the decomposition. First, it presents the Benders based algorithm and its notation, and then it presents the supporting routines as they appear in the modified Benders algorithm.

Modified Benders Decomposition

For the non-convex problem presented in this chapter, it is not possible to apply a standard Benders decomposition directly and expect to find a good feasible solution. Nonetheless, the algorithm can be modified as described in this section to address the issues related to the problem's mild non-convexity close to the optimal solution and define stopping criteria that give good solutions, as shown in the results subsection.

The algorithm presented in this section can be seen only as a heuristic method. It is not possible to assess if the point of convergence is optimal, as the thesis has not explored the theory supporting this claim, nor we expect to find true optimal points with the method. The algorithm was designed based on the observations that the test cases presented quasi-convex value surfaces around the true feasible areas

(where there is no load-shedding), so it is a modified version of standard Benders to take into account and correct effects of slight concavity.

The general algorithm is presented in Algorithm 6.1, with its subroutines in Algorithms 6.2 to 6.6. The specific required notation for the algorithms is first presented.

The main idea of the modified Benders algorithm relies on constructing locally "valid" quadratic cut approximations to each subproblem, keeping only one instance per active set.

In the first step in Algorithm 6.1 (Lines 1-4) the problem is separated into a master problem and single-period regional subproblems, and the maximum number of iterations and the convergence tolerance are set. The next step (Line 5) calls the guidance bounds subroutine (Alg. 6.2) which removes from the search area some regions where the solution is very unlikely to occur. This subroutine generates absolute bounds for each region for each time period, and these bounds will be included as box constraints in the master problem.

The iterative part of the algorithm is shown between Lines 9 and 53. The algorithm will stop when the maximum number of iterations is reached or, when the target convergence tolerance has been reached and the cuts have been checked.

The objective function of the master problem is augmented with stabilisation terms in Line 12. The master problem is solved in Line 14 and then the interface variable values are gathered.

Each subproblem is solved and the resulting information gathered (Lines 17-22). The information collected when solving each subproblem will include the subproblem objective value and dual information with respect to the interface variables. Also, the active set is determined based on the dual information from all the constraints in the subproblem (Alg. 6.3). If the active set has not been detected before, it will be added as a new entry to a list, called the dictionary of samples; on the other hand, if it was detected earlier, its relevant information (point, duals and objective value) will be added to the record of that active set. A small illustrative example of the structure of the dictionary of samples can be seen in Annex A.

In Line 24, a valid upper bound is calculated with the information from the master problem objective, replacing the value of approximations with the true value of the subproblems.

In Line 25, each subproblem is classified in one of four categories, which will be described in detail in Algorithm 6.4 as: converged, violated, slightly violated or none of those.

Each time a violated subproblem is detected, then it is updated with true active set heights (Lines 28-31) to avoid exploring an area that is not valid. A slackness check is performed in Algorithm 6.3 to detect the cuts that are causing the problem.

A cut check flag is activated when all the subproblems have converged with no significant violations in Line 33. In Line 39, all the detected active sets will add or update cuts with the closest information available and an approximation of curvature.

Finally, suppose the cut check flag was activated earlier. In that case, the guidance bounds and stabilisation term are removed, and the MP is solved again to calculate a "verified" lower bound and gap (Lines 46-51).

The notation for the algorithms can be found in the end of the notation section at the start of the Thesis.

Algorithm 6.1 Benders algorithm for the AC OPF spatial and temporal TN-DN decomposition

Separate: the undecomposed problem into regions (Section 6.3).
Map: links between the master problem and the subproblems.
Define: ϵ^{rel}, ϵ^{act}, $\bar{I}$
Set: $i \leftarrow 1$, $\bar{Z} \leftarrow \infty$, $\underline{Z} \leftarrow -\infty$, $G^{Rel} \leftarrow 1$, $\mathscr{X}_r \leftarrow \emptyset$, $\mathscr{A}_r \leftarrow \emptyset$, $C^{Check} \leftarrow 0$
for all $r \in \mathscr{R}^{\mathscr{D},\mathscr{N}}$, $s \in \mathscr{S}$, $t \in \mathscr{T}_s$ **do**
    Call: Guidance bound subroutine:
        $(\underline{\chi}^X_{rst}, \bar{\chi}^X_{rst}) \leftarrow$ **call** Alg. 6.2 (r, s, t)
    Add: Guidance bounds $(\underline{\chi}^X_{rst}, \bar{\chi}^X_{rst})$ to MP^i
end for
while $i \leq \bar{I}$ and $\left(G^{Rel} > \epsilon^{Rel}$ or $C^{Check} \neq 1\right)$ **do**
    if $i > 1$ **then**
        Calculate: stabilisation terms
        $p^{Stab} = \sum_{s\in\mathscr{S}} \sum_{t\in\mathscr{T}_s} \sum_{r\in\mathscr{R}^{DN}} \left[K^{Pen}_{rst}\right]^{\mathsf{T}} \left(x_{rst} - x^{*(i-1)}_{rst}\right)^2$
        Augment: MP Objective with stabilisation term; $z^{MP}_i \leftarrow z^{MP} + p^{Stab}_i$
    end if
    Solve: MP_i
    Set: $\underline{Z} \leftarrow z^{MP*}_i - p^{Stab*}_i$, $C^{Check} \leftarrow 0, \mathscr{P}^{Conv} \leftarrow \emptyset, \mathscr{P}^{Viol} \leftarrow \emptyset, \mathscr{P}^{SlViol} \leftarrow \emptyset$
    Collect: Interface variables values x^*_{rst}
    for all $r \in \mathscr{R}^{\mathscr{D},\mathscr{N}}$, $s \in \mathscr{S}$, $t \in \mathscr{T}_s$ **do**
        Fix: variables in the at the interface $x_{rst} \leftarrow x^*_{rst}$ in SP_{rst}
        Solve: SP_{rst}
        Classify: active set of the solution depending on its active inequalities
            $a_{rst} \leftarrow$ **call** Alg. 6.3 $\left(\lambda^{I*}_{rst}\right)$
        Collect: $\chi_{rst} \leftarrow [z^{SP*}_{rst}, \lambda^{SP*}_{rst}, x^*_{rst}]$
        Augment: the information on the active set a_{rst}
            $\mathscr{X}_{ra} \leftarrow \mathscr{X}_{ra} \cup \{\chi_{rst}\}$
    end for
    Calculate: a valid upper bound at the current point, the best upper bound and suspected gap.
        $\hat{G} \leftarrow \sum_{r\in\mathscr{R}^{SP}} \sum_{s\in\mathscr{S}} \sum_{t\in\mathscr{T}_s} \left(\frac{W_s}{|\mathscr{T}_s|} z^{SP*}_{rst} - \alpha^*_{rst}\right)$
        $\bar{Z} \leftarrow \underline{Z} + \hat{G}$
        $\bar{Z}^{Best} \leftarrow \min\left(\bar{Z}^{Best}, \bar{Z}\right)$
        $G^{Rel} \leftarrow \left(\bar{Z}^{Best} - \underline{Z}\right) / \bar{Z}^{Best}$
    for all $r \in \mathscr{R}^{\mathscr{D},\mathscr{N}}$, $s \in \mathscr{S}$, $t \in \mathscr{T}_s$ **do**
        Define: the status of the subproblem and augment the problem list
        $\left(\mathscr{P}^{Conv}, \mathscr{P}^{SlViol}, \mathscr{P}^{Viol}\right) \leftarrow$ **call** Alg. 6.4 $\left(\mathscr{P}^{Conv}, \mathscr{P}^{SlViol}, \mathscr{P}^{Viol}, SP_{rst}\right)$
    end for

Algorithm 6.1 (cont.) Benders algorithm for the AC OPF spatial and temporal TN-DN decomposition

for all $p \in \mathscr{P}^{Viol}$ **do**
  Map: $p \mapsto rst$
  Add: local information with the true height of the binding cuts:
    $\mathscr{X}_r \leftarrow$ **call** Alg. 6.5 $(\mathscr{X}_r, r, s, t)$
end for
if $|\mathscr{P}^{Conv}| + |\mathscr{P}^{SlViol}| = |\mathscr{R}^{DN}| \left(\sum_{s\in\mathscr{S}} |\mathscr{T}_s|\right)$ **then**
  Set: $C^{Check} \leftarrow 1$
  for all $p \in \mathscr{P}^{Conv} \cup \mathscr{P}^{SlViol}$ **do**
    Map: $p \mapsto rst$
    Add: local information with the true height of the binding cuts:
      $\mathscr{X}_r \leftarrow$ **call** Alg. 6.5 $(\mathscr{X}_r, r, s, t)$
  end for
end if
for all $r \in \mathscr{R}^{\mathscr{DN}}$, $a \in \mathscr{A}_r$, $s \in \mathscr{S}$, $t \in \mathscr{T}_s$ **do**
  Fit: the closest information to current exploration point:
    $(Z_{rast}, X_{rast}, L_{rast}, H_{rast}) \leftarrow$ **call** Alg. 6.6 (a, r, s, t, x^*_{rst})
  Update: cuts

$$\alpha_{rst} \geq \frac{W_s}{|\mathscr{T}_s|}\left[Z_{rast} + L_{rast}\,(x_{rst} - X_{rast}) + (x_{rst} - X_{rast})^{\mathsf{T}}\; H_{rast}\;(x_{rst} - X_{rast})\right]$$

end for
if $C^{Check} = 1$ **then**
  Remove: stabilisation term; $z_i^{MP} \leftarrow z^{MP}$
  Remove: guidance bounds from MP
  Solve: MP^i and *set:* $\underline{Z} \leftarrow z_i^{MP*}$
  Calculate: "verified" gap; $G^{Rel} \leftarrow \left(\bar{Z}^{Best} - \underline{Z}\right) / \bar{Z}^{Best}$
end if
$i \leftarrow i + 1$
end while

Guidance bounds subroutine

If the problem is an LP and the value surface is convex, the cuts are globally valid; this is the approach followed in Chapter 5 for the DC approximation case. However, this situation is more problematic when dealing with a non-convex problem because the cuts generated far away from the solution may be active but invalid at the solution,

and they may also result in the algorithm exploring areas far away from the optimal.

If this is the case, the problem status subroutine described in Algorithm 6.4 will detect this problem eventually and realise that the subproblem approximation is not accurate and has not converged, possibly being held at a wrong point because of the invalid cuts. The problem can be corrected (Alg. 6.3), but only after more iterations, and many of these may be exploring wrong areas of the problem.

The Guidance Bounds routine attempts to mitigate this issue at the expense of solving a modified version of the subproblem that will give the maximum interface variable bounds under some reasonable initial assumptions.

In the particular implementation for the examples in this chapter, the goal is to find the maximum capabilities of the subproblems for a given set of demands without load shedding. It is important to mention that the guidance bounds concept can be applied differently; some alternatives would be assuming merit-order and limiting the search around this precalculated solution or solving a fast SOCP relaxation and expanding the bounds around this solution.

As shown in Algorithm 6.2, in this particular implementation, the subproblem will be solved by removing the interface flexibility (the feasibility variables added to the interface), fixing the shedding variables to zero and then maximizing and minimizing the regional interface variables. The solution from these problems will be used to generate a box constraint in the master problem so that the master problem will not propose points that are too far away from the expected behaviour of the region. These restrictions will not exclude the optimal solution of the full problem, provided that there are no load disconnections in the optimal solution.

Furthermore, based on Line 48 of Algorithm 6.1 these bounds will be removed eventually to check whether the original optimal problem is stable even without the Guidance bounds, or move towards an unexplored better point.

It is important to mention that although in the auxiliary problems, the flexibility variables are removed and load shedding is set to zero, when the master problem proposes points, these could still require the use of flexibility variables to achieve feasibility, as these guidance bounds do not capture the detailed dynamics or inter-dependence of the variables in the subproblem; nonetheless, the proposed points are restricted to small areas where the cut corrections are likely to be smaller, and the

problem may converge more steadily.

On the other hand, solving auxiliary subproblems for each period takes time and depending on the case, it may not be worth using a routine like the one here presented. Alternative faster Guidance bounds routines, such as removing the network and only looking at demands and generation capabilities assuming a level of line losses, could also give reasonable guidance bounds and are possibly faster to compute.

For a multi-period problem, other alternatives include grouping levels of demand and only solving auxiliary subproblems for these groups, avoiding having to solve each period and speeding up the process.

Furthermore, suppose problems with the same network properties but different demands have been solved previously, and the samples stored in a dictionary. In that case, the starting point routines may not be needed, as the master problem will have access to sample information giving good approximations of the subproblems.

In the test case presented in this thesis for the AC OPF problem, it is constructed to be feasible without load shedding at the optimal. The master problem is connected to numerous regions, and without any knowledge of the capabilities of each subproblem, it will propose points that are convenient for its myopic view. If the subproblems have a relatively small generation capability compared to their demand, they are likely to import power instead of exporting. However, the master problem will initially explore points where the subproblems export power, as this appears to be possible at no cost.

Suppose no previous information is given to the master problem. In that case, the master problem will discover this situation creating cuts that model the costs of exporting power from the subproblem (such as in the DC approximation case) by exploring areas of the subproblems.

In terms of storage variables, as explained earlier in Subsection 6.3.5, the master problem already knows the capabilities of the “coordinated” storage, so it will never propose infeasible values for these.

Algorithm 6.2 Guidance bounds subroutine

Receive: r, s, t
Select: for subproblem SP_{rst}
Fix: to zero or delete feasibility variables e.g. $p^{F+}_{gst}, p^{F-}_{gst}, q^{F+}_{gst}, q^{F-}_{gst}, v^{F+}_{bst}, v^{F-}_{bst}$
Fix: to zero load shedding variables l^{Shed}_{pst}
Set: demand level for the time period D^{P}_{pst}, D^{Q}_{pst}
Set: $\underline{\chi} \leftarrow \emptyset, \bar{\chi} \leftarrow \emptyset$
for all $i \in \{p^G, q^G, v\}$ (the TN-DN exchange variables) **do**
    Set: the objective function to minimize the variable:
        $\min i$
    Solve: SP_{srt}
    if SP_{rst} is optimal **then**
        $\underline{\chi} \leftarrow \underline{\chi} \cup \{i\}$
    end if
    Set: the objective function to maximize the variable:
        $\max i$
    Solve: SP_{srt}
    if SP_{rst} is optimal **then**
        $\bar{\chi} \leftarrow \bar{\chi} \cup \{i\}$
    end if
end for
return $\underline{\chi}, \bar{\chi}$

Quadratic penalty stabilisation

The approximation of the subproblems in the master problem is made by adding quadratic cuts representing different active sets in the subproblem. The approach includes only one cut per active set, as shown in Algorithm 6.1.

In linear problems where an active set defines a linear subspace, a Benders cut will coincide exactly for the whole region where the active set is optimal, whereas adding Benders cuts in convex problems will add linear cuts that are always valid, but may not be exact in points other than the one that generated it. In these cases, adding more cuts per active set will form a progressive linear approximation (seeing by the master problem as a piecewise linear) to the subproblem, which with sufficient cuts will approximate the subproblem to high accuracy.

In linear or convex problems, stabilisation may be helpful to speed up conver-

gence, as it will restrict the steps taken by the master problem according to some metric or method. There are numerous methods for stabilisation, such as the one implemented in the results from this Chapter by adding quadratic penalties for displacement from the previous solution (Alg. 6.1, Line 11).

Suppose a method that uses a single cut per active set is used for a problem where the active set regions are not linear. In that case, stabilisation is an essential tool necessary to ensure the problem converges.

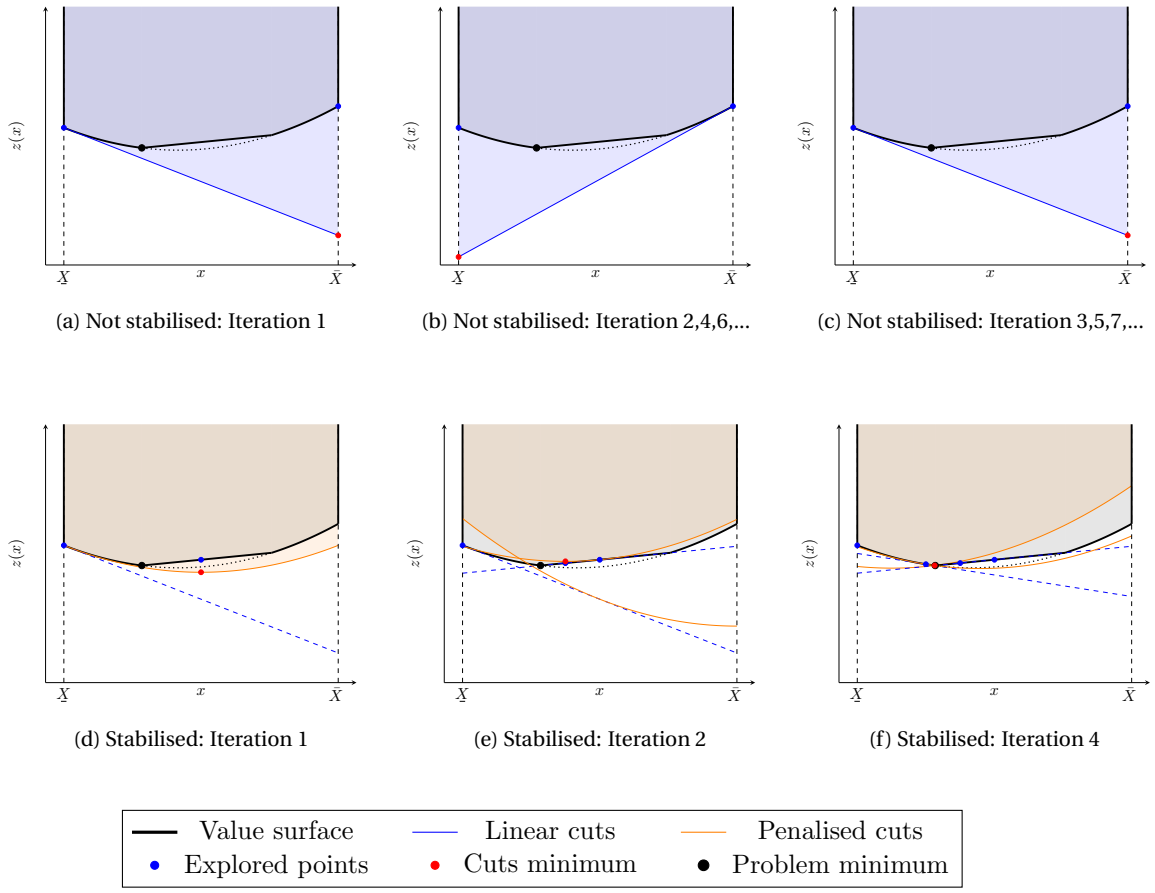
In the methodology proposed in this chapter, the quadratic cuts are always adjusted to be a good fit for the closest explored point (by weighted distance, per active set) for use in the next iteration of the master problem. If the proposed point moves, the new quadratic approximation will replace the current one, potentially making the master problem unstable and causing it to jump back to the previously explored area.

In the methodology, a simple quadratic penalty is added when there is displacement, effectively restricting the step the master problem tries to take. For this approach, a key parameter to define is the size of the penalties. In the test cases, even a simple, unique penalty (not differencing between real power, reactive power and voltage) was helpful.

In Figure 6.3 a schematic case is presented showing how the unstable situation can happen on how the stabilisation can avoid it. In Figure 6.3 a case for single linear cuts per active set is shown. The value surface comprises only two different active sets, one quasi-quadratic active set (on the left part of the function, right and the dotted line in the middle), and one linear active set (middle of the function). In Figures 6.3a and6.3b, a linear cut is replaced and the optimal given by the cut shifts between left and right (marked in red).

Suppose a quadratic penalty is added to the objective function for displacement against the previous solution. In that case, the problem then finds a minimum in the middle (Figure 6.3e), where it adds a newly discovered active set (and cut). The problem repeats the process until it converges (Figure 6.3f).

In Figures 6.3d-6.3f, for conceptual understanding, instead of minimizing the objective function ($\min z(x)$), the cuts have been curved with a quadratic penalty from the previous solution depending on the distance to the point. This approach



(a) Not stabilised: Iteration 1 (b) Not stabilised: Iteration 2,4,6,... (c) Not stabilised: Iteration 3,5,7,...

(d) Stabilised: Iteration 1 (e) Stabilised: Iteration 2 (f) Stabilised: Iteration 4

Figure 6.3: Not stabilised vs. Stabilised problem, single linear cut per active set

is equivalent for the example without having to change the objective function to $\min z(x) + \|x - X\|_2^2$, which is challenging to picture graphically.

Approximate active set detection subroutine

After the master fixes the requests for the subproblems, the subproblems are solved and depending on the inputs, the optimization will converge to a point, with a minimum that represents a mode of operation of the subproblem, e.g. the local generators producing at full capacity. This situation will make some inequality constraints active at the solution (such as the upper bound on capacity generation in this example).

The set of binding inequalities at an optimal solution plus all equalities that are binding for the subproblem to be primal feasible is called here an active set. In theory, when looking at the slackness and dual value of the inequalities, it is possible to assess if the constraint is active or not; in practice, however, depending on the

solution method of the subproblem, the active set may not be so clear to detect. Suppose the subproblem is solved with an Interior Point method. In that case, the problem solved is a modification of the original problem augmented with barrier terms that contaminate the duals values to a barrier penalty tolerance.

With an Active Set Method, the optimizer solves a sequence of subproblems based on a quadratic model of the original subproblem by seeking the active set of inequalities following an exterior path to the solution. In the subproblem testing, due to finite numerical precision, it is not so straightforward to retrieve the proper active set in the modelling language; instead, Algorithm 6.3 is being used to determine a very likely active set for the solution.

In the algorithm, there is a small value threshold ϵ^{AS}, and when the dual value is above it (either positive or negative, depending on the sign of the inequality), the constraint is classified as active. The list of constraints is then returned to the dictionary of samples for classification.

In practice, there are situations where both the dual and the slack of the constraint are exactly zero, and according to this algorithm, these constraints will not be classified as active, even if the slackness would tell otherwise. However, these are considered redundant constraints that could be removed without changing the solution.

