

Decomposition Strategies for Nonconvex Problems, a Parametric Approach

THÈSE N° 6875 (2016)

PRÉSENTÉE LE 26 FÉVRIER 2016

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE D'AUTOMATIQUE 3

PROGRAMME DOCTORAL EN GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Jean-Hubert HOURS

acceptée sur proposition du jury:

Prof. M. Paolone, président du jury

Prof. C. N. Jones, directeur de thèse

Prof. M. Diehl, rapporteur

Prof. J. Bolte, rapporteur

Prof. D. Kressner, rapporteur



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2016

Acknowledgements

I am very grateful to all the people who I have met during my PhD, who directly or indirectly contributed to this thesis.

I had the great opportunity to work under the supervision of Prof. Colin N. Jones. I would like to thank him for his support, patience, guidance and creative mind. His art of brainstorming and positive thinking were essential ingredients in order to complete this piece of work. The skills that I have acquired under his supervision will certainly be useful in my future career.

I am indebted to Prof. Victor M. Zavala for our fruitful and friendly discussions. I really enjoyed it, thanks a lot Victor ! I also had the opportunity to discuss interesting ideas with Prof. Hans-Georg Bock, I want to express my gratitude for his decisive push in one part of this thesis. I would like to thank Arvind U. Raghunathan for the passionate discussions and Prof. Quoc Tran Dinh for the very interesting scientific meetings.

I would like to thank all my coauthors who had the difficult task of reading numerous drafts and taking my comments into account. A big thanks to Ravi Gondhalekar, Prof. Melanie N. Zeilinger, Stefan Schorsch, Prof. Thomas Vetter, Prof. Marco Mazzotti and my friend and colleague Harsh Shukla. I want to express my deep gratitude to the members of my committee, Prof. Jérôme Bolte, Prof. Moritz Diehl, Prof. Daniel Kressner and Prof. Mario Paolone. This thesis benefits largely from their careful comments and suggestions.

A big thanks to my friends and colleagues Andrea, Tomasz, Giorgos, Faran, Luca, Ye, Altug, Sanket, Harsh, Martand, Tafarel, Ehsan, David, Christoph, Fransisco, Predrag, Milan, Ivan, Zlatko, Sean, Gene, Mahdieh, Sriniketh, Achille, Diogo, Yannis, Alejandro, Truong, Philippe, Christophe, Eva, Ruth, Timm, Ali, Julien, Sandra, Dominique, Roland, Ali, Francine, Basile, Wilson and Sarah. They all contributed to the great atmosphere that makes LA a nice place to live. I would like to thank the Zürich gang, Alexander, Stefan, Juan, Dave and Christian, for the pleasant meetings and discussions. A big thanks to my master and semester students Walid, Yuejiang, Romain, Philipp, Alex, Mathieu and Loïc. You did a great job !

This thesis would not exist without the unconditional love and support of my mother, my father and my two sisters, thanks a lot !

Abstract

This thesis deals with the development of numerical methods for solving nonconvex optimisation problems by means of decomposition and continuation techniques. We first introduce a novel decomposition algorithm based on alternating gradient projections and augmented Lagrangian relaxations. A proof of local convergence is given under standard assumptions. The effect of different stopping criteria on the convergence of the augmented Lagrangian loop is investigated. As a second step, a trust region algorithm for distributed nonlinear programs, named TRAP, is introduced. Its salient ingredient is an alternating gradient projection for computing a set of active constraints in a distributed manner, which is a novelty for trust region techniques. Global convergence as well as local almost superlinear convergence are proven. The numerical performance of the algorithm is assessed on nonconvex optimal power flow problems. The core of this thesis is the development and analysis of an augmented Lagrangian algorithm for tracking parameter-dependent optima. Despite their interesting features for large-scale and distributed optimisation, augmented Lagrangian methods have not been designed and fully analysed in a parametric setting. Therefore, we propose a novel optimality-tracking scheme that consists of fixed number of descent steps computed on an augmented Lagrangian subproblem and one dual update per parameter change. It is shown that the descent steps can be performed by means of first-order as well as trust region methods. Using the Kurdyka-Lojasiewicz property, an analysis of the local convergence rate of a class of trust region Newton methods is provided without relying on the finite detection of an optimal active-set. This allows us to establish a contraction inequality for the parametric augmented Lagrangian algorithm. Hence, stability of the continuation scheme can be proven under mild assumptions. The effect of the number of primal iterations and the penalty is analysed by means of numerical examples. Finally, the efficacy of the augmented Lagrangian continuation scheme is successfully demonstrated on three examples in the field of optimal control. The first two examples consists of a real-time NMPC algorithm based on a multiple-shooting discretisation. In particular, it is shown that our C++ software package is competitive with the state-of-the-art codes on NMPC problems with long prediction horizons, and can address a more general class of real-time NMPC problems. The third case study is the distributed computation of solutions to multi-stage nonconvex

optimal power flow problems in a real-time setting.