Algorithm 6.3 Approximate active set detection subroutine

Receive: $\lambda^{I*} \quad \forall j \in \mathscr{I};\ \mathscr{I} = \mathscr{I}^+ \cup \mathscr{I}^-$
Set: $a \leftarrow \emptyset$
for all $j \in \mathscr{I}$ **do**
    if $j \in \mathscr{I}^+$ **and** $\lambda_j^{I*} \geq \epsilon^{AS}$ **then**
        $a \leftarrow a \cup \{j\}$
    else if $j \in \mathscr{I}^-$ **and** $\lambda_j^{I*} \leq -\epsilon^{AS}$ **then**
        $a \leftarrow a \cup \{j\}$
    end if
end for
return a

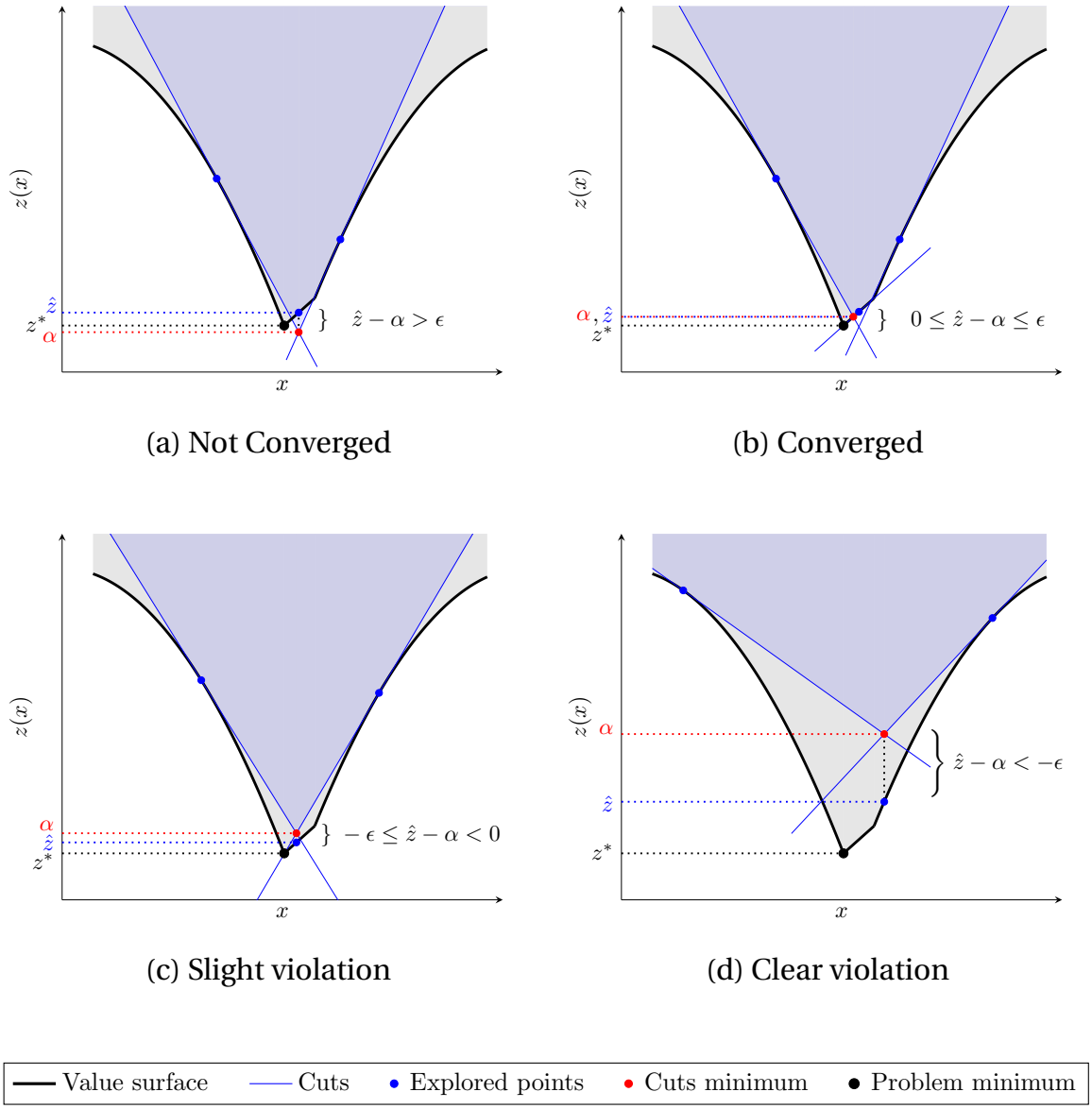
Subproblem status subroutine

When solving the decomposed problem, a way of assessing if the subproblems have converged at a point is to look at the height of the master problem approximation at the point compared to the subproblem real value at the given point. Each solved subproblem is classified in one of four statuses that depend on this metric. This classification methodology is shown in Algorithm 6.4 and the different convergence statuses are illustrated in Fig. 6.4.

If the approximation value is far below the real subproblem value, then it is assumed that the approximation is still loose and the subproblem has *not converged* (Fig. 6.4a). When adding more cuts (or adjusting the existing ones), the value surface of the subproblem can be more accurately defined. If the approximation is close and marginally below or exact when compared to the real value then the subproblem is given a temporary status of *converged* at the current point (Fig. 6.4b).

If the approximation is marginally above the real subproblem value, then the subproblem is given the status of *slightly violated* (Fig. 6.4c). It is assumed that even if the cuts defining the convergence point are not valid, their errors will not affect the master problem solution significantly.

A different case happens when there is a clear overestimation of the subproblem by the cuts defining the approximation. When that happens, it is better to address the overestimation issues as soon as this is detected to avoid exploring parts of the problem that will not be relevant to the solution (as shown in Algorithm 6.1). The subproblem is then given the status of *violated* (Fig. 6.4d).



(a) Not Converged

(b) Converged

(c) Slight violation

(d) Clear violation

Figure 6.4: Subproblem status

Algorithm 6.4 Subproblem status subroutine

Receive: $\mathscr{P}^{Conv}, \mathscr{P}^{SlViol}, \mathscr{P}^{Viol}$
if $0 \leq z^* - \alpha^* \leq \bar{Z}\epsilon^{Conv}$ **then**
    $\mathscr{P}^{Conv} = \mathscr{P}^{Conv} \cup \{rst\}$;
else if $z^* - \alpha^* \leq 0$ **then**
    if $z^* - \alpha^* \leq -\bar{Z}\epsilon^{SlViol}$ **then**
        $\mathscr{P}^{Viol} = \mathscr{P}^{Viol} \cup \{rst\}$;
    else
        $\mathscr{P}^{SlViol} = \mathscr{P}^{SlViol} \cup \{rst\}$;
    end if
end if
return $\mathscr{P}^{Conv}, \mathscr{P}^{SlViol}, \mathscr{P}^{Viol}$

Active cut check with true height

Algorithm 6.5 shows a practical way of calculating the true height of a given active set in a point of the subproblem. The algorithm shows the simple process, where the active set inequalities of the cut are fixed, and the other constraints are relaxed. Depending on the degrees of freedom of the subproblem, the true height calculation can be either an optimization problem or an equation solving problem. If there are no degrees of freedom left after fixing the active set, such as in a vertex solution, it will only be an equation solving problem; if the point lies in the middle of a face, then a minimization procedure will take place. In either case and given the reduced dimension (degrees of freedom) of the problem, the true height calculation is an inexpensive process.

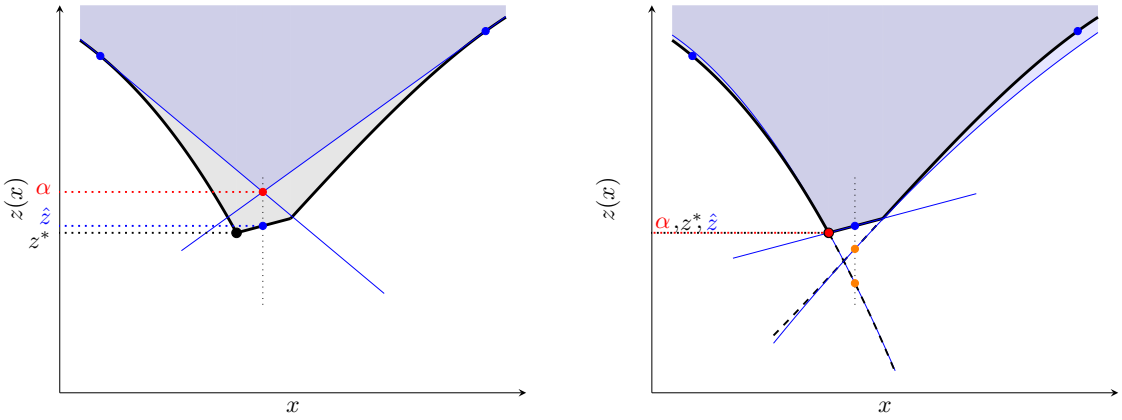
Algorithm 6.5 Active cut check with true height sampling subroutine

Receive: $\mathscr{X}_r$, r, s, t
for all $a \in \mathscr{A}_r^{DN}$ **do**
  Calculate: clearance of cut
  $C_{rast}^{Cut} \leftarrow \alpha_{rst}^* - \left(\frac{W_s}{|\mathscr{T}_s|} \left[Z_{rast} + L_{rast} \left(x_{rst}^* - X_{rast} \right) + \left(x_{rst}^* - X_{rast} \right)^\mathsf{T} H_{rast} \left(x_{rst}^* - X_{rast} \right) \right] \right)$
  if $C_{rast}^{Cut} < \epsilon^{Act}$ **then**
    for all $j \in \mathscr{J}_r^{SP}$ **do**
      if $j \in a_{rst}$ **then**
        Fix: constraint j to its bound
      else
        Relax: constraint j (remove bounds)
      end if
    end for
    minimize: SP_{rst}
    Augment: $\mathscr{X}_{ra} \leftarrow \mathscr{X}_{ra} \cup \{\chi_{rst}\}$
  end if
end for
return $\mathscr{X}_r$

Figure 6.5 illustrates the process of updating the cuts with the information given by the true height of each active set. First, in Figure 6.5a two linear cuts, generated at the left and right blue points, led the master problem to the blue middle point. When the subproblem is solved at the blue point in the middle, a clear violation

is detected because the height returned by the subproblem ($\hat{z}$) is clearly below the master problem approximation (α). The subproblem also returns the gradients and has discovered a new active set.

In Figure 6.5b, the true height at the point for the cuts defining the violation has to be determined. The true height of the active sets (orange points) is calculated using Algorithm 6.5. The old cuts have now 2 points of information and are fitted quadratically. Finally, the master problem is solved again with updated cuts and converges to a near-optimal point (closer to it would have been if the updated cuts had been linear).



(a) Before cut check

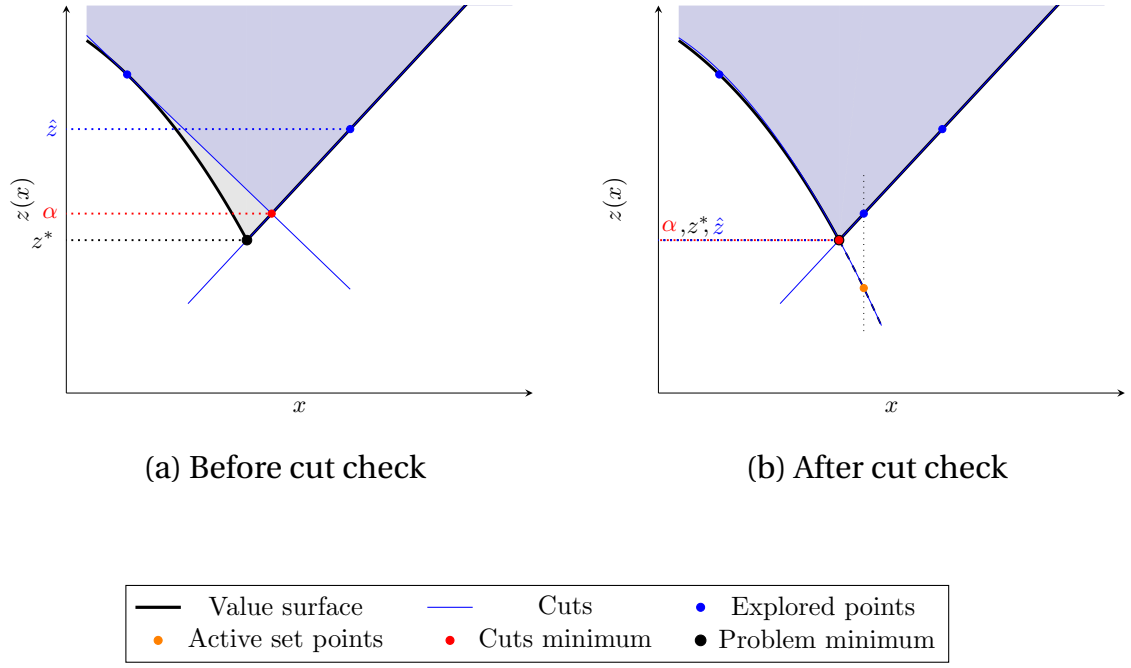
(b) After cut check

Figure 6.5: True height adjustment: clear violation case

The detection of the clear violation in Figure 6.5 starts the active cut check subroutine immediately (Lines 28-31 in Alg. 6.1). In other cases, detecting cuts potentially cutting of the optimal soution is more difficult. Figure 6.6, shows the rationale behind the cut check procedure when all the subproblems have been classified as converged (Lines 32-37 in Alg. 6.1).

In Figure 6.6a, the explored blue point to the right adds the linear cut on the right. When the master is solved again, the new convergence point (shown in red) is used to sample the subproblem. When sampled, the subproblem value agrees with the approximation exactly, and the subproblem is given the status of converged.

Then, in Figure 6.6b an active cut check subroutine is performed, which detects the true height of the active set (orange point). The cut is updated with an approximate curvature, and the master problem is solved one more time. Finally, the problem is converged close to the optimal point.



(a) Before cut check

(b) After cut check

Figure 6.6: True height adjustment: converged case

Cut fitting subroutine

The cut definition used in Algorithm 6.1 requires a cut height Z_{rast}, gradients with respect to fixed variables L_{rast} and curvature information H_{rast} all around a point X_{rast}

Defining a gradient and a height is performed by selecting the closest information to the current exploration point suggested by the master problem (hence the subindex a). In contrast, the curvature information is approximated by a Quasi-Newton method fitted with respect to that same point.

Quasi-Newton methods require only the gradient of functions at each iteration. By measuring the changes in these gradients, they construct approximations to the second-order information that are good enough to produce superlinear convergence [69]. Unlike exact Newton's method, they do not require to be provided with second-

order information, so they can provide a good speed advantage when it is expensive to calculate the Hessian. These methods are commonly used for unconstrained optimization.

In Quasi-Newton methods, the underlying idea is that two successive iterates x^k and x^{k+1} together with their gradients $\nabla f(x^k)$ and $\nabla f(x^{k+1})$ contain curvature information, i.e.

$$\left(\nabla f(x^{k+1}) - \nabla f(x^k)\right) \approx H\left(x^{k+1}\right)\left(x^{k+1} - x^k\right)$$

This expression is known as the secant equation. Different Quasi-Newton methods have been developed, such as BFGS, DPD and SR1. In BFGS and DPD, the updates are done by a rank-2 matrix, and they guarantee that the updated matrix will be positive definite. In a *symmetric-rank-1* (SR1) the updates are rank-1 updates which will not enforce the positive definiteness of the matrix.

As with the other Quasi-Newton methods, the SR1 makes updates that satisfy the secant equation. The updates to the Hessian (curvature) approximations are shown in Algorithm 6.6. While superseded in many cases by BFGS, SR1 still holds some advantages, according to [69], that are relevant to the type of problem and methods that are being used in Algorithm 6.1:

1. The matrices generated by the SR1 formula tend to be good approximations of the true Hessian matrix (often better than the BFGS approximations).
2. There are simple safeguards that prevent the breakdown of the method and the occurrence of numerical instabilities. By skipping updates when certain conditions are observed, the method will become more reliable and not break down.

In Algorithm 6.6, a condition for skipping updates is shown, and the logic behind it is that if part of the denominator is small or the x iterates are close enough, then the update is stopped. In 6.6, the Hessian approximation is recalculated using all the previous sampled points (in the same active set), but in the order of decreasing distance from the current sample point. This approach is different to the traditional Quasi-Newton framework, where only the last information at the newly sample point

is used to update the existing Hessian approximation. The proposed method can potentially make the approximation more accurate close to the current search area.

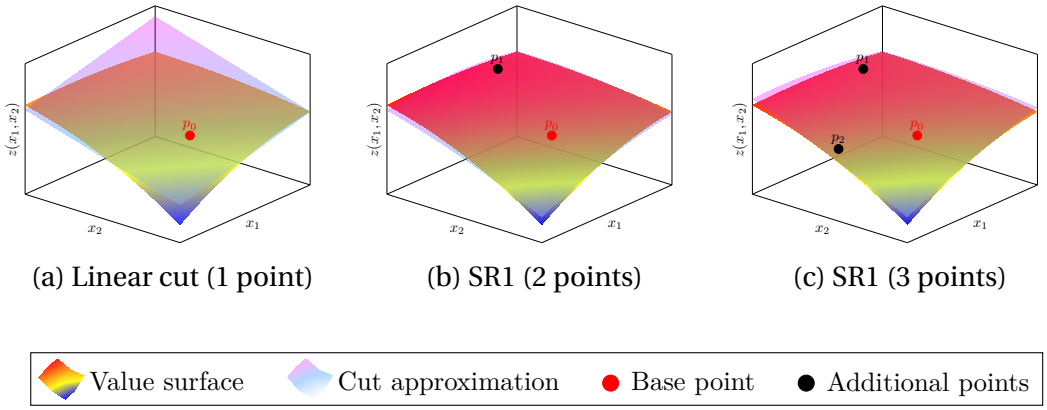
The basic idea is to capture the local curvature (with respect to the current exploration point) in an inexpensive way that prioritises the closest information without forgetting the updates in other directions. In practice, the minimum distance in x is imposed to avoid numeric issues that would overestimate curvature based on numeric tolerances.

Algorithm 6.6 Cut fitting subroutine

Receive: r, a, s, t, x^*_{rst}
Set: $K \leftarrow B^0$
Calculate: norm-2 distance between the current point x^*_{rst} and each of sampled points in the dictionary for the active set $\mathscr{X}_{ra}$:
for all $j \in \mathscr{X}_{ra}$ **do**
    $D^X := \left\| \chi^X_j - x^*_{rst} \right\|_2$
end for
Sort: D^X by decreasing value and define sorted points as $\mathscr{X}^{Ord}$.
Name: the closest point (to the exploration point) as X^C , with its gradient L^C and height Z^C
for all $j \in \mathscr{X}^{Ord} \cap \{X^C\}$ **do**
    $y = L^C - \chi^L_j$
    $r = X^C - \chi^X_j$
    if $\left\| X^C - \chi^X_j \right\|_2 \leq \epsilon^{norm}$ or $|y - K\,r| \leq \epsilon^{SR1}$ **then**
        exit loop
    end if
    $\Delta B = \dfrac{\left(y - K\,r\right)\left(y - K\,r\right)^\top}{\left(y - K\,r\right)^\top r}$
    $K \leftarrow K + \Delta B$
end for
$H^C \leftarrow 1/2\left(K - B^0\right)$
return Z^C, X^C, L^C, H^C

Figure 6.7 shows the process for updating an active set with SR1 updates, based around the closest point in the active set surface (red point) to the closest point proposed by the Master Problem (not shown). In the Figure, the red point is the point of fitting and the more additional points are used, the better the estimation of the curvature of the function. According to the algorithm shown earlier, the first instance

of the cut will be linear (Fig. 6.7a) and then will start fitting estimates of the curvature based on SR1. In this reduced dimension problem, the approximation is good after those 2 points have been sampled. In the figure, note that $\|p_1 - p_0\| > \|p_2 - p_0\|$.



(a) Linear cut (1 point) (b) SR1 (2 points) (c) SR1 (3 points)

Figure 6.7: SR1 fitting

An interesting observation is that SR1 does simple curvature updates for each new point, and depending on the chosen point, the quality of the fitting varies. In Figure 6.8 instead of using first p_1 and then p_2, only p_2 is used. The effect is clear, the gradients match at p_0 and p_2 points, but the curvature is misestimated in the direction perpendicular to the line joining these 2 points. After adding p_1, the resulting figure would look similar (though not identical) to that of Figure 6.7c. As seen in the SR1 algorithm, even if only the last 2 points match in gradient, the historical information that does not contradict the new gradient fitting information will be kept in the curvature matrix.

Figure 6.9 shows a comparison between an SR1 fitting and an exact quadratic model (Figs. 6.9a and 6.9b, respectively) for a near to quadratic function with a small quartic component. Figures 6.9c and 6.9d show the objective value error against the true function value and the fitted curves.

Getting exact second-order information is more expensive than collecting "free" first-order information from the solver. A way of getting the second-order information for fixed variables is shown in [70], and it requires solving n systems of equations (where n is the dimension of the interface) each time that exact curvature at a point for each active set is required. This process, while neat, may not be so valuable as the algorithm only requires a likely optimization direction (coming mainly from the

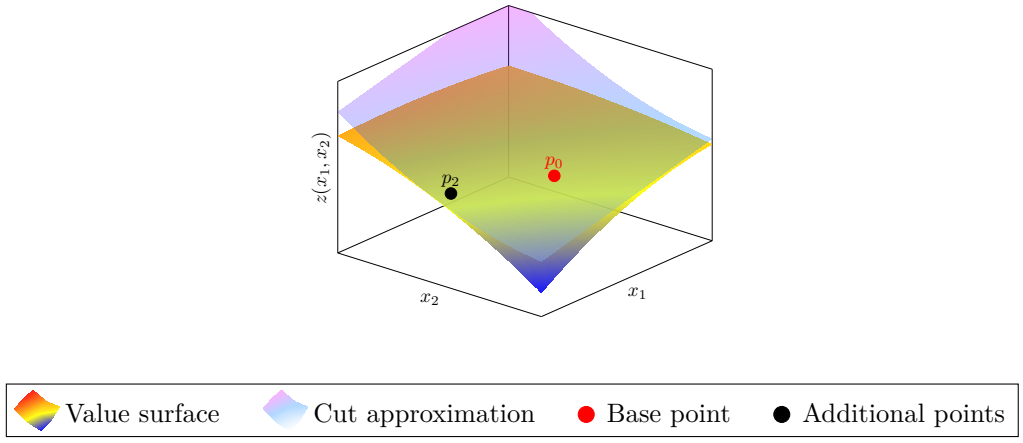
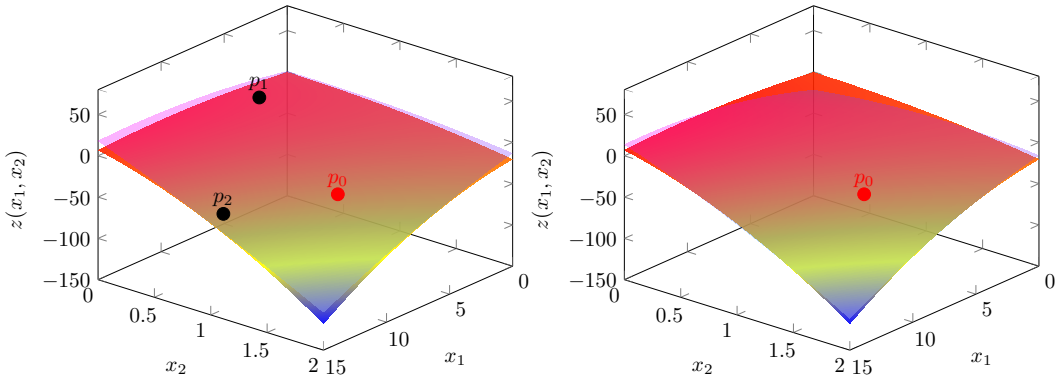
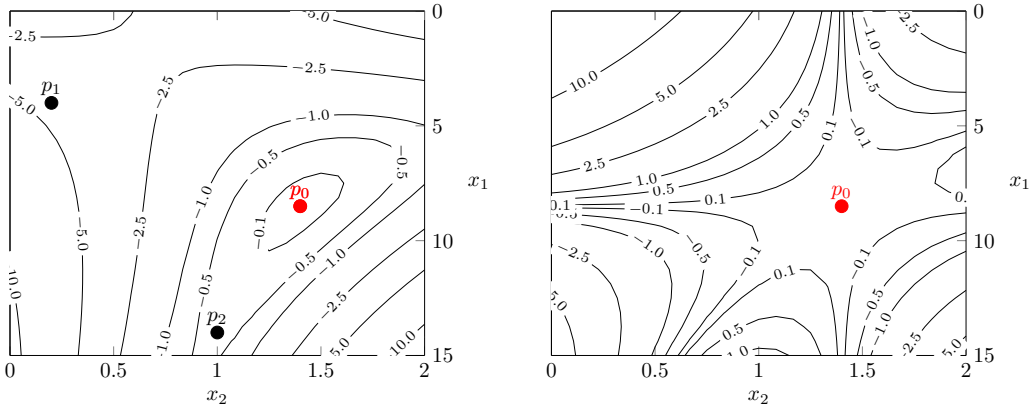


Figure 6.8: SR1 with a different point



(a) SR1 (3 points)

(b) Exact curvature



(c) SR1 curv. vs. true surface

(d) Exact curv. vs. true surface

Figure 6.9: Quadratic model vs. true surface (Exact curvature and SR1 approx.)

gradient) and some sensitivity about the curvature of the algorithm to define the size of the step.

Approximating the curvature by SR1 also fits naturally with the algorithm proposed, as many different points from the same active set are likely to be sampled, especially when sharing cuts. The hope is that the sampled points will be diverse enough, which will help form a good quadratic approximation at the point of interest with the information contained in the dictionary.

6.4 Results

The algorithm presented in this chapter has been tested and benchmarked, and the results are presented in this section. The specific instances being solved are defined in chapter 4 and include a multi-period large network with 986 buses divided into 30 regions (1 transmission network and 29 distribution networks)

Different cases with a diverse number of periods have been tested and solved. First, a single-period problem is introduced where the speed of the decomposition without historical information is tested. Then, a dictionary of samples is built and used to solve other single-period cases, highlighting the value of having this information at the start of the decompositions.

Later in the chapter, multi-period problem results are presented, with fewer periods than the ones presented in Chapter 5, as these are non-linear non-convex and the memory limits are quickly reached.

In the cases presented throughout this chapter, there is no load shedding in the optimal solution; this is not explicitly given but learned during the solution process, i.e. there is a feasible and optimal solution where the generators can supply and deliver the power requested by the demand nodes.

As in the previous chapter, all the algorithms have been coded in Julia 1.5.3 and JuMP 0.21.5, but the solvers used here are IPOPT, KNITRO or Mosek with different optimization techniques, as specified in the results. The examples have all been tested and timed using the same i7-9700K computer with 64GB DDR4-2666 RAM. The CPU has 8 physical cores and 16 threads, and the solutions reported used as many cores as possible by the solver and method selected.

6.4.1 Single-period problem

Problem size

The undecomposed single-period model was presented in Section 6.2. Its size depends on the number of elements in the network: buses, lines, generators and demand profiles as shown in Table 6.1 (Page 99). In the decomposition, the size also depends on the model chosen for the distribution networks (which for the tests in this chapter is a non-convex QP) and on the interface needed for coordination.

In a single-period problem, where the storage level does not change, the interfaces do not need to include storage. Also, the sensitivity around demand profile variables is not needed for isolated single-period problems with no cut sharing between different periods. However, the extra cost of gathering the sensitivity information (for storage use and demand level) is small. In the implementation used for the tests in this chapter, this information is gathered so the same version can be used to share cuts between related subproblems.

In the non-linear case, the model is passed directly to the solver, which only performs a very primitive pre-solve routine. In the pre-solve routine, some variables will be fixed, and only some precompiled bounds will be enforced; thus, the size of the model will not be reduced dramatically by pre-solve (in contrast to the pre-solve done for linear cases presented in Sec. 5.4.1).

Table 6.2 shows the number of variables, constraints and non-zeros for one example of the single-period problem. The table shows the undecomposed model elements compared with the Master Problem elements at the solution, including the cuts but excluding the elements within each subproblem; as expected, the master problem model is significantly smaller than the undecomposed model, since the master problem only includes the transmission network elements plus the interface elements and cuts.

The size difference means that the master problem can be solved much more quickly than the undecomposed problem. As a result, if the number of iterations needed to solve the decomposed problem is small and subproblems are solved fast, the decomposition can be faster than the undecomposed solution.

The sizes of the multi-period problems grow approximately linearly to the number

of periods, and the same is true for the master problem after a fixed number of iterations. However, the size of the subproblems is fixed and does not grow with the number of periods.

Table 6.2: Number of elements in single-period AC-OPF

Periods	Undecomposed			Decomposition (only MP at solution)		
	Variables	Constraints	Non-zeros (Jacobian)	Variables	Constraints	Non-zeros (Jacobian)
1	2.8×10^4	3.8×10^4	1.2×10^5	1.7×10^3	3.1×10^3	1.5×10^4

Solution time

As explained in the motivation section of this chapter, one of the core objectives of the decomposition is the possibility of speeding up the solution of the problem while getting a high quality (feasible) solution with partial information. The convergence tolerance for the single-period examples is set to 0.0001%.

This section presents the results for single-period problems solved using different algorithms and decomposition strategies. Some use an existing dictionary of samples to demonstrate the value of such dictionaries when solving similar problems repeatedly, as in cases of operational problems that have to be solved periodically.

Table 6.3 shows the convergence time for different methods. In the Table, there are five solution strategies compared: (1) the proposed decomposition with a starting dictionary, (2) the decomposition without any starting cuts, (3) undecomposed interior point solved with IPOPT [71] (and MA57 [72] as linear solver), (4) SLQP without any starting points using KNITRO and (5) KNITRO SLQP hot started from the IPOPT optimal solution. For strategies (1) and (2), the master problem is always solved with IPOPT and subproblems with KNITRO.

Without any initial information, the undecomposed interior point strategy performs the best in the 4 cases shown, which is expected as the solver has more information to work with than in the decomposition, which only uses local first-order information and approximate curvatures.

As explained earlier in the thesis, it is important to mention that the subproblems (all distribution networks with a single link to the TN) in the decomposition use the

Table 6.3: Solving time (in seconds) for different single-period problems with different methods

Period	Decomposition		Undecomposed		
	With sample dict.	Without sample dict.	Interior Point	SLQP (cold)	SLQP (hot)[1]
1	**2.5**	6.2	5.6	58.5	16.8
2	**1.6**	6.9	6.5	81.1	17.8
22	**2.7**	11.1	5.5	46.8	34.5
39	**4.3**	11.4	9.1	77.0	18.8

[1] An Interior Point model is solved first, and then the solution point is given to the SLQP solver.

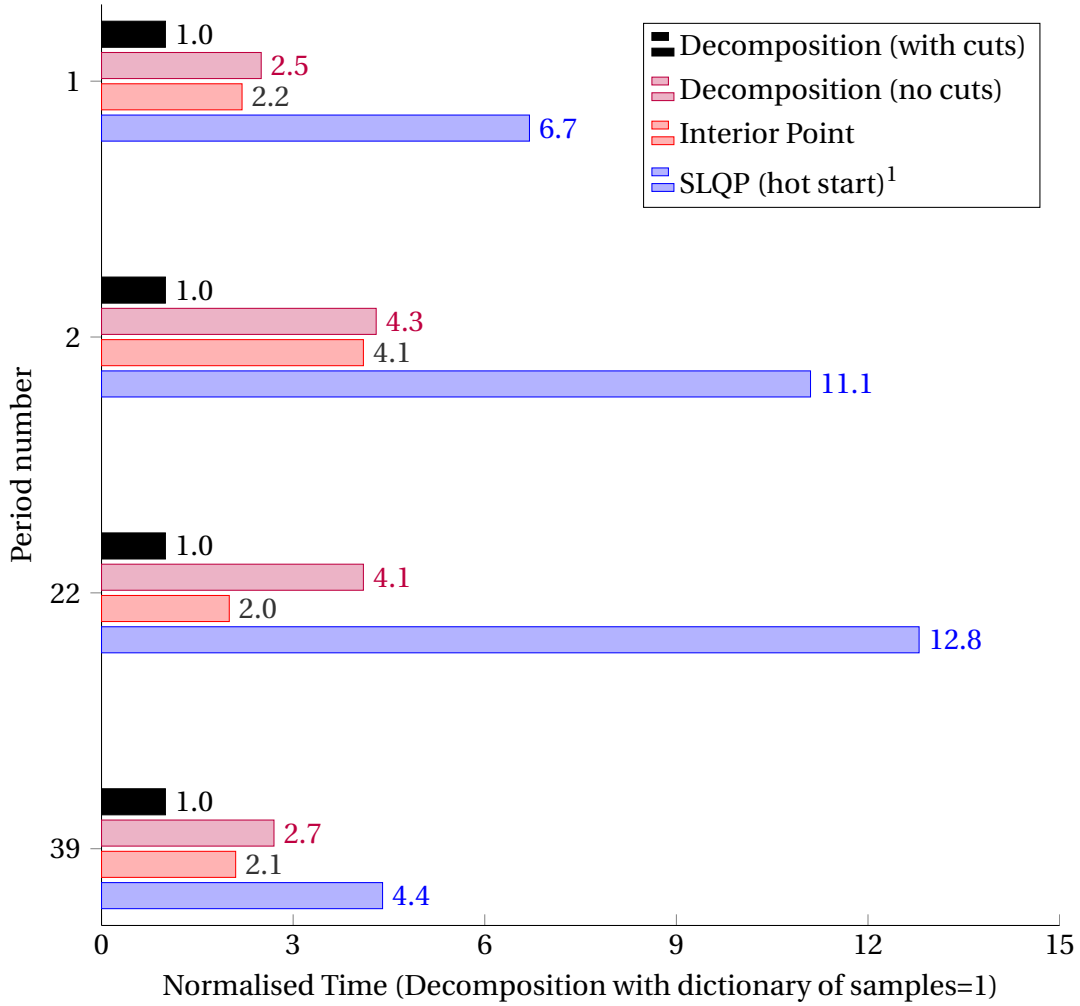


Figure 6.10: Solving time from Table 6.3, normalised time, **Selected single-period problems**

non-convex QP formulation. In contrast, the undecomposed model uses the same power injection polar formulation presented earlier.

When compared to the decomposition using an existing dictionary of samples, the story is different: in every case, this method converged in a small number of

iterations and outperformed all the other methods explored by a significant margin. When the information in the dictionary of samples is relatively close, the convergence is improved significantly; however, if the information is distant, then the cuts generated are likely to need more significant corrections, and the number of iterations and time required is unlike to improve to the same extent. In the results shown in this section, the cuts were generated from a problem with 5% random noise in each of the 87 demand profiles, which is considered close.

SLQP solving strategies

For benchmarking of the test cases, the solution strategies 4 and 5 were solved by SLQP. In SLQP, as described in [73], a sequence of subproblems based on a quadratic model of the original problem is solved; and unlike interior point methods, this algorithm seeks active inequalities and follows a more exterior path to the solution. The SLQP algorithm is similar to SQP, but uses linear programming subproblems to estimate the active set.

The SLQP method works best when a good starting point can be provided. Numerous efforts have been allocated to this end for OPF problems, such as in [74] where the authors use two different strategies to warm-start the problems: (1) solving a linear approximation (with the DC model) and feeding the solution as a starting point, or (2) by solving a SOCP relaxation and estimating the variables not present in the SOCP model (voltage magnitudes and angles). After solving the SOCP relaxation, the voltage magnitude and angles are calculated using a least-squares problem to fit the voltages with the smallest deviations to the generation schedules. In their paper, the best starting points come from this methodology.

In this section, four different starting point strategies have been tested to assess its solution time:

- Cold Start. (No initialisation) Allow the solver to precompute starting bounds and set no initial values for the variables.
- Warm Start 1. Solve a SOCP relaxation, then start the SLQP with SOCP the generation schedule from the SOCP relaxation and set all the voltage magnitudes to 1 p.u. and voltage angles of 0.0.

- Warm Start 2. Solve a SOCP relaxation, then solve the undecomposed problem by interior point, and start the SLQP with the generation schedule and line flows from the SOCP information and the voltage magnitudes and angles from the interior point solution.
- Hot Start. Solve the undecomposed problem by interior point and then use its solution as the starting point for the SLQP.

Strategies "Warm Start 2" and "Hot Start" are not viable solution methods as they require the full problem to be solved by interior point before setting the starting points, but they are presented here to understand the potential time savings from having good starting points.

The "Hot Start" is used as a benchmark of what is achievable by the solver if the starting point had been almost perfect. In practice, the algorithm converges in 1 or 2 internal SLQP iterations, showing that the starting point is (almost) optimal.

In terms of solution time, only the time to solve the SLQP is reported in Table 6.4 for the the 4 starting point alternatives.

Table 6.4: Solving time with diverse starting strategies for the Active Set Method (SLQP)

Period	Cold Start		Warm Start 1		Warm Start 2		Hot Start	
	It.	Time	It.	Time	It.	Time	It.	Time
1	153	58.5	379[1]	124.4	84[1]	59.5	1	16.8
2	172	81.1	273	95.0	219[1]	88.3	1	17.8
22	55	46.8	972[1]	225.1	94[1]	53.6	1	34.5
39	178	77.0	230	67.8	122	93.7	1	18.8

[1] Stopped by KNITRO and reported as suboptimal.

Algorithm progress

The number of iterations required in the decomposition for selected single-period problems is shown in Table 6.5. The four examples presented had significant reductions in iterations when using a starting dictionary of samples.

The progress of the algorithm for a selected single-period problem (Period 2), without the use of a dictionary of samples, is shown in Table 6.6. The algorithm progresses toward convergence as it starts adding cuts in the master problem.

Table 6.5: Number of iterations for convergence of single-period problems

Period	Without sample dict.	With sample dict.	Change
1	15	4	- 73%
2	17	2	- 88%
22	18	5	- 72%
39	17	8	- 53%

The lower bound in iteration 1 will not start from a lower value (such as zero), as the guidance bounds subroutine provides limits to the variables in the interface. The box bounds calculated in the Guidance Bounds subroutine force the master problem to propose points where it needs to generate power in the transmission network, and these costs will be shown in the objective of the master problem. Looking at the optimal objective value in the first iteration, it is clear that the box constraint, while helping to avoid bad cuts, is still very loose in terms of where the algorithm converges.

A reason for this situation is that the guidance bounds do not capture the network dynamics and the correlations between real power, reactive power and voltage, variables which are highly coupled but assumed independent by the guidance bounds subroutine. After 12 iterations, the bounds are relatively close, but some subproblems are still above the convergence tolerance, so the algorithm keeps running. In iteration 15, all the subproblems have converged to the required tolerance, and the cut check subroutine is performed.

After the check, the convergence gap increases outside of the target convergence, so the algorithm keeps running with the newly learned information at the points of iteration 15. In iteration 17 another check is performed, and this time the convergence gap is within the target. In the end, 214 active sets have been learned.

It is worth mentioning that the lower bound in iteration 15 is higher than in iteration 16; this is a clear indicator that the local updates to the cuts have freed space in the master problem and that the cuts were overestimating the true height of the active set.

For Table 6.7, period 2 has been solved, but the results are different to the ones presented in Table 6.6. instead of solving without starting cuts, the cuts from period

Table 6.6: Decomposition for Period 2 **without** a starting dictionary of samples

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Act. Sets
1	27 603.47	83 816 449.40*	8.4×10^7	99.97%	No	29
2	58 029.70	1 070 949 966.17	1.1×10^9	99.99%	No	57
3	166 836.16	237 478 511.92	2.4×10^8	99.93%	No	75
4	217 444.03	116 184 654.56	1.2×10^8	99.81%	No	100
5	362 720.70	42 145 719.81*	4.2×10^7	99.14%	No	126
6	1 023 097.18	31 470 217.90*	3.0×10^7	96.75%	No	145
7	1 070 356.04	5 815 839.16*	4.7×10^6	81.60%	No	167
8	1 298 176.85	16 759 645.66	1.5×10^7	92.25%	No	179
9	1 303 240.18	10 252 738.97	8.9×10^6	87.29%	No	187
10	1 305 025.95	1 877 917.45*	5.7×10^5	30.51%	No	193
11	1 306 474.60	2 528 697.85	1.2×10^6	48.33%	No	196
12	1 306 859.52	1 308 096.70*	1.2×10^3	0.09%	No	199
13	1 307 183.66	1 307 479.03*	3.0×10^2	0.02%	No	207
14	1 307 249.69	1 307 297.19*	4.8×10^1	4E-3%	No	212
15	1 307 263.80	1 307 265.15*	1.3	1E-4%	Yes	214
Check	1 307 262.46			2E-4%		
16	1 307 262.46	1 307 269.04	6.6	5E-4%	No	214
17	1 307 264.29	1 307 264.22*	-7.3×10^{-2}	-6E-6%	Yes	214
Check	1 307 263.83			3E-5%		
Summary	**1 307 263.83**	**1 307 264.22**	$\mathbf{3.9 \times 10^{-1}}$	**3E-5%**		**214**

* Incumbent best upper bound

1 have been added this time. Period 1 is a higher demand period with an objective value of 1.75×10^6 i.e. 35% above period 2, so even if the periods are close in time, the demands are not too close. Also, note that this is a different case from the Period 2 sample dictionary case reported in Table 6.3, in that case, the dictionary was formed from a case where the demands deviated from Period 2 by 5% and its detail is later reported on Table 6.8.

The solution from Period 2 is found in fewer iterations when using the information (222 Active Sets) gathered when solving Period 1. The bounds at Iteration 1 are not too distant from the solution, and the gap is quickly closed at Iteration 3. After only 3 iterations, the problem enters the cut check subroutines, which updates all the active cuts to precise local information.

For illustrative purposes, this example performs the cut checks earlier with a broader gap for the "converged" classification; nonetheless, the algorithm converges fast and gets a remarkably accurate solution; this shows that the algorithm has independent tolerances for cut checks and for stopping the algorithm.

In a situation where the demand levels are similar, e.g. two consecutive weekdays

Table 6.7: Decomposition for Period 2 **with** a starting dictionary of samples (generated at Period 1)

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Act. Sets
1	1 306 971.00	1 343 570.58*	3.7×10^4	2.72%	No	222
2	1 306 939.83	1 397 602.41	9.1×10^4	6.49%	No	222
3	1 307 302.01	1 307 305.06*	3.1	3E-3%	Yes	222
Check	1 307 263.17			3E-3%		
4	1 307 263.83	1 307 263.85*	1.8×10^{-2}	4E-6%	Yes	222
Check	1 307 263.80			4E-6%		
Summary	**1 307 263.80**	**1 307 263.85**	$\mathbf{5.7 \times 10^{-2}}$	**4E-6%**		**222**

* Incumbent best upper bound

at 1 am, the samples from one of the days are likely to be close to the point on the second day if the wind generation did not change significantly, and as a result, the cuts transferred are likely to be good fits for the new demand levels.

For a similar load scenario, each of the 87 profile demands (29 distribution networks with 3 demand profiles in Period 2) was perturbed at random by at most 5%, and the samples were stored in the dictionary during its solution. The problem for Period 2 with the original loads was solved using the cuts from the perturbed instance, and the results are shown in Table 6.8 (A similar process was performed for the other example periods from Table 6.6, but their detail is not shown). As suspected, the problem converges faster, in this example, in only 2 iterations. The bounds are remarkably close to the undecomposed solution.

Table 6.8: Detailed progress using 5% perturbations, for **Period 2**

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Act. Sets
1	1 307 262.46	1 307 269.03*	6.6	5E-4%	No	216
2	1 307 264.59	1 307 264.51*	-8.4×10^{-2}	-6E-6%	Yes	216
Check	1 307 263.83			5E-5%		
Summary	**1 307 263.83**	**1 307 264.51**	$\mathbf{6.8 \times 10^{-1}}$	**5E-5%**		**216**

* Incumbent best upper bound

The newly sampled subproblems update the cuts with true local information, then the cut check corrects the overestimations of the upper bound, and the problem converges.

An interesting case happens when imposing the cuts generated at Period 2 to the demand profile levels of Period 39; Period 39 has an objective value more than 4

Table 6.9: Decomposition **with** sample dictionary (from Period 2) for the single-period problem, **Period 39**

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Act. Sets
1	6 416 968.95	5 278 819 792.81*	5.3×10^{9}	99.88%	No	214
2	5 393 153.11	194 112 766.56*	1.9×10^{8}	97.22%	No	228
3	5 393 977.13	52 211 286.57*	4.7×10^{7}	89.67%	No	239
4	5 397 440.91	8 363 001.34*	3.0×10^{6}	35.46%	No	250
5	5 399 120.37	5 547 936.93*	1.5×10^{5}	2.68%	No	256
6	5 443 855.62	5 999 151.43	5.6×10^{5}	9.26%	No	264
7	5 450 241.58	5 461 340.71*	1.1×10^{4}	0.20%	No	270
8	5 450 253.52	5 450 539.02*	2.9×10^{2}	0.01%	No	272
9	5 450 381.56	5 450 396.01*	1.4×10^{1}	3E-4%	No	272
10	5 450 389.63	5 450 389.56*	-7.2×10^{-2}	-1E-6%	Yes	273
Check	5 438 551.47			0.22%		
11	5 438 561.53	5 456 870.94	1.8×10^{4}	0.34%	No	273
12	5 438 568.39	5 438 575.26*	6.9	1E-4%	No	274
13	5 438 572.56	5 438 576.24	3.7	7E-5%	Yes	274
Check	5 438 563.42			2E-4%		
14	5 438 564.19	5 438 564.18*	-8.0×10^{-3}	-1E-7%	Yes	275
Check	5 438 563.41			1E-5%		
Summary	**5 438 563.41**	**5 438 564.18**	$\mathbf{7.7 \times 10^{-1}}$	**1E-5%**		**275**

* Incumbent best upper bound

times higher than Period 2, so the samples contained in the dictionary correspond to a far away explored area. In Table 6.9 the progress for this case is shown.

Given that the samples in the dictionary are far away demands for Period 39, only linear cuts have been added in the first iteration because the approximated curvature is not helpful as it does not include local information. In Iteration 1, the cuts grossly overestimate the true lower bound, illustrating the non-convexity of the problem.

The problem then runs until Iteration 10, where the first cut validity routine shows a significant overestimation of the lower bound. These cuts are updated with accurate local information, and the master problem becomes loose again.

After two additional cut checks, in Iteration 14, the problem has converged to a feasible nearly-optimal point. In this case, the problem converges after 14 iterations, and if the problem had not been started with these cuts, the problem would have converged in 17 iterations; making clear here that the addition of distant information is not very useful and could complicate and even slow down the solution process.

Solution Quality

The single-period problems presented in this section have been pushed to tight gaps of convergence (0.0001%); they are easier problems to solve than the multi-period problems not only given their smaller size but also because the circularity constraint on storage level does not allow for any storage to be used. This fact benefits both the decomposed and the undecomposed problems, so the comparison is equivalent.

Without storage, the subproblems have reduced flexibility, i.e. once the interface is fixed by the master problem (for p, q and v), the subproblem feasible response is determined by the local generator and the reactive support devices only. However, the subproblem will look for a solution that minimizes its cost.

In Table 6.10 are compared the results of the upper and lower bounds for the decomposition to what the solver in the undecomposed problem claimed as optimal. This table shows that the single-period problems converged to the optimal point, and even if there are small overestimations for the lower bounds, they are in the range of numerical noise.

Table 6.10: Comparison between optimal for the undecomposed vs. decomposition, single-period problems

Without starting dictionary of samples

Period	Undec. solution	LB	Gap		UB	Gap	
			Abs.	Rel.		Abs.	Rel.
1	1 746 741.81	1 746 741.60	–0.21	-1E-5%	1 746 742.77	0.96	6E-5%
2	1 307 263.82	1 307 263.83	0.01	9E-7%	1 307 264.22	0.40	3E-5%
22	3 323 436.35	3 323 436.32	–0.03	-9E-7%	3 323 436.38	0.03	8E-7%
39	5 438 563.30	5 438 563.00	–0.29	-5E-6%	5 438 567.87	4.57	8E-5%

With starting dictionary of samples

Period	Undec. solution	LB	Gap		UB	Gap	
			Abs.	Rel.		Abs.	Rel.
1	1 746 741.81	1 746 741.74	–0.07	-4E-6%	1 746 741.83	0.02	1E-6%
2	1 307 263.82	1 307 263.83	0.01	1E-6%	1 307 264.51	0.69	5E-5%
22	3 323 436.35	3 323 436.33	–0.03	-9E-7%	3 323 436.38	0.02	6E-7%
39	5 438 563.30	5 438 563.31	0.02	3E-7%	5 438 567.08	3.79	7E-5%

6.4.2 Multi-period problem

Problem size

The problem size for a multi-period undecomposed AC-OPF problem is similar to the one presented in Subsection 6.4.1 but increased linearly depending on the number of periods solved. For a 2-period problem is almost twice the number of variables, constraints, and non-zeros as a single-period problem.

This section decomposes a multi-period AC-OPF problem for the transmission network (embedded in the master problem) and time-linked storage variables for each distribution network. The size of the master problem scales up linearly with the number of periods; the distribution network subproblems, however, are solved as single-period problems. The subproblem size is unaffected, but the number of subproblems to solve in each iteration increases linearly with the number of periods.

The results show that the proposed decomposition can solve larger instances than the undecomposed version. The decomposition also allows for the problems to be solved separately. e.g. the master problem and the subproblems do not even need to be located in the same machine.

Suppose the size of the interface remains the same. In that case, a much more complex distribution network will not affect the master problems capabilities to be solved apart from the possible increase in the number of cuts added to the master problem.

Table 6.11 shows the number of elements for a given number of periods; the decomposition part shows only the number of elements in the master problem at the solution (i.e. including cuts).

Table 6.11: Number of elements in multi-period AC-OPF cases

	Undecomposed			Decomposition (only MP at solution)		
Periods	Variables	Constraints	Non-zeros (Jacobian)	Variables	Constraints	Non-zeros (Jacobian)
4	1.1×10^5	1.5×10^5	4.6×10^5	6.7×10^3	1.2×10^4	3.7×10^4
8	2.3×10^5	3.0×10^5	9.2×10^5	1.3×10^4	2.4×10^4	7.4×10^4
24	6.8×10^5	9.1×10^5	2.8×10^6	4.0×10^4	7.4×10^4	2.3×10^5
48	1.4×10^6	1.8×10^6	5.5×10^6	8.0×10^4	1.5×10^5	5.1×10^5
96	2.7×10^6	3.6×10^6	1.1×10^7	1.6×10^5	3.0×10^5	1.0×10^6

Solution time

In the multi-period case, a feasible solution is required. However, unlike single-period problems, the need to converge to such high tolerances (as those in single-period problems from Section 6.4.1) is not so demanding. Some uncertainties affect the objective more than the high precision in single deterministic scenarios in real systems.

A clear example happens when converging a multi-period deterministic problem that solves 24 hours to a high tolerance. The "high" precision solution becomes less relevant 30 minutes later when the wind and demand forecasts are updated. However, it is valuable to have a clear idea of the system's evolution in the following hours to detect the general working patterns of the network. Solving an immediate (single) period problem to high precision and fixing its outputs for a more extended time-scale model in a rolling horizon fashion may be more valuable than having tight convergence tolerances.

Based on this logic, the convergence tolerance for multi-period problems has been relaxed, up to 0.05%. The algorithm can cope with tighter tolerances at the expense of numerous iterations, which without any specific reason, are not required. In practice, the convergence of the multi-period problems in many cases reached tolerances of 0.01% because the stopping condition of Algorithm 6.1 refers to convergence bounds for each subproblem instead of a global convergence tolerance.

In this section, the results for different length multi-period problems are presented using the proposed decomposition strategy with and without starting information from a dictionary of samples, that can be gathered in similar problems that have been historically solved. In the results presented, only a small dictionary (obtained when solving a similar problem with a 5% perturbation) is used.

Table 6.12 shows the convergence time for different methods for multi-period problems.

As shown in Figure 6.11, if a feasible result within 0.05% is required, the decomposition provided this faster than the undecomposed problem. In the decomposition, the 96-period instance provided significant challenges to the solver, as can be seen by the non-linear relation between the size of the problem and the solving time; the

Table 6.12: Solving time (in seconds) for different multi-period problems with different methods

Periods	Decomposition		Undecomposed		
(length)	With sample dict.[1]	Without sample dict.[1]	Interior Point	SLQP (cold) [2]	SLQP (hot)[3]
39-42 (4)	**19.8**	33.9	36.1	DNC	1245.4
32-39 (8)	**55.9**	69.7	86.3	DNC	3836.8
33-48 (16)	**138.9**	214.5	530.6	DNC	8695.2
25-48 (24)	**308.9**	350.7	989.4	DNC	18 849.1
1-48 (48)	**1090.6**	1711.5	7377.1	DNC	DNC
1-96 (96)	**5077.4**	5322.2	NEM[4]	NEM[4]	NEM[4]

[1] Converged to a feasible soultion within 0.05% of the optimal.
[2] Did not converge (DNC) in 6 hours (21 600s).
[3] An Interior Point model is solved first, and then the solution point is given to the SLQP solver.
[4] Not Enough Memory (NEM), reported by the solver.

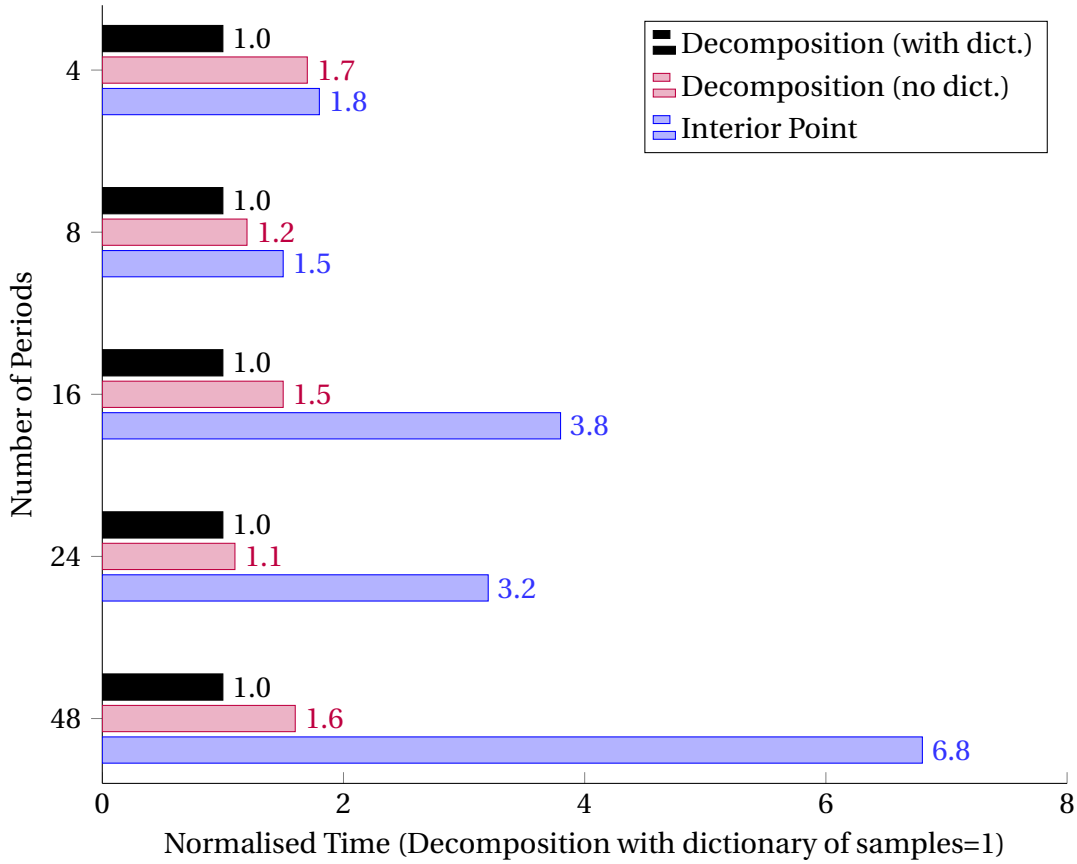


Figure 6.11: Solving time from Table 6.12, normalised time, **Selected multi-period problems**

undecomposed problem did not even fit into memory.

Table 6.13 shows the percentage of time spent in different routines of the algorithm. In the small instances, the guidance bounds subroutine and the active set cut check take more (relative) time, so they are proportionally more expensive. Similar

to the undecomposed problem, when the master problem grows in the number of periods, there is a non-linear increase in solving time in the decomposition.

Table 6.13: Elapsed time (s) in different routines, without a starting dictionary of samples

Periods	Guidance bounds	Master problem	Subproblems	Active cut check	Total
4	3.6	18.9	9.7	1.6	33.9
8	7.1	47.2	11.7	4.0	69.7
16	13.8	167.2	23.2	10.3	214.5
24	18.2	286.7	30.2	15.6	350.7
48	30.7	1556.1	72.6	52.1	1711.5
96	45.2	5022.5	161.1	93.5	5322.2

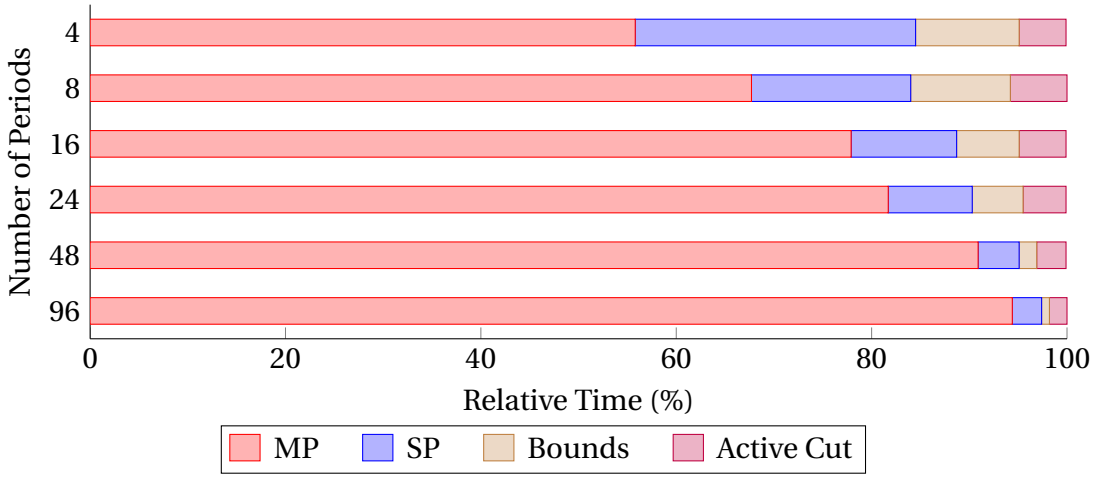


Figure 6.12: Relative time elapsed in different routines from Table 6.13

Algorithm progress

As with the single-period problem, a fundamental step to start the decomposition is Algorithm 6.2 (guidance bounds); this subroutine avoids sampling the problem in irrelevant areas unlikely to lead to where the optimal solution lies. As illustrated in Figure 6.4d if the value surface is not convex in these areas, then a linear cut formed there can mislead the master problem. In both single-period problems and multi-period with fixed storage policies, the guiding bounds can be tightened depending on the use of the storage for that period. In the multi-period problems solved here, there is no indication *a priori* if a store is charging or discharging in a period, so the bounds are widened to account for the storage capabilities. The change makes the

Table 6.14: Number of iterations for convergence of multi-period problems

Number of periods	Without sample dict.	With sample dict.	Change
4	13	5	- 53%
8	10	7	- 30%
16	14	7	- 50%
24	17	10	- 40%
48	17	9	- 47%
96	17	12	- 24%

guidance bounds looser, as seen in the algorithm progress for the lower bound of the first iteration. If a dictionary of samples is used to add starting cuts, the lower bound in Iteration 1 is likely to be higher as the cuts may have better information than the guidance bounds subroutine.

While the number of iterations required for convergence is shown in Table 6.14. The cases which started with cuts from the dictionary of samples have significant reductions in iterations. Nevertheless, the time for convergence was not reduced in the same ratio, which is particularly evident in the 24 and 96-period cases. In both cases, the first iteration of the master problem took at least four times longer than in the cases starting without cuts. The master problem is challenging in the first iteration because it contains numerous cuts and no starting point given to the solver. In contrast with the single-period case (sec. 6.4.1), the benefit of a dictionary is not as evident in the multi-period case.

The progress of a 16-period problem is shown in Tables 6.15 and 6.16, without starting cuts and with starting cuts from the sample dictionary, respectively.

The other multi-period problems tested had a similar behaviour. After finding the right area, the algorithm quickly finds a feasible point within the target gap. After the cuts are verified and the guidance bounds removed, the algorithm stops.

In the 48-period problem, without a starting sample dictionary, all the subproblems converged in 15 iterations, but when the cuts were verified the convergence gap increased, so additional iterations were required. The progress of the algorithm is presented in Table 6.17.

The 48-period problem starting with a dictionary of samples converges faster, closing the convergence gap earlier, especially in the first iterations. Even if the cuts

Table 6.15: Decomposition for the 16-period case **without** a starting dictionary of samples

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Cuts	Act. Sets
1	1 421 557.20	186 458 429.64*	1.9×10^8	99.24%	No	0	81
2	2 798 398.40	173 428 820.08*	1.7×10^8	98.39%	No	1296	157
3	3 422 556.56	25 856 110.08*	2.2×10^7	86.76%	No	2512	199
4	3 800 035.16	10 695 287.20*	6.9×10^6	64.47%	No	3184	214
5	3 990 467.22	6 635 018.64*	2.6×10^6	39.86%	No	3424	227
6	4 001 748.98	4 353 517.59*	3.5×10^5	8.08%	No	3632	232
7	4 003 743.64	4 015 435.25*	1.2×10^4	0.29%	No	3712	232
8	4 004 545.12	4 005 707.25*	1.2×10^3	0.03%	No	3712	234
9	4 004 573.07	4 005 673.92*	1.1×10^3	0.03%	No	3744	234
10	4 004 574.65	4 005 452.36*	8.8×10^2	0.02%	No	3744	234
11	4 004 572.62	4 005 269.37*	7.0×10^2	0.02%	No	3744	234
12	4 004 570.94	4 005 135.29*	5.6×10^2	0.01%	No	3744	235
13	4 004 568.93	4 005 022.07*	4.5×10^2	0.01%	No	3760	236
14	4 004 574.44	4 004 808.70*	2.3×10^2	0.01%	Yes	3776	240
Check	4 004 538.96			0.01%			
Summary	**4 004 538.96**	**4 004 808.70**	$\mathbf{2.7 \times 10^2}$	**0.01%**	**Yes**	**3840**	**240**

* Incumbent best upper bound

Table 6.16: Decomposition for the 16-period case **with** a starting dictionary of samples

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Cuts	Act. Sets
1	3 986 812.17	60 469 393.96*	5.6×10^7	93.41%	No	3824	273
2	4 004 601.16	4 927 326.68*	9.2×10^5	18.73%	No	4368	284
3	4 003 659.30	4 026 082.16*	2.2×10^4	0.56%	No	4544	287
4	4 004 364.72	4 017 235.40*	1.3×10^4	0.32%	No	4592	288
5	4 004 496.70	4 016 658.97*	1.2×10^4	0.30%	No	4608	289
6	4 004 554.82	4 005 472.98*	9.2×10^2	0.02%	No	4624	290
7	4 004 572.26	4 005 238.33*	6.7×10^2	0.02%	Yes	4640	292
Check	4 004 523.12			0.02%			
Summary	**4 004 523.12**	**4 005 238.33**	$\mathbf{7.2 \times 10^2}$	**0.02%**	**Yes**	**4672**	**292**

* Incumbent best upper bound

come from a dictionary of samples (one 48-period 5% perturbed instance), the lower bound is very close to the optimal from the first period; the algorithm then spends the time refining its view to the value surface with the newly sampled points.

Solution quality

As explained at the beginning of this section, there is more interest in getting a good feasible solution in multi-period cases rather than spending time finding the exact optimal. For this purpose, the convergence gap has been relaxed to 0.05%, and the cut validity tolerances increased, allowing slightly larger cut violations. Therefore,

Table 6.17: Decomposition for the 48-period case **without** a starting dictionary of samples

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Cuts	Act. Sets
1	505 083.12	120 623 050.39*	1.2×10^8	99.58%	No	0	132
2	1 513 066.23	9 683 829.72*	8.2×10^6	84.38%	No	6336	201
3	2 444 538.05	153 336 545.00	1.5×10^8	98.41%	No	9648	245
4	2 460 550.51	6 775 096.12*	4.3×10^6	63.68%	No	11 760	264
5	2 471 000.52	2 723 691.88*	2.5×10^5	9.28%	No	12 672	272
6	2 474 811.80	2 504 000.90*	2.9×10^4	1.17%	No	13 056	276
7	2 475 561.90	2 488 898.81*	1.3×10^4	0.54%	No	13 248	279
8	2 475 698.50	2 485 893.52*	1.0×10^4	0.41%	No	13 392	281
9	2 475 688.48	2 481 221.49*	5.5×10^3	0.22%	No	13 488	282
10	2 475 670.40	2 477 709.67*	2.0×10^3	0.08%	No	13 536	286
11	2 475 656.71	2 479 639.07	4.0×10^3	0.16%	No	13 728	287
12	2 475 652.36	2 477 101.84*	1.4×10^3	0.06%	No	13 776	289
13	2 475 652.20	2 478 666.58	3.0×10^3	0.12%	No	13 872	292
14	2 475 651.83	2 476 428.03*	7.8×10^2	0.03%	No	14 016	292
15	2 475 650.74	2 475 739.73*	8.9×10^1	4E-3%	Yes	14 016	292
Check	2 473 654.67			0.08%			
16	2 473 901.43	2 484 809.24	1.1×10^4	0.44%	No	14 064	296
17	2 474 815.78	2 475 340.11*	5.2×10^2	0.02%	Yes	14 208	296
Check	2 474 509.66			0.03%			
Summary	**2 474 509.66**	**2 475 340.11**	$\mathbf{8.3 \times 10^2}$	**0.03%**	**Yes**	**14 208**	**296**

* Incumbent best upper bound

Table 6.18: Decomposition for the 48 period case **with** a starting dictionary of samples (extract)

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Cuts	Act. Sets
1	2 463 125.16	67 795 537.90*	6.5×10^7	96.37%	No	14 880	338
2	2 477 040.65	12 520 252.13*	1.0×10^7	80.22%	No	16 224	344
3	2 476 479.04	2 717 126.96*	2.4×10^5	8.86%	No	16 512	345
4	2 476 109.36	2 478 571.61*	2.5×10^3	0.10%	No	16 560	347
5	2 475 887.02	2 477 243.39*	1.4×10^3	0.05%	No	16 656	348
6	2 475 754.10	2 476 892.10*	1.1×10^3	0.05%	No	16 704	348
7	2 475 672.56	2 478 329.41	2.7×10^3	0.11%	No	16 704	349
8	2 475 622.93	2 476 963.46	1.3×10^3	0.05%	No	16 752	349
9	2 475 591.78	2 475 650.47*	5.9×10^1	2E-3%	Yes	16 752	349
Check	2 474 648.19			0.04%			
Summary	**2 474 648.19**	**2 475 650.47**	$\mathbf{1.0 \times 10^3}$	**0.04%**	**Yes**	**16 752**	**349**

the solution's quality is lower than in the single-period case but still much higher than probably needed in practical terms.

Table 6.19 compares the bounds obtained with the decomposition to the optimal value reported by the solver for the undecomposed problem. The negative differences show cases where the lower bound in the decomposition is still below the optimal objective; therefore, they do not show any violations at the exploration point.

Table 6.19: Comparison between optimal for the undecomposed vs. decomposition, multi-period problems

Without starting dictionary of samples

Number of periods	Undec. solution	LB	Gap Abs.	Gap Rel.	UB	Gap Abs.	Gap Rel.
4	4 844 341.5	4 844 093.1	−248.4	−0.005%	4 844 579.8	238.3	0.005%
8	4 497 852.5	4 497 096.8	−755.7	−0.017%	4 498 368.1	515.6	0.011%
16	4 004 536.6	4 004 539.0	2.4	6E-5%	4 004 808.7	272.1	0.007%
24	3 469 775.1	3 469 769.0	−6.1	-2E-4%	3 469 883.2	108.1	0.003%
48	2 474 868.6	2 474 509.7	−358.9	−0.015%	2 475 340.1	471.5	0.019%

With starting dictionary of samples

Number of periods	Undec. solution	LB	Gap Abs.	Gap Rel.	UB	Gap Abs.	Gap Rel.
4	4 844 341.5	4 844 277.9	−63.6	−0.001%	4 844 709.1	367.6	0.008%
8	4 497 852.5	4 497 771.2	−81.3	−0.002%	4 498 467.7	615.2	0.014%
16	4 004 536.6	4 004 523.1	−13.5	-3E-4%	4 005 238.3	701.7	0.018%
24	3 469 775.1	3 469 778.3	3.2	9E-5%	3 470 045.7	270.6	0.008%
48	2 474 868.6	2 474 648.2	−220.4	−0.009%	2 475 650.5	781.9	0.032%

On the other hand, the positive differences show that even though the problem has verification routines to detect violations, there are cuts that overestimate the minimum of the problem. The lower bound precision requirements and the importance of getting a good upper bound (feasible) solution in a reduced amount of time will dictate the convergence an verification criteria, balancing speed and precision.

Table 6.20 shows additional iterations of the case presented in Table 6.15 when it is forced to continue running requiring a tighter gap for stopping. The new point at Iteration 16 has a better lower bound (below the true optimal from Table 6.19) and a better upper bound.

Table 6.20: Additional iterations for the 16-period case w/o dict. (cont. of Table 6.15)

Iter.	LB	UB	Gap (Abs)	Gap (Rel)
15	4 004 538.96	4 004 808.70*	2.7×10^2	0.01%
16	4 004 532.43	4 004 629.29*	9.7×10^1	2E-3%
Check	4 004 531.35			
Summary	**4 004 531.35**	**4 004 629.29**	$\mathbf{9.8 \times 10^1}$	**2E-3%**

* Incumbent best upper bound

The cut validity routine performed at the last step fixes the active sets of the active cuts and evaluates them to get precise information at the point of convergence; once the sample dictionary has been augmented with precise local information, the

master problem is re-optimized with the updated versions of the cuts. If the master problem finds a new point near the previous point, the cuts are likely to be accurate there; however, if it moves a great distance, the cuts may be imprecise at the new location.

If a guaranteed valid lower bound is needed, a relaxation can be used, such as the SOCP relaxation previously presented. In this relaxation, the bound tightness will depend on the "stress" to which the network is subjected, as the SOCP fails to capture the true nature of the meshed power flow. Nevertheless, the SOCP relaxation can be helpful because it is fast to compute and may help stop the decomposition earlier if certainty about a prescribed target gap has been reached with this bound and the upper bound of the decomposition.

With the SOCP model presented earlier and using Mosek [75] as the conic solver, the relaxation can be solved efficiently; Table 6.21 shows the performance in terms of elapsed time and the slackness of the lower bound for the instances presented in this chapter. For the test cases here presented, the bound provided by the SOCP relaxation is too loose, and it would only be helpful if the target convergence tolerance is around 1%.

Table 6.21: Solving time (s) and the bound given by the SOCP relaxation

Number of periods	Undec. solution	SOCP Relaxation	Gap Abs.	Gap Rel.	Time
4	4 844 341.5	4 818 867.1	–25 474.4	-0.5 %	0.8
8	4 497 852.5	4 472 668.8	–25 183.7	-0.6 %	1.9
16	4 004 536.6	3 972 990.6	–31 546.0	-0.8 %	4.8
24	3 469 775.1	3 436 147.9	–33 627.2	-1.0 %	8.0
48	2 474 868.6	2 457 508.0	–17 360.6	-0.7 %	26.9

6.4.3 Practical implementation issues

The implementation of algorithm 6.1 presents some critical practical challenges (not found in the linear version of the algorithm) that need to be addressed for it to work. The implementation requires special attention in detecting the active set correctly, filtering dual noise, and evaluating the active set height robustly and quickly.

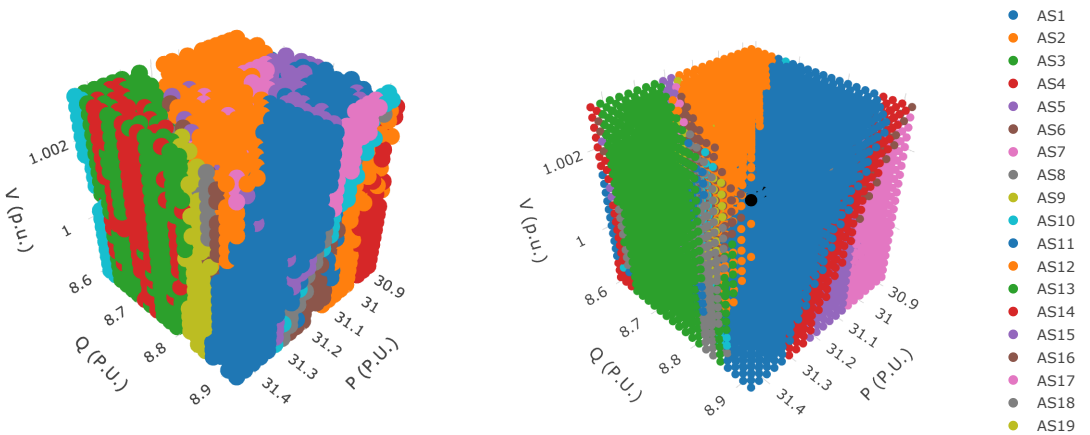
Active set classification

Correctly identifying the active set is fundamental. First, mistakenly classifying two different active sets as a single active set will result in a missing cut; this can also have de-stabilising effects since the first-order information is based only on the closest previously sampled point to the exploration point. If this closest sampled point jumps from one of the two active sets to the other during iterations, then inconsistent first-order information is provided to the master problem, which may prevent convergence. The second important consequence is that if an approximate curvature routine, such as SR1, is used, the curvature estimate is likely to be much higher than the actual curvature of either active set and will not be consistent between iterations, and this may also cause the algorithm to fail to converge. Conversely, if two samples in the same active set are not correctly identified as belonging to the same set, there will be redundant cuts; this does not cause convergence problems but results in unnecessarily large master problems.

Estimating the active set from a solution given by an Interior Point method is challenging, especially where points are close to the boundary between active sets. In these circumstances, the numeric noise given by the barrier parameter (even if reduced in the last interior point iterations) contaminates the duals, and then Algorithm 6.3 can produce false groupings.

Because of these difficulties with using an interior point method to detect the active set, the algorithm for solving the subproblems was set to SLQP (also known as "Active Set" in the KNITRO solver)to produce cleaner active sets. In the KNITRO solver, there are parameters to adjust the precision of the active set calculation, and one of those, *"act_lpfeastol"*, has a great impact on the success of this grouping. In Figure 6.13 a subproblem was selected and explored in a box range (in the p, q and v directions) and the reported active sets are shown for 2 different "act_lpfeastol" values.

When comparing both figures, Figure 6.13a shows more contamination (i.e. a mix of detected active sets) than the one with tighter tolerances; while in these images is not possible to look at the internal points, when explored, they also show much cleaner active set divisions with the tighter tolerance.



(a) Looser Active Set tolerance

(b) Tighter Active Set tolerance

Figure 6.13: Classification of Active Sets for different KNITRO Active Set solver tolerances

Figure 6.13 only shows the classification by active sets of these points, but it does not show the function objective value or the dual values for the fixing constraints. In Figures 6.14 and 6.15, another subproblem was selected and explored in a random direction (the same in each figure), gathering its dual and objective values while classifying its active sets. Note that the active set names and colours in these figures are independent of each other. Figure 6.14 shows the results using a low (1×10^{-6}) KNITRO active set precision. Figure 6.15 shows the results with a high precision (1×10^{-10}).

The four graphs in each figure show the following in the y-axis: (upper-left) dual values on the constraint fixing real power at the interface, (upper-right) dual values on the constraint fixing reactive power, (lower-left) dual value on the voltage constraint and finally, (lower-right) showing the objective value; on the x-axis, the real power component of the line is used as it is the main driver for changes to the objective value. Note that the x-axis (real power) range shown in these figures is very small around the point where the active sets change.

Figure 6.14 has not only looser tolerances for the active set estimation in the solver but also has different feasibility and optimality requirements. The jumps along the exploration line (mainly green points) show that the reported optimal may lie randomly in a different active set depending on the stopping tolerance. With tighter tolerances, Figure 6.15 shows clean and coherent active sets.

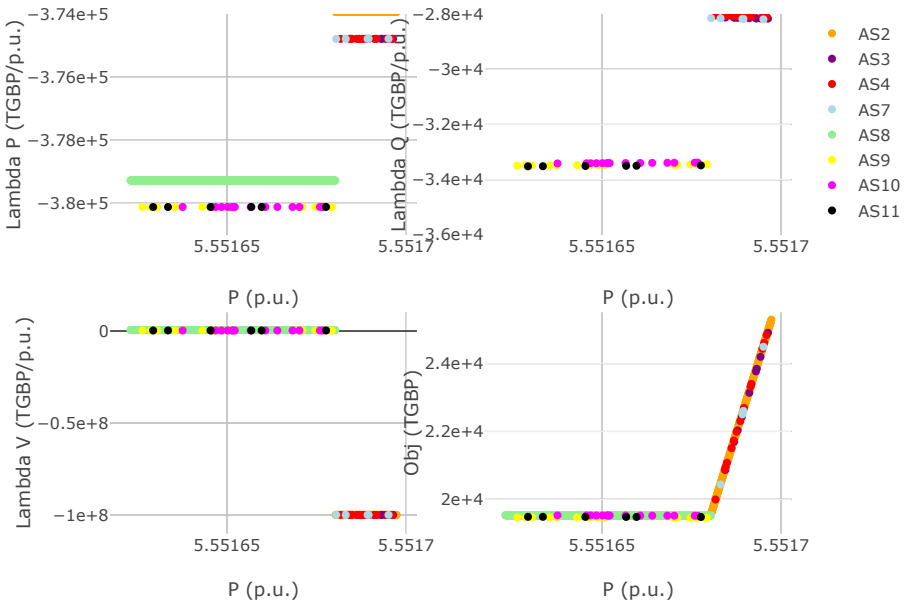


Figure 6.14: Objective and dual values under **low precision** tolerances for a random line

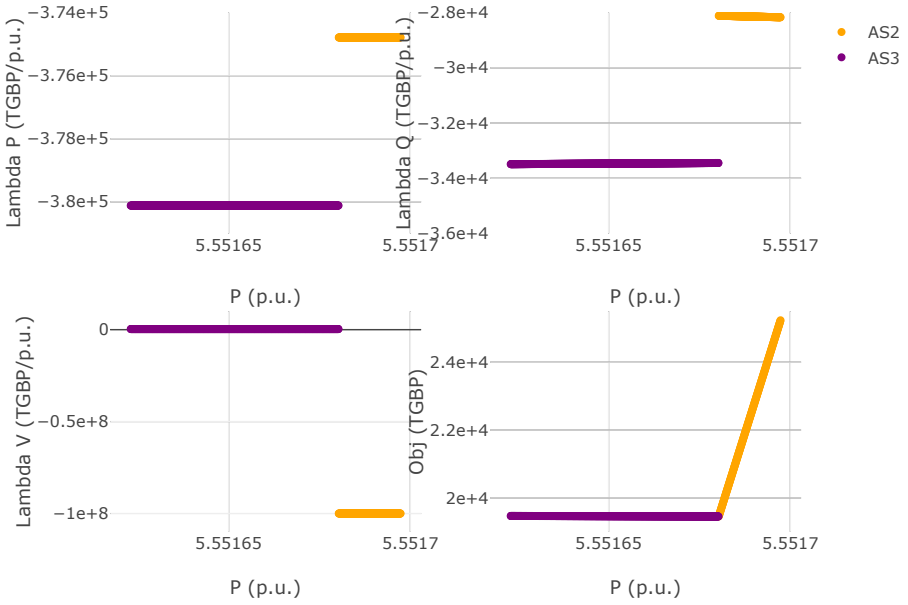


Figure 6.15: Objective and dual values under **high precision** tolerances for a random line

Another important observation from these graphs is that they all look to be piecewise constant. The constant gradients appear to be showing that there is no significant gradient change along that line. In this scale, the dual on reactive

power (upper right subfigure) is the only one showing a slightly slanted non-linear component on the $AS2$. In fact, the duals are not constant, but they change slightly. The change is primarily linear, so a quadratic cut would fit them well over this range of P values, and higher-order components may not be so relevant; a quadratic approximation fits well most of the active sets found close to the feasible areas of the subproblems.

The need for a high precision AS solution has been presented in this section. To achieve it in practice in a reasonable time, the following process was implemented: first, the subproblem is solved using KNITRO's interior point algorithm with crossover (forcing Active Set method iterations after convergence), and then once the solution has been reached, the same subproblem is solved again by SLQP, starting from a point close to the interior point solution. While this step seems unnecessary given that interior point performs a crossover, in practice, the results obtained by this routine did not have the precision required and could not be tuned, so they presented similar problems to the ones shown in this section. The solution point found by interior point method needs to be perturbed when passed as a starting point for the SLQP to force at least 1 iteration; if this is not the case, even if the problem can verify that the solution is optimal, it will not pass the dual values to the programming package (JuMP), which is needed for the cuts to be defined.

Although this seems like a convoluted solution process, it is faster to get a high precision solution than solving the SLQP without a starting point.

Noisy dual values

In Subsection 6.3.8 and, in particular, in Algorithm 6.6 has been mentioned that sampled points having the same active set are sorted by distance, and the SR1 updates are performed starting with those that are further away and finishing with those that are closer to the current exploration point. As mentioned earlier, a minimum required distance is imposed so that numeric noise does not disturb the curvature fitting significantly. If an already explored point is closer to the current point than this minimum distance, then no SR1 update is performed.

In Figure 6.16, a zoomed-in version of the dual values for a real power fixing

constraint is plotted in a very small range of power (600W variation out of 555MW). Even if the points are correctly identified as belonging to the same active set, the solver returned noisy duals for the section on the left part of the graph. If an update between adjacent points in the left part is performed, it is likely to lead to a highly inaccurate curvature estimate. The minimum distance threshold tries to dampen this situation by preventing updates using points that are too close; even if the approximation of the curvature does not use all the available points, the effects of the loss in precision due to this are smaller than the effects of highly inaccurate estimates.

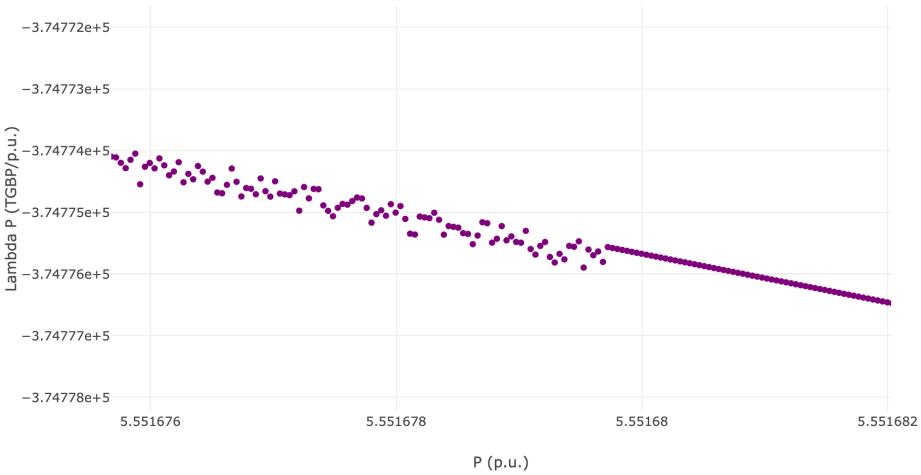


Figure 6.16: Noisy real power dual values

True height evaluation

The active cut check with true height evaluation (Algorithm 6.5) involves detecting cuts whose height is close to the objective of the converged solution and then fixing the inequality constraints of the active set that defined it.

In the examples presented in this chapter, sometimes, when trying to evaluate the accurate height of a cut, the problem returned an infeasible status; on deeper investigation, if an ϵ-value perturbed the fixed interface point (e.g. 1×10^{-9}) the solver then was often able to find an optimal solution. The practical approach adopted to bypass this issue was to flag the active set for the subproblem as invalid at the current iteration and avoid adding it until it was sampled again.

6.5 Discussion

At the beginning of the chapter, the proposed methodology had to fulfil the goals of solving a more detailed representation of the distribution networks with more realistic physics of AC-OPF, including variables that could be optimized together with the transmission network and solved in a fast manner for multi-period problems.

The modified Benders algorithm is a heuristic for the AC-OPF problems. Depending on the instance type, the decomposition achieve the goals with different performance and caveats. Single period instances were solved up to four times faster using existing information gathered historically by solving similar recurring problems. When no historical related information was used, the performance was slightly slower than in the undecomposed problems.

Single-period problems were pushed to high convergence tolerances of 0.0001%, and the bounds obtained by the heuristic were high-quality, very close to the undecomposed solution. In the multi-period problems, the convergence gap was relaxed to 0.05%, and the decomposition found feasible ponts with good performance in large cases, whereas the 48-period problem converged almost seven times faster than the undecomposed solution.

Like in the linear case, these decompositions use significantly less memory as they only represent one temporal instance of the subproblems. Given the model's reduced size, it is likely to solve bigger problems with the same amount of memory, e.g., the 96-period instance is only solvable with decomposition. Unlike the linear case, in the AC-OPF problem, the transmission network was always contained in the master problem and given this fact, it was affected by a similar drop in performance with larger models as the undecomposed problem.

As presented in the methodology, the amount of exchanged information is limited compared to the amount needed to solve the undecomposed model.

A neat feature of the decomposition is that if one of the distribution networks changes, the cuts for the other (un-modified) networks are still valid, so the sample dictionary from a previous solution can still be used. However, this does not mean that the unmodified subproblems will not be sampled, as changing one distribution network could push the other networks for different areas in the solution, but these

additional points will be used in combination with the existing valid points in the un-modified subproblems.

If a distribution network changes its conditions, e.g. a new reactive support device is installed, new sampling points and active sets will be required, and a new dictionary of samples will be collected. If later the same reactive support device is offline, the old dictionary of samples is still helpful and valid; meaning that a dictionary of samples can be stored for every operating condition, and then a valid dictionary can be put together when needed.

A drawback of the methodology is that it requires high precision active set method solutions (or certainty on the active set) to avoid grouping issues that could make the solution fail or make convergence difficult. After checking the documentation, there was no clear way in the software used (Julia, JuMP and KNITRO) to know the active set with certainty, and this had to be assessed numerically. Many practical implementation difficulties could be overcome if a solver or software package returns a clean active set.

In summary, for this type of instance, the non-linear methodology provides a practical way of getting good feasible solutions using limited information in a reduced time. Additional development could bring this outstanding performance to the next level making it more powerful and valuable.

Chapter 7

Experimental results using ADMM

ADMM has gained popularity as a decomposition method for OPF problems in recent years. Before exploring the methodology proposed in this thesis (Benders-based), an implementation of ADMM was tested and benchmarked.

ADMM works by building an augmented Lagrangian objective function for the consensus constraints, duplicating variables on both sides of the interface and updating the multipliers iteratively on the consensus constraints until both sides agree.

The general idea has been explained in Chapter 3 and applying this to the type of problem explored in this thesis requires defining augmented Lagrangian objective functions and defining a consensus interface.

The spatial interface used to explore ADMM for this type of problem is shown in Figure 7.1. It involves augmenting both sides of the network (transmission and distribution) with consensus variables, p, q, θ, v or ν. The nomenclature found in this Chapter is similar to the one explained in Chapter 6.

Unlike Benders decomposition, in ADMM there is no master and subproblem structure but problems at the same level in the hierarchy. These problems agree on a solution via a central coordination step (gathering and updating the multipliers). The problems will be referred to as transmission and distribution problems solely based on their locations.

A benefit of ADMM is that the subproblems are always feasible. Each problem has the flexibility to choose its imports, exports and voltage levels for the proposed prices (multipliers); this fact is seen when comparing Figure 7.1 to Figure 6.1. In Figure

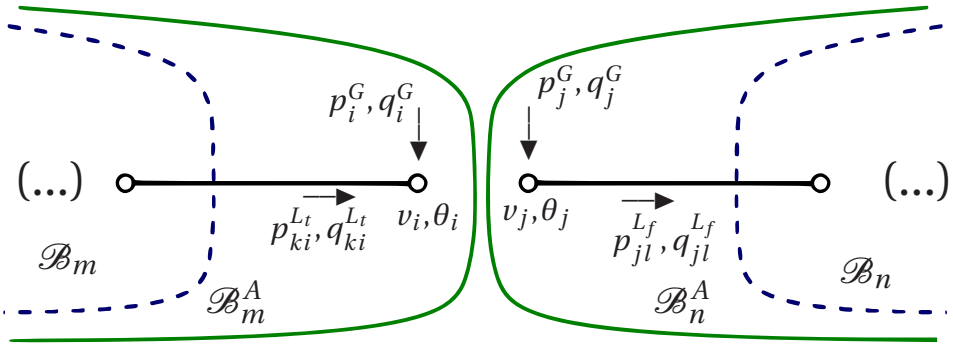


Figure 7.1: ADMM interface in the AC OPF problem

6.1 there are additional elements in one side of the interface to keep the problem feasible, whereas in Figure 7.1 are not needed.

Another benefit is that the problems in ADMM will not change their size with iterations; in the Benders methodology, constraints are added to the master problem in each iteration, and as the number of constraints grows, it can become slower to solve.

On the other hand, if a linear problem is to be solved, as in the case explored in Chapter 5, both the transmission network and distribution networks become quadratic programming problems (QPs) which may be more expensive to handle by the solver. If applied to the AC OPF problem, this will not represent an issue as the starting problem is already non-linear.

The consensus constraints (7.1) will ensure that both sides of the problem agree on a generation schedule, voltages and angles based on the proposed prices set by the multiplier update step. The constraints are similar to the ones found in the previous chapter, and the difference lies later in how the problem handles these consensus constraints.

As with the Benders decomposition formulation in the previous chapters, the signs for real and reactive power need to be changed to allow for the fact that export in one network will be seen as import power in the other and vice versa.

$$p_i^G = -p_j^G \tag{7.1a}$$

$$q_i^G = -q_j^G \tag{7.1b}$$

$$v_i = v_j \tag{7.1c}$$

$$\theta_i = \theta_j \tag{7.1d}$$

The undecomposed problem is modified to form an augmented Lagrangian version by including consensus constraints and relaxing them with linear and quadratic terms in the objective function.

The modified objective function is shown in Equation 7.2a. The first part corresponds to the original objective function; the expression in the second line corresponds to the relaxation of the consensus constraints augmented with the quadratic penalty term. Variables x_i and x_j represent the collection of variables from the consensus equations in 7.1. The subindices i, j represent the variable name at each side of each tie-line (interface); finally, the set containing all the tie lines is called $\mathscr{L}^{Tie}$.

At optimality and with consensus in the variables $x_i = x_j$, the terms found in the second line will be zero. Then, the objective value will be the same as that of the original objective function.

$$L(x_i, x_j, \lambda) = \min \left[\sum_{s \in \mathscr{S}} \frac{W_s}{|\mathscr{T}_s|} \sum_{r \in \mathscr{R}} \left(c_{rs}^{Var} + c_{rs}^{Shed} + c_{rs}^{Curt} \right) + \right.$$
$$\left. \sum_{l \in \mathscr{L}^{Tie}} \left(\lambda_l (x_i - x_j) + \frac{\rho_l}{2} \left\| x_i - x_j \right\|_2^2 \right) \right] \tag{7.2a}$$

The objective function is non-separable due to the introduction of the quadratic terms. The variables on each side of the interface are fixed alternatingly to create separable problems, and these can then be solved independently, coordinated by a common multiplier.

In order to test the ADMM methodology, a small test case was prepared to understand the effect of the penalty parameters and to see if the problem maintained the performance when scaled up. Then, a single-period AC OPF case was solved similar

to the one presented in Chapter 4.

As in previous chapters the models have been coded in Julia 1.5.3 [64] and JuMP 0.21.5 [65] as the programming modelling package, the solver used for this section is IPOPT.

7.1 Small Test Case

A small test model was developed to start the implementation of ADMM. The model consisted of 12 buses, with three buses in the transmission network where the bulk generation, storage and load are located. The distribution networks only accounts for small storage and demand. The topology of the network is presented in Figure 7.2.

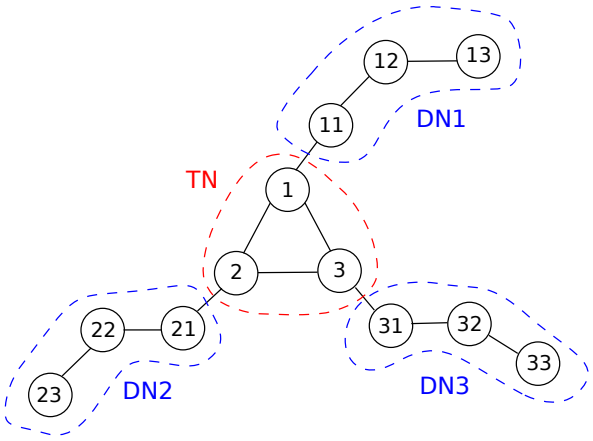


Figure 7.2: Small network: 4 Regions, 12 Bus system

The general information about the dimensions of the problem is found in Table 7.1. For this case no line limits were enforced. Each bus, however, is subjected to voltage limits. In the table, under generators column “T” refers to thermal, “R” to renewable and “S” to storage technologies.

The small test network case presented was solved undecomposed with IPOPT and then solved with a standard ADMM for every region, solving each region (including the transmission network) with IPOPT, for 48 connected periods. For this multi-period small case, the decomposition was performed in terms only of location (spatial decomposition), and both transmission and distribution problems were solved as multi-period.

Table 7.1: Small network overview

Region	Buses	Lines	Tie-lines	Gemerators	Max. demand (GW)
TN	3	3	3	11 (5T, 3R, 3S)	48.16
DN1	3	2	1	2 (1T, 1S)	0.33
DN2	3	2	1	2 (1T, 1S)	0.35
DN3	3	2	1	2 (1T, 1S)	0.36

Different penalty parameter ρ values were tested to analyse how the convergence is affected. After sweeping through different ranges, the ρ selected as a good baseline was defined as $\rho^{bl} = [200, 200, 3000]$ for the real power, reactive power and voltage magnitude, respectively. Table 7.2 shows the difference in iteration number to reach the specified convergence ($\epsilon^{primal} = 1 \times 10^{-4}$, $\epsilon^{dual} = 1 \times 10^{-4}$).

Table 7.2: ADMM Convergence with different penalty parameters, Small Test Case

Method	Penalty	Iter.	Obj. val.	Time
Unecomposed		-	7909.01	3.2
ADMM	$1/8\rho^{bl}$	76	7908.97	189.5
	$1/4\rho^{bl}$	40	7908.97	105.3
	$1/2\rho^{bl}$	22	7908.97	63.5
	ρ^{bl}	23	7909.00	62.1
	$2\rho^{bl}$	45	7909.01	115.4
	$4\rho^{bl}$	86	7909.01	208.4
	$8\rho^{bl}$	176	7909.01	435.3

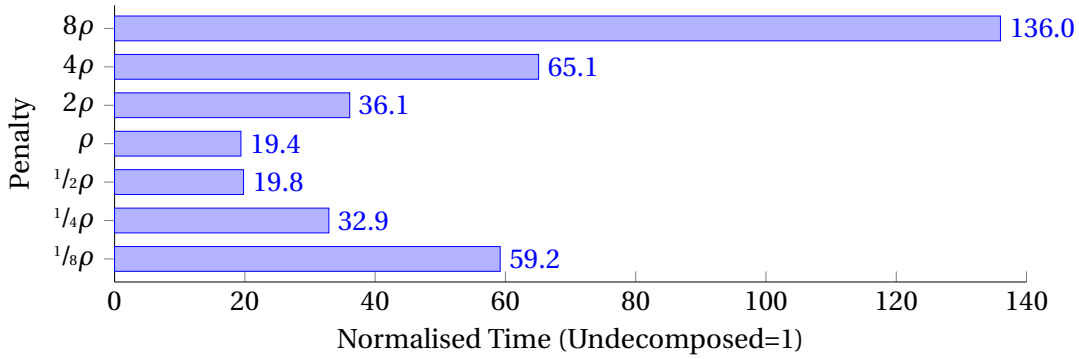


Figure 7.3: ADMM Convergence normalised time, Small Test Case

As explained in [39], a high penalty will prioritise reducing primal residuals (primal infeasibility mismatch of the consensus constraint). In contrast, a low penalty will prioritise dual residuals (a measure of suboptimality by the change in multipliers).

The algorithm implemented for all the test cases presented in this chapter is shown in 7.1. Note that the slices and periods have been dropped, but both the transmission and distribution networks are solved as multi-period in the small problem. In the algorithm, l refers to a particular line in a set of tie-lines $\mathscr{L}^{Tie}$, and the variables x_{l_i} are the ones in the tie-line from the transmission network side, with x_{l_j} being the variables on the distribution network side.

Algorithm 7.1 ADMM algorithm for the problem of interest (TN with one link to each DN)

List: problems by enumerating the tie-lines, $l \in \mathscr{L}^{Tie}$ detecting the linking elements in the distribution side, indexed by j, the index for TN-side is called i.
Require: $\hat{x}^1_{l_j st}$, μ^1_{lst}, ρ^x_{lst}, ϵ^{abs}, ϵ^{rel}, $\bar{K}$
Set: $k \leftarrow 1$, $\|r\|_2 \leftarrow \infty$, $\|s\|_2 \leftarrow \infty$, $\epsilon^{pri} \leftarrow 0$, $\epsilon^{dual} \leftarrow 0$
while $k \leq \bar{K}$ **do**
    if $\|r\|_2 \leq \epsilon^{pri}$ and $\|s\|_2 \leq \epsilon^{dual}$ **then**
        exit loop
    end if
    Solve for the TN, while the linking variables of DNs are fixed:
    $\hat{x}^k_{l_i} := \text{argmin}_{x_{l_i}} L(x_{l_i}, \hat{x}^k_{l_j}, \mu^k_l)$
    for all $l \in \mathscr{L}^{Tie}$ **do**
        $\hat{x}^k_{l_j} := \text{argmin}_{\hat{x}_{l_j}} L(\hat{x}^k_{l_i}, x_{l_j}, \mu^k_n)$
        $\mu^{k+1}_l := \mu^k_l + \rho_l \left(\hat{x}^k_{l_i} - \hat{x}^k_{l_j} \right)$
    end for
    $r := \hat{x}^k_{l_i} - \hat{x}^k_{l_j}$
    $s := \rho_l \left(\hat{x}^k_{l_j} - \hat{x}^{k-1}_{l_j} \right)$
    $\epsilon^{pri} := \sqrt{4\,|\mathscr{L}^{Tie}| \sum_{s \in \mathscr{S}} |\mathscr{P}_s|}\, \epsilon^{abs} + \epsilon^{rel} \max\left(||\hat{x}^k_{l_i}||_2, ||\hat{x}^k_{l_j}||_2 \right)$
    $\epsilon^{dual} := \sqrt{8\,|\mathscr{L}^{Tie}| \sum_{s \in \mathscr{S}} |\mathscr{P}_s|}\, \epsilon^{abs} + \epsilon^{rel} ||\mu^{k+1}_l||_2$
    $k := k + 1$
end while

7.2 Test Case

With the same network topology to the one presented in Chapter 4 a single-period problem was solved and compared to its undecomposed version.

Note that neither the demand values nor the generator costs are the exactly same

as in the test cases presented in Chapters 5 or 6 since this work was developed at an early preparatory stage before choosing a Benders based algorithm, however the scale of the problem and its cost coefficients is similar.

As seen in Table 7.3, the number of iterations required to reach convergence in the best case was 310. The "best case" was selected by independently sweeping each penalty parameter's dimensions. The best penalty factors for this problem were $\rho^{bl} = [3 \times 10^4, 100, 2 \times 10^7]$ which are entirely different to the ones that were suitable for the small network problem. The table also shows that the penalty parameters significantly affected the convergence when scaled by factors of 8, where it did not manage to converge in 4000 iterations. Other important parameters are the starting multipliers for the interface, as they dictate the baseline for the updates. Different "uninformed" combinations were tested, with the one presented for these results as 0, illustrating that if the starting multipliers or the penalty parameters are poorly selected, the convergence can be seriously affected.

Table 7.3: Convergence with different penalty parameters, Test Case

Method	Penalty	Iter.	Obj. val.	Time
Unecomposed	-	-	7529.77	7.9
ADMM	$8\rho^{bl}$	4000[1]	-	-
	ρ^{bl}	310	7257.51	604.5
	$^1/_8\rho^{bl}$	4000[1]	-	-

[1] Reached the limit of iterations.

The interesting observation about the behaviour of the networks and the prices at the boundaries (tie-lines) for power and voltage is that in the solution for the period with maximum demand, every distribution network behaves roughly the same except for the one connected to Bus 25. In the undecomposed optimal solution, the DN25 needs to do load-shedding to cope with the voltage limits inside its network. The prices at this interface are affected by the load-shedding penalty. The updates in the ADMM multipliers are slow and, for a significant number of iterations, keep demanding the transmission network to increase its voltage as the best solution for cost reduction, without knowing that although Bus 25 is not at its voltage upper bound, the voltage at it cannot be increased without causing infeasibility due to

other regional restrictions in the surrounding buses. The prices in the interface at the optimal solution are really high, and the multipliers crawl slowly towards it until a consensus is achieved.

When the loads on the distribution network DN25 are scaled down by 20%, enough to avoid load shedding in the optimal solution, and the solving process is repeated without modifying the penalties, the problem still is slow to converge. If the voltage penalties are adjusted 100 times smaller, then the problem converges in 40 iterations; however, this still takes 10 times longer to solve than the undecomposed problem.

7.3 Discussion

The simple implementation of ADMM presented in this chapter gave insight into some possible drawbacks of the methodology when applied to the type of problem of this thesis. There are two clear drawbacks of the method: high sensitivity to starting parameters and penalty factors; and a higher number of iterations.

Most of the papers reviewed that discussed how to accelerate the convergence of ADMM are directed to convex problems, and the papers reviewed that discuss accelerated techniques for non-convex problems reported limited success. The paper on adaptive ADMM for OPF [76] decomposed each element of the problem (e.g. generators and lines) up to the point where the optimization subproblems had a straightforward closed-form solution, and the penalties were adapted every 100 iterations, managed to converge the problem faster but still taking several hundred iterations.

In this chapter's second example, the load shedding profoundly impacted convergence and the magnitude of the penalty factors required to reach it. In [44] the authors show the performance of ADMM for networks between 5 and 300 buses for single-period problems; for the 5-bus case, when the authors added line limit constraints, the number of iterations jumped from 30 to 1500.

In [44] the authors present ALADIN as an alternative to ADMM, but ALADIN requires more information than ADMM: it requires the active set to be correctly estimated and to retrieve the multipliers for some of the variables fixed, being also

dependant on a penalty parameter matrix that they estimated by exploration. Multi-period problems are not considered, nor how a temporal decomposition could be achieved with this method.

Unlike ADMM, which is highly sensitive to user-defined parameters in the examples where the line limits are binding, with Benders, the high-cost multipliers would be directly retrieved and learned in one iteration, so it is not likely to be as sensitive as ADMM. As presented in [76] if the problems to be solved can be decomposed to the point where they can be written in a closed form, then the high number of iterations required may not be detrimental to the solution time. However, in the type of problem and decomposition presented in this thesis, the subproblems are still expensive to evaluate, so running a high number of iterations has significant speed consequences. Secondly, it is not evident in either ADMM or ALADIN how information can be shared between periods in terms of temporal decomposition.

In summary, before choosing to explore a Benders-based methodology for this thesis, a simple ADMM implementation was done, which gave insight into some of the drawbacks of the method when applied to the type of problem for the transmission-distribution interface discussed in this thesis, based on three reasons:

1. High sensitivity to user-defined penalty parameters and starting multipliers.
2. A very large number of iterations in some cases depending on the details of the problem.
3. No obvious extension that allows for temporal decomposition.

Chapter 8

Conclusions

This thesis has explored ways a multi-period OPF problem can be decomposed. As described in the Introduction, the motivation for this is to provide an efficient solution method for the OPF problem that can capture a topology with different representations used by transmission level operators and distribution level operators while preserving the common objective of full-network cost minimization.

In this chapter, we present reflections on the research aims described in the introductory chapter, along with reflections on the results obtained. The main work dealt with proposing Benders-based decompositions tailored for cases, such as the one presented in Chapter 4. Different decomposition interfaces have been presented for different problems, and these have been implemented to show them practical methods. The interfaces were designed to allow both temporal and spatial decomposability with the primary objective of speeding up the solution process.

8.1 Conclusions

Chapters 5 and 6 presented novel ad-hoc decomposition interfaces and methodologies adapted to the type of OPF representation used.

Chapter 5 dealt with problems using the DC approximation of the OPF, and as such, it is a linear programming problem. This methodology is based on standard Benders; being a linear problem, it can provide an exact solution and eventually converge to the optimum provided that the problem is feasible and bounded. As

only optimality cuts are added, the interface between the master problem and subproblems was modified; this allows to keep the problem feasible by introducing hefty linear penalties.

Furthermore, for this type of problem, two different interfaces were presented: one where the master problem did not include any explicit subnetwork and one where the transmission network was embedded into the master problem. The version without an embedded subnetwork proved to be faster to solve in the test cases regardless of the increased dimension of the optimality cuts.

The problems were decomposed both regionally and in time to speed up the solution process. When the time-linking constraints were transferred to the master problem, the subproblems could be solved as single-period subproblems. Once the time-varying conditions were parametrized, the optimality cuts could be shared between periods. The parametrized cuts, while not necessarily tight, are always valid. Cut sharing with similar parameter values was an efficient technique that reduced the number of required Benders iterations and the solution time. On the flip side, cut sharing proved less effective when the parametrized demands significantly differed.

In Subsection 6.3.8, we presented an algorithm to efficiently exploit the information gathered by single-period subproblems by progressively exploring the timescale of the problem; this involved learning single-period, short-term and medium-term patterns before starting to solve the target problem.

In the test problems presented, the decomposition increased the performance up to 50 times compared to the optimal undecomposed solutions. It could also solve larger instances within the same amount of memory.

The decomposition approach showed significant benefits in model building time and solution time over the undecomposed approach. When the instance was slightly modified, it solved these cases significantly faster than the modified undecomposed instances. Also, the decomposition approach maintained this advantage when problems were warm-started from previously solved related cases.

Chapter 6 presented a heuristic algorithm targeted at multiperiod AC-OPF based problems. Unlike the linear problem from Chapter 5, this AC-OPF algorithm does not ensure convergence to the optimal and can only be seen as a heuristic. The core algorithm is based on Benders decomposition. It includes modifications to minimize

overestimation errors arising from the problem's concavity; it also includes auxiliary procedures to guide the solution reducing the risk of instability and infeasibility.

First, based on auxiliary problems, the master problem builds box constraints as guidance bounds where the optimal solution is likely located. Only one quadratic cut is added per active set, and the cut is fitted with a quasi-Newton method; quadratic regularization of the master problem was necessary. Once the heuristic upper and lower bounds are close, additional cut correcting routines are performed to better approximate the problem locally (Section 6.3.8).

The single-period problems were pushed to high convergence tolerances (0.0001%) to assess the heuristic performance. The upper and lower bounds obtained were extremely close to the undecomposed solution. In the multi-period problems, the convergence gap was relaxed to 0.05%, and the decomposition showed outstanding performance. In cases with up to 48 periods, the decomposition obtained close to optimal feasible solutions up to 4 times faster than the undecomposed version without previous information and up to 7 times faster if using previous related information.

As in the linear case, these decompositions only represent one temporal instance of the subproblems, making the model faster to construct and less memory intensive. For example, the decomposition could solve 96-period problems, whereas the undecomposed model could not fit into memory.

In the decomposition approach used for the AC-OPF problems, a multi-period transmission network problem was embedded in the master problem, making it expensive to solve problems with many periods. The embedded network represents a significant drawback, as the master problem becomes expensive, and most of the iteration time is spent solving it.

Before exploring the methods from Chapters 5 and 6, ADMM was tested as the first solution approach for decomposition (Chapter 7). The ADMM implementation tested gave insight into some of the possible drawbacks of the method that may make it unsuitable for the OPF problem for the proposed Transmission-Distribution network interface.

ADMM is successful in OPF problems when the subproblems are decomposed to a level that is cheap computationally and benefit from a highly parallelizable interface. However, this was not the case with the type of interface presented in this thesis;

it was concluded that getting the first derivative information for "free" by solving a single subproblem is preferable to learning it over a long iterative process. On the other hand, using primal and first derivative information increases the amount of data to be stored and reduces the level of privacy offered by the method.

8.2 Further Work

The results from the computational experiments presented in Chapters 5 and 6 demonstrate the advantages of the proposed methods against the undecomposed solution for the test case. However, work is still left to be done to verify its behaviour. First, it is necessary to know if the performance gains are maintained with different instances and topologies. For that, using standard unmodified large IEEE or Pegase instances could be a stepping point for the DC and AC-OPF based problems.

As mentioned in Chapter 6 and at the beginning of this chapter, the algorithm for the AC-OPF problem is only a heuristic. Currently, it is not possible to assess if the point of convergence is optimal, as the thesis has not explored the theory supporting this claim, e.g. KKT conditions. Even if the algorithm in its current form is not guaranteed to move towards an optimum, the algorithm could be modified with other proven methodologies, e.g. SQP-theory, to allow for it. From a theory point of view, it is the most critical work to be developed.

A significant challenge that affects stability, precision and speed is the uncertainty in numerically estimating the active set, which is used to define and adjust the cuts added to the master problem. Currently, estimating the active set is a relatively expensive process involving re-solving subproblems using the SLQP method with high precision requirements in KNITRO; yet in some cases produced false groupings that had to be filtered out to avoid problems. Whether the active sets can be automatically detected by a solver, by a different numerical method, or determined analytically is open work that will improve the algorithm.

Including an explicit network in the master problem in the AC-OPF interface makes it significantly expensive to solve as the number of periods to be solved is increased. The removal of the network, such as in the linear case, is a potential direction to explore with the caveat that it would increase the dimension of the

non-linear cuts, potentially slowing down the solution.

Further work is required in the AC-OPF problem decomposition, where a guidance-bound subroutine is used in the master problem to initially restrict the exploration space. Faster and more precise bounds would be beneficial. An interesting approach to investigate would be to solve the undecomposed problem with a SOCP relaxation (which is very fast) and then place the guidance bounds around this solution.

For the non-linear AC OPF problems, an approximation of curvature is used based on SR1 updates. However, an alternative could be to use the exact curvature at the current sample point, which can be obtained from the subproblem solve with a small amount of extra work.

The progressive exploration variation used for DC-OPF multi-period problems solves auxiliary problems with subsets of periods from the same problem, creating a dictionary of samples that can be shared with the other periods, which was shown to speed up the convergence. The selection of periods for the auxiliary problems was made randomly, but a natural improvement would be to select the periods based on knowledge of the problem characteristics. For instance, specific days could be selected to represent many other days, or clustering techniques could be used to capture the most significant variations.

An exciting extension of the linear problem occurs naturally in stochastic investment planning problems. In a decomposition approach for this problem, the upper-level deals with the investment decisions under uncertainty and the lower level deals with the operational problem. This lower level could itself be decomposed using the methodology presented in this thesis. Suppose the retrieved data from operational subproblems were extended to give sensitivity information about parameters that vary between the different investment nodes. In that case, the operational subproblem cuts could also be shared between different nodes of the investment planning tree.

In summary, possible ways of decomposing OPF problems have been explored. The results obtained show that the decompositions presented in this thesis had benefits and worked well in the test cases, which are large problems with multiple regions and periods. Further work theoretical work is needed to extend its potential. Also, we

believe there is significant room for further extending the approaches developed in this thesis and applying them to new problem areas.

Bibliography

[1] A. Mary, B. Cain, and R. O'Neill, "History of optimal power flow and formulations," vol. 1, pp. 1–36, Dec 2012.

[2] International Energy Agency, *World Energy Outlook 2021*. Paris: IEA, 2021.

[3] bp, *bp Energy Outlook 2022*. London: bp, 2022.

[4] National Grid ESO, *Future Energy Scenarios (FES) 2022*. Warwick: NatGrid ESO, 2022.

[5] O. Zinaman, M. Miller, A. Adil, D. Arent, J. Cochran, R. Vora, S. Aggarwal, M. Bipath, C. Linvill, A. David, R. Kauffman, M. Futch, E. V. Arcos, J. M. Valenzuela, E. Martinot, M. Bazilian, and R. K. Pillai, "Power systems of the future: A 21st century power partnership thought leadership report," tech. rep., Feb. 2015.

[6] D. J. Arent, C. Barrows, S. Davis, G. Grim, J. Schaidle, B. Kroposki, M. Ruth, and B. V. Zandt, "Integration of energy systems," *MRS Bulletin*, vol. 46, pp. 1139–1152, Dec. 2021.

[7] N. K. Arora and I. Mishra, "COP26: more challenges than achievements," *Environmental Sustainability*, vol. 4, pp. 585–588, Dec. 2021.

[8] V. F. Martins and C. L. T. Borges, "Active distribution network integrated planning incorporating distributed generation and load response uncertainties," *IEEE Transactions on Power Systems*, vol. 26, pp. 2164–2172, Nov. 2011.

[9] A. Chisholm, "Energy policy now and the direction it's headed," 06 2018. Alex Chisholm, Permanent Secretary at BEIS, speaking on energy policy at the Utility Week Energy Summit.

[10] A. Vega, F. Santamaria, and E. Rivas, "Modeling for home electric energy management: A review," *Renewable and Sustainable Energy Reviews*, vol. 52, pp. 948–959, Dec. 2015.

[11] A. Kailas, V. Cecchi, and A. Mukherjee, "A survey of contemporary technologies for smart home energy management," in *Handbook of Green Information and Communication Systems*, pp. 35–56, Elsevier, 2013.

[12] International Renewable Energy Agency, *Global Renewables Outlook: 2020 Edition*. Abu Dhabi: IRENA, 2020.

[13] H. Gerard, E. I. R. Puente, and D. Six, "Coordination between transmission and distribution system operators in the electricity sector: A conceptual framework," *Utilities Policy*, vol. 50, pp. 40–48, Feb 2018.

[14] H. G. M. Valente, E. Lambert, and J. Cantenot, "Transmission system operator and distribution system operator interaction," in *Local Electricity Markets*, pp. 107–125, Elsevier, 2021.

[15] J. Hao, Y. Yang, C. Xu, and X. Du, "A comprehensive review of planning, modeling, optimization, and control of distributed energy systems," *Carbon Neutrality*, vol. 1, Aug. 2022.

[16] Z. Yuan and M. R. Hesamzadeh, "Hierarchical coordination of TSO-DSO economic dispatch considering large-scale integration of distributed energy resources," *Applied Energy*, vol. 195, pp. 600–615, June 2017.

[17] "Design for distributed energy resources," *IEEE Power and Energy Magazine*, vol. 6, pp. 30–40, May 2008.

[18] "Status review of renewable and energy efficiency support schemes in europe."

[19] OFGEM, "Smart export guarantee (SEG)." `https://www.ofgem.gov.uk/environmental-and-social-schemes/smart-export-guarantee-seg`, 2022.

[20] S. Y. Hadush and L. Meeus, "DSO-TSO cooperation issues and solutions for distribution grid congestion management," *Energy Policy*, vol. 120, pp. 610–621, Sept. 2018.

[21] H. Farmad and S. Biglar, "Integration of demand side management, distributed generation, renewable energy sources and energy storages," in *CIRED 2012 Workshop: Integration of Renewables into the Distribution Grid*, IET, 2012.

[22] E. Lambert, H. Morais, F. Reis, R. Alves, G. Taylor, A. Souvent, and N. Suljanovic, "Practices and architectures for tso-dso data exchange: European landscape," in *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pp. 1–6, 2018.

[23] H. Bouchekara, "Solution of the optimal power flow problem considering security constraints using an improved chaotic electromagnetic field optimization algorithm," *Neural Computing and Applications*, vol. 32, pp. 2683–2703, June 2019.

[24] J. Taber, D. Shawhan, R. Zimmerman, C. Marquet, M. Zhang, W. Schulze, R. Schuler, and S. Whitley, "Mapping energy futures using the superopf planning tool: An integrated engineering, economic and environmental model," in *2013 46th Hawaii International Conference on System Sciences*, pp. 2120–2129, 2013.

[25] R. Bo, "Congestion and price prediction in locational marginal pricing markets considering load variation and uncertainty," 08 2009.

[26] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische Mathematik*, vol. 4, pp. 238–252, Dec 1962.

[27] J. Lavaei and S. H. Low, "Zero duality gap in optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 27, pp. 92–107, Feb 2012.

[28] M. Ebeed, S. Kamel, and F. Jurado, "Chapter 7 - optimal power flow using recent optimization techniques," in *Classical and Recent Aspects of Power System Optimization* (A. F. Zobaa, S. H. Abdel Aleem, and A. Y. Abdelaziz, eds.), pp. 157–183, Academic Press, 2018.

[29] D. K. Molzahn and I. A. Hiskens, "A survey of relaxations and approximations of the power flow equations," *Foundations and Trends in Electric Energy Systems*, vol. 4, no. 1-2, pp. 1–221, 2019.

[30] S. H. Low, "Convex relaxation of optimal power flow—part i: Formulations and equivalence," *IEEE Transactions on Control of Network Systems*, vol. 1, pp. 15–27, Mar 2014.

[31] M. E. Baran and F. F. Wu, "Optimal capacitor placement on radial distribution systems," *IEEE Transactions on Power Delivery*, vol. 4, pp. 725–734, Jan 1989.

[32] R. Jabr, "Radial distribution load flow using conic programming," *IEEE Transactions on Power Systems*, vol. 21, pp. 1458–1459, Aug. 2006.

[33] L. Gan, N. Li, U. Topcu, and S. H. Low, "Exact convex relaxation of optimal power flow in radial networks," *IEEE Transactions on Automatic Control*, vol. 60, pp. 72–87, Jan. 2015.

[34] T. Erseghe, "Distributed optimal power flow using ADMM," *IEEE Transactions on Power Systems*, vol. 29, pp. 2370–2380, Sept. 2014.

[35] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.

[36] R. Glowinski and A. Marrocco, "On the solution of a class of non linear dirichlet problems by a penalty-duality method and finite elements of order one," in *Optimization Techniques IFIP Technical Conference*, pp. 327–333, Springer Berlin Heidelberg, 1975.

[37] D. Gabay, "Applications of the method of multipliers to variational inequalities," 1983.

[38] J. Eckstein and D. Bertsekas, "On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, pp. 293–318, Apr. 1992.

[39] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, pp. 1–122, Jan 2011.

[40] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid*, vol. 8, pp. 2941–2962, Nov 2017.

[41] S. Mhanna, A. C. Chapman, and G. Verbič, "Component-based dual decomposition methods for the OPF problem," *Sustainable Energy, Grids and Networks*, vol. 16, pp. 91 – 110, 2018.

[42] C. Duan, L. Jiang, W. Fang, and J. Liu, "Multi-period OPF with energy storages and renewable sources: A parallel moment approach," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pp. 1–5, Jul 2016.

[43] B. Houska, J. Frasch, and M. Diehl, "An augmented lagrangian based algorithm for distributed nonconvex optimization," *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1101–1127, 2016.

[44] A. Engelmann, Y. Jiang, T. Mühlpfordt, B. Houska, and T. Faulwasser, "Distributed optimal power flow using ALADIN," Feb 2018.

[45] A. Engelmann, Y. Jiang, T. Mühlpfordt, B. Houska, and T. Faulwasser, "Towards distributed OPF using ALADIN," 2018.

[46] A. J. Conejo, F. J. Nogales, and F. J. Prieto, "A decomposition procedure based on approximate newton directions," *Mathematical Programming*, vol. 93, pp. 495–515, Dec. 2002.

[47] A. Conejo, E. Castillo, R. Minguez, and R. Garcia-Bertrand, *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*. Springer, 2006.

[48] F. J. Nogales, "A decomposition methodology applied to the multi-area optimal power flow problem," *Annals of Operations Research*, vol. 120, no. 1/4, pp. 99–116, 2003.

[49] N. Mazzi, A. Grothey, K. McKinnon, and N. Sugishita, "Benders decomposition with adaptive oracles for large scale optimization," *Mathematical Programming Computation*, vol. 13, pp. 683–703, Nov. 2020.

[50] A. M. Geoffrion, "Generalized Benders decomposition," *J. Optim. Theory Appl.*, vol. 10, pp. 237–260, 1972.

[51] M. Bagajewicz and V. Manousiouthakis, "On the generalized benders decomposition," *Computers; Chemical Engineering*, vol. 15, pp. 691–700, Oct 1991.

[52] Z. Yuanxi, "Generalized Benders decomposition (GBD)." `https://optimization.mccormick.northwestern.edu/index.php/Generalized_Benders_decomposition_(GBD)`, 2015.

[53] Y. Fu, M. Shahidehpour, and Z. Li, "Security-constrained unit commitment with AC constraints," *IEEE Transactions on Power Systems*, vol. 20, pp. 1538–1550, Aug 2005.

[54] R. Jamalzadeh and M. Hong, "Microgrid optimal power flow using the generalized benders decomposition approach," *IEEE Transactions on Sustainable Energy*, vol. 10, pp. 2050–2064, Oct. 2019.

[55] N. Alguacil and A. Conejo, "Multiperiod optimal power flow using benders decomposition," *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 196–201, 2000.

[56] M. Belivanis and W. Bukhsh, "Power systems test case archive," Mar 2013.

[57] M. Baran and F. Wu, "Network reconfiguration in distribution systems for loss reduction and load balancing," *IEEE Transactions on Power Delivery*, vol. 4, pp. 1401–1407, Apr 1989.

[58] Department for Business, Energy & Industrial Strategy, "Power Stations in the United Kingdom." https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1006721/DUKES_5.11.xls, May 2021.

[59] K. McKinnon, "System data." https://www.maths.ed.ac.uk/mckinnon/old_OREI/19-20/, Jan 2020.

[60] J. Jardini, C. Tahan, M. Gouvea, S. Ahn, and F. Figueiredo, "Daily load profiles for residential, commercial and industrial low voltage consumers," *IEEE Transactions on Power Delivery*, vol. 15, no. 1, pp. 375–380, 2000.

[61] National Grid Electricity System Operator, "Historic demand data 2015." https://data.nationalgrideso.com/demand/historic-demand-data/r/historic_demand_data_2015, Jan 2020.

[62] Department for Business, Energy & Industrial Strategy, "The future for small-scale low-carbon generation: a call for evidence." https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/727138/Call_for_evidence-Future_SSLCG.pdf, Jul 2018.

[63] S. Kar, G. Hug, J. Mohammadi, and J. M. F. Moura, "Distributed state estimation and energy management in smart grids: A consensus+innovations approach," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, pp. 1022–1038, Dec 2014.

[64] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[65] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[66] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2022.

[67] W. A. Bukhsh, A. Grothey, K. I. M. McKinnon, and P. A. Trodden, "Local solutions of the optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 28, pp. 4780–4788, Nov. 2013.

[68] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, pp. 12–19, Feb. 2011.

[69] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY, USA: Springer, second ed., 2006.

[70] A. Grothey, *Decomposition Methods for Nonlinear Nonconvex Optimization Problems*. PhD thesis, University of Edinburgh, 2001.

[71] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, Apr 2005.

[72] I. S. Duff, "MA57 – a new code for the solution of sparse symmetric definite and indefinite systems," 2002.

[73] R. H. Byrd, J. Nocedal, and R. A. Waltz, "Knitro: An integrated package for nonlinear optimization," in *Nonconvex Optimization and Its Applications*, pp. 35–59, Springer US, 2006.

[74] J. F. Marley, D. K. Molzahn, and I. A. Hiskens, "Solving multiperiod OPF problems using an AC-QP algorithm initialized with an SOCP relaxation," *IEEE Transactions on Power Systems*, vol. 32, pp. 3538–3548, Sep 2017.

[75] E. D. Andersen and K. D. Andersen, *The Mosek Interior Point Optimizer for Linear Programming: An Implementation of the Homogeneous Algorithm*, pp. 197–232. Boston, MA: Springer US, 2000.

[76] N. Huebner, Y. Rink, M. Suriyah, and T. Leibfried, "Distributed AC-DC Optimal power flow in the European Transmission Grid with ADMM," 2019.

Appendix A

Dictionary of samples

Throught the thesis, in particular in Chapter 6, the concept of Dictionary of Samples is used as a list of historical solutions to add local approximations of Benders cuts and to warm-start unseen problems. It is present in the algorithms both in DC OPF and AC OPF-based problems. In this appendix, with the objective clarifying this concept, we describe the information that is gathered and classified in a dictionary of samples, and in Section A.2 we present a toy non-linear problem that also includes the routines described in Chapter 6.

A.1 Description

The structure of the dictionary of samples used in this thesis is shown in Figure A.1. A global dictionary of samples S contains information of all the subproblems $DN_1, DN_2, \ldots, DN_n$; although all subproblems are contained in the dictionary, they are independent and could be considered as different dictionaries.

The part of the dictionary for a subproblem S_{DN_1} will have an array of structures. Each element in the array corresponds to an active set. The elements in the structure are a, which is a list of inequalities detected as active when the subproblem was solved at a given point; Z, historical heights in the active set; L the historical gradient vectors and X, the historical sampled points.

When a subproblem is solved, the first step is to determine its active set; once determined, the list of inequalities will be compared to the list a for each active set

in the subproblem sample dictionary. If there is a match in the list of inequalities, then the gradients, point, and height information will be augmented in that active set. In the other case, if the active set list of inequalities does not have a match, a new position will be created in X_{sp}, and the information will be stored there.

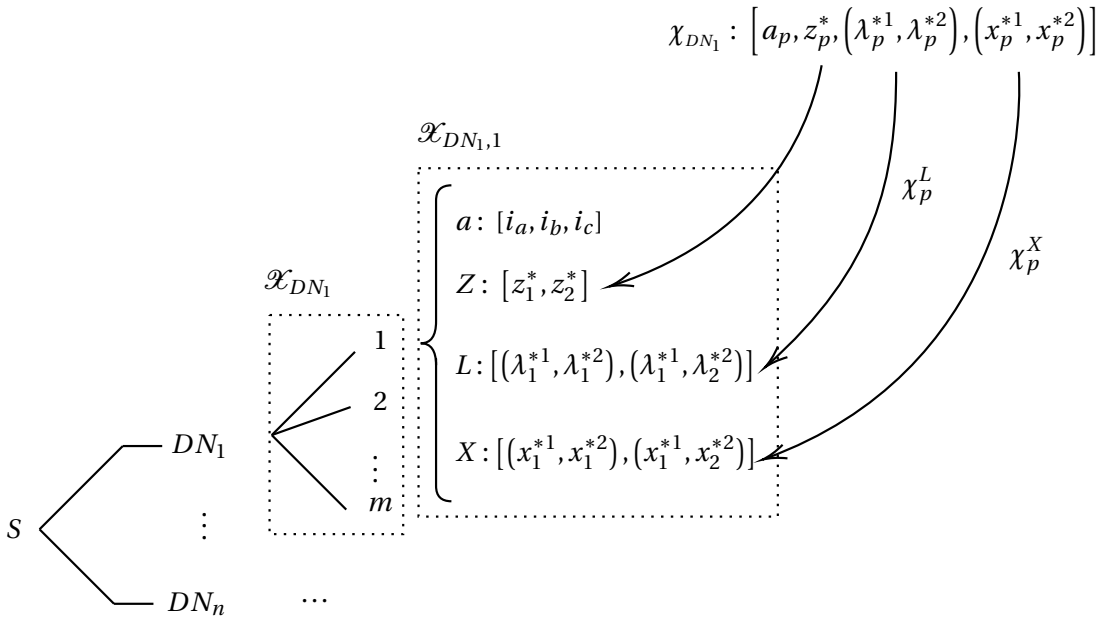


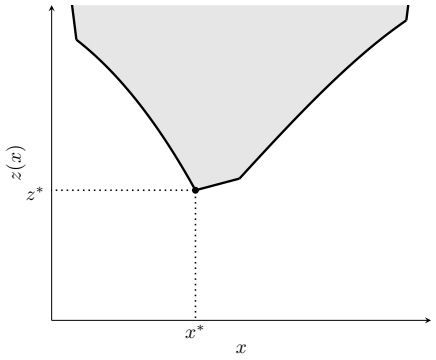
Figure A.1: Structure of the dictionary of samples

A.2 Example

The following section presents an illustrative example (previously shown in Subsec. 6.3.8), where the Benders' modified algorithm is shown in detail, highlighting the collection of samples in the dictionary. Note that the illustrative example does not consider cut sharing due to its difficulty to be illustrated.

Problem Statement

The target problem is defined in (A.1).



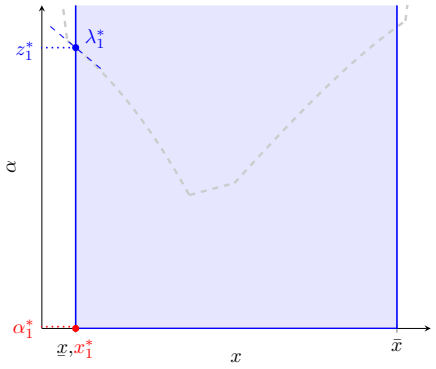
$$\min_{x} \quad z \tag{A.1}$$

subject to:

$$z \geq \quad 0.75 - 0.25x + 0.65x^2 - 0.11x^3 \tag{A.1a}$$
$$z \geq \quad 3.0625 + 0.125x - 0.25x^2 - 0.1x^3 \tag{A.1b}$$
$$z \geq \quad 1.5 + 0.25x \tag{A.1c}$$
$$z \geq \quad 10.675 - 7.5x \tag{A.1d}$$
$$z \geq -19.962 + 7.5x \tag{A.1e}$$

Iteration 1

Problem (A.1) is decomposed with Benders. In this illustrative example, the sub-problem is the same as the original target problem. The Master Problem (A.2) only includes the variable α and bounds $\underline{x}$ and $\bar{x}$ given by some guidance bounds routine; and starts with an empty sample dictionary.



$$\min_{x} \quad \alpha \tag{A.2}$$

subject to:

$$\alpha \geq 0 \tag{A.2a}$$
$$1.1 \leq x \leq 3 \tag{A.2b}$$

The information proposed and sampled in Iteration 1 is:

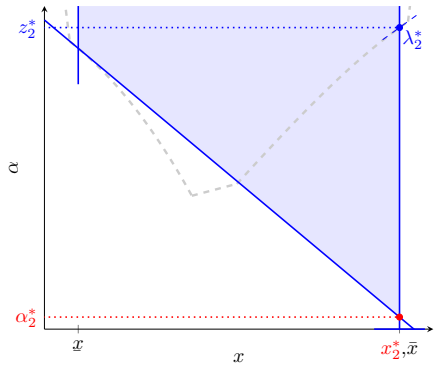
Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	−0.7880	A.1b	2.7644	100.00%

The sample dictionary is then:

$$S \text{ — } SP \text{ — } 1 \begin{cases} a: \text{[A.1b]} \\ Z: \text{[2.7644]} \\ L: \text{[(−0.7880)]} \\ X: \text{[(1.1000)]} \end{cases}$$

Iteration 2

The dictionary of samples contains information of 1 active set; this adds a linear cut, and the MP is re-optimized.



$$\min_{x} \quad \alpha \tag{A.3}$$

subject to:

$$\alpha \geq 0 \tag{A.3a}$$

$$\alpha \geq 2.7644 - 0.788\,(x - 1.1) \tag{A.3b}$$

$$1.1 \leq x \leq 3 \tag{A.3c}$$

The summary of sampled points up to Iteration 2 is:

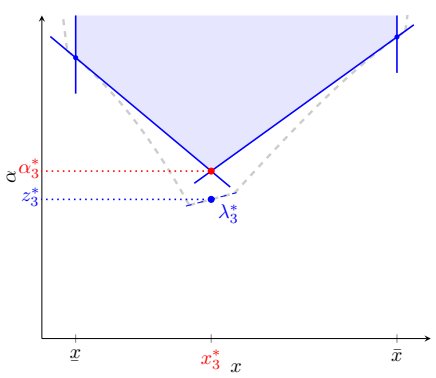
Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	−0.7880	A.1b	2.7644	100.00%
2	2	3.0000	1.2672	2.8800	0.6800	A.1a	1.6128	56.00%

The sample dictionary is then:

- S — SP
  - 1
    - a: [A.1b]
    - Z: [2.7644]
    - L: [(−0.7880)]
    - X: [(1.1000)]
  - 2
    - a: [A.1a]
    - Z: [2.8800]
    - L: [(0.6800)]
    - X: [(3.0000)]

Iteration 3

The dictionary of samples now contains information of 2 active sets which will add 2 linear cuts, as there is only one sample in each AS. The MP is re-optimized.



$$\min_{x} \quad \alpha \tag{A.4}$$

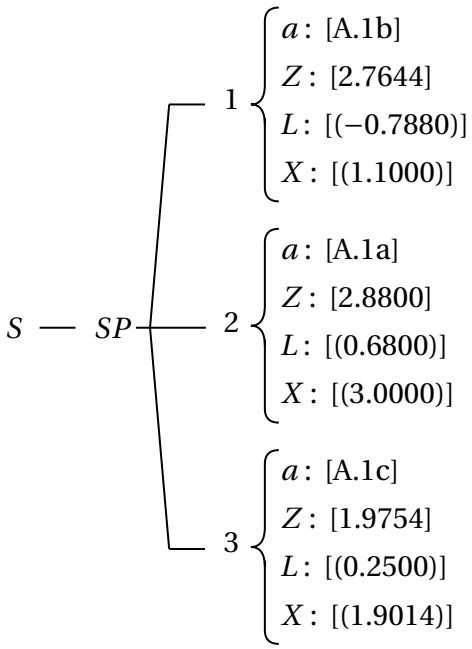
subject to:

$$\alpha \geq 0 \tag{A.4a}$$
$$\alpha \geq 2.7644 - 0.788\,(x - 1.1) \tag{A.4b}$$
$$\alpha \geq 2.88 + 0.68\,(x - 3) \tag{A.4c}$$
$$1.1 \leq x \leq 3 \tag{A.4d}$$

The summary of sampled points up to Iteration 3 is:

Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	–0.7880	A.1b	2.7644	100.00%
2	2	3.0000	1.2672	2.8800	0.6800	A.1a	1.6128	56.00%
3	3	1.9014	2.1330	1.9754	0.2500	A.1c	–0.1576	–7.39%

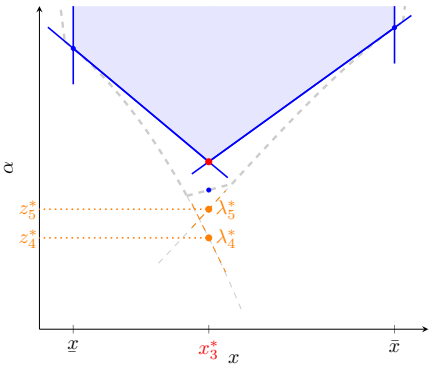
The sample dictionary is then:



In this iteration, two things happened; first, a new active set has been detected when sampling at the proposed point and; second, there is a clear violation since $\alpha_3^* >> z_3^*$, triggering a cut checking subroutine.

Cut check 1: violation

The violation found in Iteration 3 triggers a cut checking subroutine. The active sets of all the active cuts in Problem (A.4) have to be evaluated at point x_3^*. The active cuts are (A.4b) and (A.4c). Note that the MP is not solved during the cut check.



The MP is not re-optimized during the cut check for violations.

Two new samples are collected for active sets (A.1a) and (A.1b).

Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	−0.7880	A.1b	2.7644	100.00%
2	2	3.0000	1.2672	2.8800	0.6800	A.1a	1.6128	56.00%
3	3	1.9014	2.1330	1.9754	0.2500	A.1c	−0.1576	−7.39%
	4	1.9014		1.7089	−1.9103	A.1b		
	5	1.9014		1.8684	1.0288	A.1a		

The sample dictionary is then:

$$
S \;\text{—}\; SP \;\text{—}\;
\begin{cases}
1 \;\{ & a: \text{[A.1b]} \\
& Z: [2.7644, 1.7089] \\
& L: [(-0.7880), (-1.9103)] \\
& X: [(1.1), (1.9014)]
\end{cases}
\qquad
\begin{cases}
2 \;\{ & a: \text{[A.1a]} \\
& Z: [2.8800, 1.8684] \\
& L: [(0.6800), (1.0288)] \\
& X: [(3.0000), (1.9014)]
\end{cases}
\qquad
\begin{cases}
3 \;\{ & a: \text{[A.1c]} \\
& Z: [1.9754] \\
& L: [(0.2500)] \\
& X: [(1.9014)]
\end{cases}
$$

At the end of the iteration, the cuts are updated with respect to the closest information to x_3^*, the gradient and height come from the closest sample, and the curvature is fitted with SR1 by decreasing distance to point x_3^*.

The closest point to x_3^* in AS A.1a and A.1b is the same x_3 point their heights were checked there. The resulting cuts will have the gradient and heights in the sample done in x_3^* and the curvature matching the last two gradients. Since the curvature is a scalar, the SR1 formula is reduced to the secant equation.

For AS A.1a:

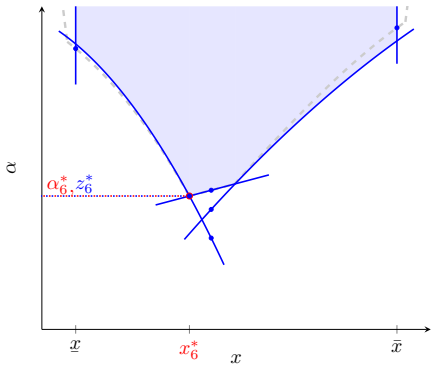
$$H = \frac{1}{2}\left(\frac{y}{r}\right) = \frac{1}{2}\left(\frac{-0.7880 + 1.9103}{1.1000 - 1.9014}\right) = -0.7002$$

For AS A.1b:

$$H = \frac{1}{2}\left(\frac{y}{r}\right) = \frac{1}{2}\left(\frac{0.6800 - 1.0288}{3.0000 - 1.9014)}\right) = 0.1587$$

Iteration 4

The new samples from the cut check update the cuts, replacing (A.4b) and (A.4c) for (A.5b) and (A.5c), respectively.



$$\min_x \quad \alpha \tag{A.5}$$

subject to:

$$\alpha \geq 0 \tag{A.5a}$$

$$\alpha \geq 1.7089 - 1.9103\,(x - 1.9014) - 0.7002(x - 1.9014)^2 \tag{A.5b}$$

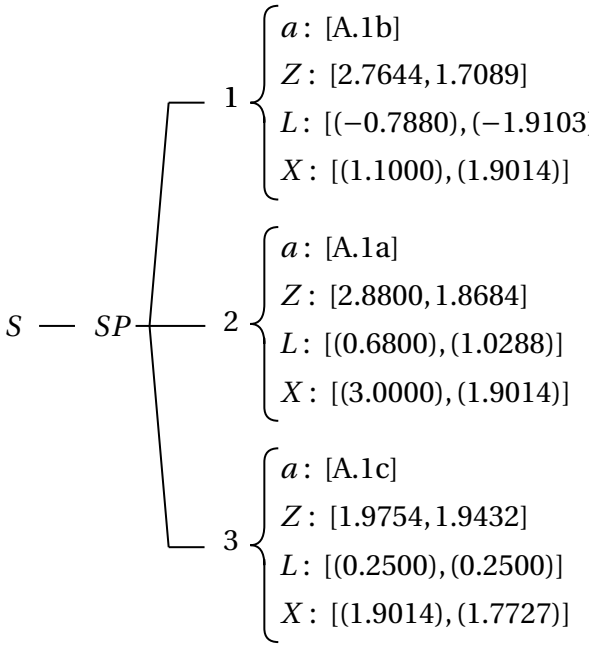
$$\alpha \geq 1.8684 + 1.0288(x - 1.9014) - 0.1587(x - 1.9014)^2 \tag{A.5c}$$

$$\alpha \geq 1.9754 + 0.25(x - 1.9014) \tag{A.5d}$$

$$1.1 \leq x \leq 3 \tag{A.5e}$$

The summary of sampled points up to Iteration 4 and the updated sample dictionary are:

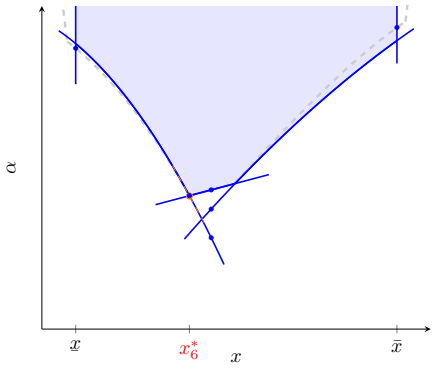
Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	−0.7880	A.1b	2.7644	100.00%
2	2	3.0000	1.2672	2.8800	0.6800	A.1a	1.6128	56.00%
3	3	1.9014	2.1330	1.9754	0.2500	A.1c	−0.1576	−7.39%
	4	1.9014		1.7089	−1.9103	A.1b		
	5	1.9014		1.8684	1.0288	A.1a		
4	6	1.7727	1.9432	1.9432	0.2500	A.1c	0.0000	0.00%



In this iteration, the approximation α_6^* matched the result given by the subproblem z_6^*, nonetheless it is necessary to perform a cut check in order to verify that the true height of the cut for the active set (A.1a) is not poorly estimated at the convergence point and preventing from going to minimum of the target problem.

Cut check 2: convergence

The problem is given the status of converged since there is no gap between the approximation and the height of the subproblem at the last explored point. The active cuts need to be checked. The 2 cuts that define the solution in Iteration 4 are (A.5b) and (A.5d). The active set (A.1c) has already been explored at the convergence point in Sample 6. Only the active set (A.1a) needs to be sampled at x_6^*.



The MP is not optimized at the start of the cut check.

The new sample is collected for active set (A.1a).

Iteration	Sample	MP		SP			Gap	
		x^*	α^*	z^*	λ^*	a		
1	1	1.1000	0.0000	2.7644	−0.7880	A.1b	2.7644	100.00%
2	2	3.0000	1.2672	2.8800	0.6800	A.1a	1.6128	56.00%
3	3	1.9014	2.1330	1.9754	0.2500	A.1c	−0.1576	−7.39%
	4	1.9014		1.7089	−1.9103	A.1b		
	5	1.9014		1.8684	1.0288	A.1a		
4	6	1.7727	1.9432	1.9432	0.2500	A.1c	0.0000	0.00%
	7	1.7727		1.9414	−1.7041	A.1b		

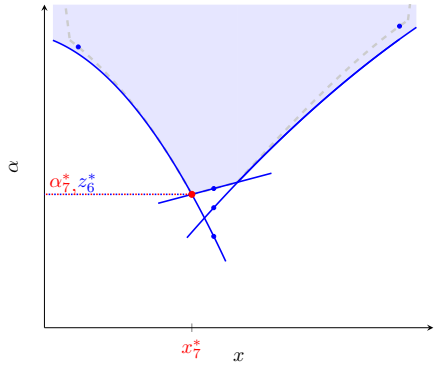
The sample dictionary is then:

$$
S \;\text{—}\; SP \;\text{—}\;
\begin{cases}
1 \;\{ & a: \text{[A.1b]} \\
& Z: [2.7644, 1.7089, 1.9414] \\
& L: [(-0.7880), (-1.9103), (-1.7041)] \\
& X: [(1.1000), (1.9014), (1.7727)] \\
2 \;\{ & a: \text{[A.1a]} \\
& Z: [2.8800, 1.8684] \\
& L: [(0.6800), (1.0288)] \\
& X: [(3.0000), (1.9014)] \\
3 \;\{ & a: \text{[A.1c]} \\
& Z: [1.9754, 1.9432] \\
& L: [(0.2500), (0.2500)] \\
& X: [(1.9014), (1.7727)]
\end{cases}
$$

At the end of the iteration, the cuts are updated with respect to the closest information to x_3^*, the gradient and height come from the closest sample, and the curvature is fitted with SR1 by decreasing distance.

Cut check 2: convergence (cont.)

As the problem is classified as converged and the binding cuts have been re-evaluated, the MP's cuts are updated (cut (A.5b) is replaced by (A.6b), and (A.5d) by (A.6d)) with the information gathered by the check and the curvatures and gradients updated with the closest information; note that the curvature for the active set (A.1b) is 0. The guidance bounds are removed as well. The subproblem is not solved during this part of the check. The algorithm will stop if the new gap between the best upper bound seen and the "verified" lower bound is small (according to the desired convergence); otherwise, a new iteration will be performed.



$$\min_{x} \quad \alpha \tag{A.6}$$

subject to:

$$\alpha \geq 0 \tag{A.6a}$$

$$\alpha \geq 1.9414 - 1.7041(x - 1.7727) - 0.8011(x - 1.7727)^2 \tag{A.6b}$$

$$\alpha \geq 1.8684 + 1.0288(x - 1.9014) - 0.1587(x - 1.9014)^2 \tag{A.6c}$$

$$\alpha \geq 1.9432 + 0.25(x - 1.9432) \tag{A.6d}$$

The value of α_6^* is 1.9429 at point $x_7^* = 1.7718$.

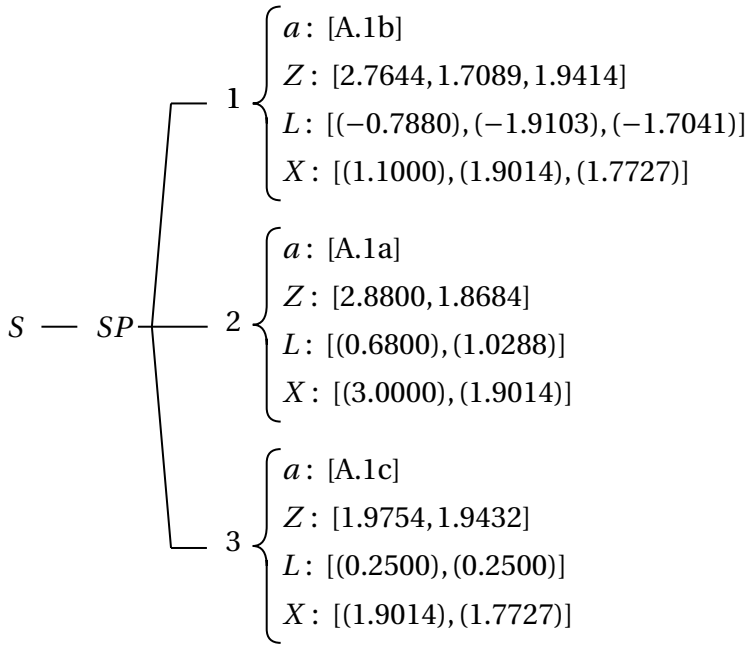
Summary

- Progress

Iter.	LB	UB	Gap (Abs)	Gap (Rel)	Converged	Act. Sets
1	0.0000	2.7644*	2.7644	100.00%	No	1
2	1.2672	2.8800	1.6128	56.00%	No	2
3	2.1330	1.9754*	−0.1576	−7.98%	No	3
4	1.9432	1.9432*	0.0000	0.00%	Yes	3
Check	1.9429			0.02%		
Summary	**1.9429**	**1.9432**	**0.0003**	**0.02%**		**3**

* Incumbent best upper bound

- Final dictionary of samples



Appendix B

Demand description

The structure of demand, demand profile and demand base are clarified in this section. A demand is a load applied to a bus, and its value depends on the base demand and the profile level at a specific period.

Figure B.1 shows a 3-bus network with 6 demands with 3 different profiles. In the Figure, there are 3 buses, each one subjected to different demands, and the colours represent 3 different profiles: residential, commercial and industrial.

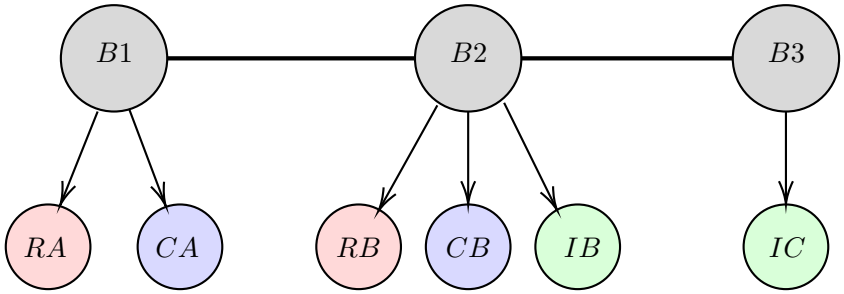


Figure B.1: Example 3-bus network with 6 demands

Each demand has the following properties: name, location, profile, real power base and reactive power base. Profiles have a name, a cost for shedding, a scaling factor and a time series attached. Tables B.1 and B.2 show an example of the data file related to demands and profiles matching Figure B.1.

Each demand has a base for real power and another for reactive power; the bases are calculated with the methodology explained in Section 4.3 (Page 37).

In Tables B.1 and B.2 the header corresponds to the name, then in the second row to the parameter units and in the last row to the name in the models found in the

Table B.1: Demand description

Name	Location	Profile	BaseP (MW)	BaseQ (MVAr)
$[d]$	$[b]$	$[p]$	$[B^P_d]$	$[B^Q_d]$
RA	$B1$	Res	10.2	2.3
CA	$B1$	Com	2.5	0.7
RB	$B2$	Res	6.1	1.2
CB	$B2$	Com	3.0	0.2
IB	$B2$	Ind	3.2	-0.9
IC	$B3$	Ind	4.1	1.3

Table B.2: Profile description

Name	CShed (GBP/MWh)	Level -	Scaling -
$[p]$	$[C^{Shed}]$	$[D_{pst}]$	
Res	3000	[[0.78,0.85,0.97,0.87,0.80],[0.30,0.35,0.33]]	1.0
Com	4000	[[0.92,0.95,0.78,0.82,0.90],[0.11,0.25,0.32]]	1.0
Ind	5000	[[1.02,0.92,0.95,0.93,0.97],[0.00,0.12,0.04]]	1.0

notation section in Chapters 5 and 6.

The column *Scaling* in Table B.2 refers to a property that modifies the whole profile in each period; this is done for convenience in problems such as investment planning where for example, in 5 years, the industrial demand is expected to reduce to 90%. The scaling factor is only a common multiplier of the *Level* property and could be omitted.

Level presents a time series indexed by slice and then by period, hence the double brackets. The example in the table includes Slice 1 with 5 periods and a Slice 2 with 3 periods.

The real power demand for period t in slice s is $P^D_{dst} = B^P_d \, D_{m(d)st}$ where B^P_d is the base of real power, and m is a mapping function such that $m : d \mapsto p$.

In this thesis for convenience in the load bus balance constraints (Eqs. 5.3c, 5.7e 6.3d, 6.3e, 6.6b, 6.6c, 6.9i and 6.9j), the shorthand of the demand-bus incidence is defined as $\mathscr{D}_b$ which means the subset of all incident demands to bus b. In this example $\mathscr{D}_{B1} = \{RA, CA\}$, such that the toal reactive load at the bus $B1$ for slice 1 and period 2 is:

$$
\begin{aligned}
\sum_{d \in \mathscr{D}_{B1}} Q^{D}_{d,1,2} &= Q^{D}_{RA,1,2} + Q^{D}_{CA,1,2} \\
&= B^{Q}_{RA,1,2}\, D_{m(RA),1,2} + B^{Q}_{CA,1,2}\, D_{m(CA),1,2} \\
&= B^{Q}_{RA,1,2}\, D_{Res,1,2} + B^{Q}_{CA,1,2}\, D_{Com,1,2} \\
&= 2.3 \text{ MVAr } (0.85) + 0.7 \text{ MVAr } (0.95) \\
&= 2.62 \text{ MVAr}
\end{aligned}
$$

Appendix C

Additional Tables

Table C.1: **Progressive exploration**, Decomposition (TN in MP), **Case B**, 1440 periods

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
***Auxiliary Problem 1**, 1-period [1288]*							
1	0.02	0.00	0.02	29	29	100.00%	BC
2	0.03	0.00	0.02	29	58	100.00%	BC
3	0.02	0.00	0.02	29	86	99.99%	BC
4	0.02	0.00	0.01	28	104	99.97%	BC
5	0.01	0.00	0.01	18	110	93.62%	BC
6	0.01	0.00	0.01	17	114	84.50%	BC
7	0.01	0.00	0.00	4	116	2.84%	BC
8	0.00	0.00	0.00	2	116	0.00%	PS
Subtotal	*0.12*	*0.03*	*0.09*	*156*			
***Auxiliary Problem 2**, 1-period [1305]*							
1	0.01	0.00	0.01	29	140	99.31%	BC
2	0.01	0.00	0.01	24	147	72.06%	BC
3	0.01	0.00	0.00	7	148	53.05%	BC
4	0.00	0.00	0.00	1	148	0.00%	PS
Subtotal	*0.04*	*0.02*	*0.02*	*61*			
***Auxiliary Problem 3**, 1-period [1387]*							
1	0.01	0.00	0.01	29	158	0.21%	BC
2	0.00	0.00	0.00	6	158	0.00%	PS
Subtotal	*0.02*	*0.00*	*0.01*	*35*			
***Auxiliary Problem 4**, 1-period [1249]*							
1	0.01	0.00	0.01	29	171	40.65%	BC
2	0.01	0.00	0.00	12	171	0.00%	PS
Subtotal	*0.02*	*0.01*	*0.01*	*41*			
***Auxiliary Problem 5**, 48-period [1157:1204]*							
1	0.96	0.56	0.40	1392	190	5.97%	BC
2	0.78	0.56	0.22	750	194	0.00%	BC
Subtotal	*1.74*	*1.12*	*0.62*	*2142*			
***Auxiliary Problem 6**, 48-period [50:97]*							
1	2.53	2.13	0.40	1392	199	0.27%	BC
2	0.10	0.05	0.06	198	199	0.00%	PS
Subtotal	*2.63*	*2.18*	*0.46*	*1590*			
***Auxiliary Problem 7**, 336-period [707:1042]*							
1	6.70	3.88	2.82	9744	222	0.39%	BC
2	1.55	1.18	0.37	1224	231	0.00%	PS
3	0.57	0.29	0.28	948	231	0.00%	PS
Subtotal	*8.82*	*5.35*	*3.47*	*11 916*			
***Target Problem**, 1440-period [1:1440]*							
1	40.38	28.98	11.23	39 248	238	0.00%	BC
2	2.27	1.63	0.64	2114	240	0.00%	PS
Subtotal	*42.65*	*30.61*	*11.87*	*41 362*			
TOTAL	**56.04**	**39.31**	**16.55**	**57 303**	**240**	**0.00%**	

Table C.2: **Progressive exploration**, Generalised decomposition (no TN in MP), **Case B**, 1440 periods

Iter.	Time (s)	MP Time (s)	SP Time (s)	SP solved	Act. Sets	Rel. Gap	Alg.
***Auxiliary Problem 1**, 1-period [1288]*							
1	0.02	0.00	0.02	30	30	100.00%	BC
2	0.02	0.00	0.02	30	60	99.78%	BC
3	0.01	0.00	0.01	30	86	99.86%	BC
4	0.01	0.00	0.01	30	90	0.39%	BC
5	0.00	0.00	0.00	4	90	0.00%	PS
Subtotal	*0.07*	*0.00*	*0.06*	*124*			
***Auxiliary Problem 2**, 1-period [1305]*							
1	0.01	0.00	0.01	30	117	99.31%	BC
2	0.01	0.00	0.01	28	130	72.25%	BC
3	0.01	0.00	0.01	19	131	53.05%	BC
4	0.00	0.00	0.00	2	131	0.00%	PS
Subtotal	*0.04*	*0.01*	*0.03*	*79*			
***Auxiliary Problem 3**, 1-period [1387]*							
1	0.01	0.00	0.01	30	145	0.48%	BC
2	0.00	0.00	0.00	8	145	0.00%	PS
Subtotal	*0.02*	*0.00*	*0.01*	*38*			
***Auxiliary Problem 4**, 1-period [1249]*							
1	0.01	0.00	0.01	30	159	46.36%	BC
2	0.01	0.00	0.00	15	159	0.00%	PS
Subtotal	*0.02*	*0.00*	*0.02*	*45*			
***Auxiliary Problem 5**, 48-period [1157:1204]*							
1	0.75	0.34	0.41	1440	192	7.30%	BC
2	0.63	0.34	0.28	999	199	0.01%	BC
3	0.06	0.03	0.03	104	199	0.00%	PS
Subtotal	*1.44*	*0.71*	*0.73*	*2543*			
***Auxiliary Problem 6**, 48-period [50:97]*							
1	0.84	0.44	0.41	1440	201	0.17%	BC
2	0.11	0.03	0.08	285	201	0.00%	PS
Subtotal	*0.95*	*0.46*	*0.49*	*1725*			
***Auxiliary Problem 7**, 336-period [707:1042]*							
1	6.72	3.84	2.88	10 080	227	0.41%	BC
2	1.33	0.78	0.55	1882	234	0.00%	PS
3	0.80	0.19	0.61	2118	238	0.00%	PS
Subtotal	*8.85*	*4.81*	*4.04*	*14 080*			
***Target Problem**, 1440-period [1:1440]*							
1	29.64	18.09	11.55	40 689	251	0.00%	BC
2	3.66	1.08	2.58	9049	254	0.00%	PS
Subtotal	*33.30*	*19.17*	*14.13*	*49 738*			
TOTAL	**44.68**	**25.17**	**19.51**	**68 372**	**254**	**0.00%**	