Keywords: Decomposition, continuation, nonconvex optimisation, parametric optimisation, alternating minimisations, trust region methods, augmented Lagrangian methods, Kurdyka-Lojasiewicz property, optimal control, nonlinear model predictive control.

Résumé

Cette thèse porte sur le développement de méthodes numériques basées sur des techniques de décomposition et de continuation pour la résolution de problèmes non-convexes. En premier lieu, nous proposons un algorithme de décomposition qui utilise des projections de gradients alternées appliquées à des relaxations du type Lagrangien augmenté. Une preuve de convergence locale est établie sous des conditions standards. L'effet de différents critères d'arrêt sur la convergence de la boucle du Lagrangien augmenté est analysé. Dans un deuxième temps, un algorithme à région de confiance est présenté pour la résolution de problèmes non-linéaires distribués. Son ingrédient essentiel est une projection de gradient alternée pour le calcul d'un ensemble de contraintes actives de façon distribuée. La convergence globale et une convergence presque super-linéaire localement sont prouvées. La performance numérique de l'algorithme est évaluée sur des problèmes de flux de puissance optimale non-convexes. La partie centrale de cette thèse est le développement et l'analyse d'un algorithme de Lagrangien augmenté pour le suivi d'optima dépendant de paramètres. Malgré leur potentiel pour les problèmes d'optimisation de grande taille ou distribués, les méthodes de Lagrangien augmenté n'ont pas été réellement étudiées dans un cadre paramétrique. C'est pourquoi nous proposons un nouvel algorithme de continuation qui consiste en un nombre fixe de pas de descente sur un problème de Lagrangien augmenté, suivis d'un pas dual par changement de paramètre. On montre que les pas de descente peuvent être effectués au moyen de méthodes du premier ordre ou à région de confiance. Avec l'aide de l'inégalité de Kurdyka-Lojasiewicz, la convergence locale d'une classe de méthodes à région de confiance est analysée sans recourir au fait qu'un ensemble de contraintes actives optimal a été identifié après un nombre fini d'itérations. Ce résultat nous permet d'établir une inégalité de contraction locale pour l'algorithme de continuation. De ce fait, sa stabilité peut être prouvée sous des conditions raisonnables. L'effet du nombre d'itérations primales et de la pénalité est analysé au travers d'exemples numériques. Finalement, l'efficacité de notre algorithme est démontrée au moyen de trois exemples du domaine de la commande optimale. Les deux premiers exemples sont un algorithme de NMPC temps-réel basé sur une discrétisation à tirs multiples. On montre que notre code C++ est compétitif avec les méthodes de NMPC temps-réel existantes pour des horizons de

prédiction longs, et peut être appliqué à une plus grande classe de problèmes. Le troisième exemple est la résolution distribuée de problèmes de flux de puissance optimal non-convexes sur un horizon de prédiction dans un contexte temps-réel.

Mots-clés: Décomposition, continuation, optimisation non-convexe, optimisation paramétrique, minimisations alternées, méthodes à région de confiance, méthodes de Lagrangien augmenté, propriété de Kurdyka-Lojasiewicz, commande optimale, commande prédictive non-linéaire.

Contents

Abstract	2
Résumé	4
1 Introduction	16
1.1 Overview and motivation	16
1.2 Contributions of this thesis	18
1.3 Background	21
1.3.1 Normal and tangent cones	21
1.3.2 Differentiable functions, critical points and descent Lemma	22
1.3.3 Convergent sequences	23
1.3.4 Matrix notation	23
1.3.5 Fundamental results	23
1.3.6 Dependency graph, colouring and parallel updates	25
2 Decomposition Strategies for Nonlinear Programs	26
2.1 A Method of Multipliers with Alternating Proximal Gradients	30
2.1.1 Augmented Lagrangian with relaxation of coupling constraints	31
2.1.1.1 Algorithm description	32
2.1.1.2 Convergence analysis	34
2.1.2 Splitting the augmented Lagrangian subproblem	42
2.1.2.1 Algorithm formulation	43
2.1.2.2 Convergence analysis	46
2.1.3 A coordination-decomposition algorithm	49
2.1.4 Numerical experiments	52
2.2 Trust Region with Alternating Projections (TRAP)	60
2.2.1 A trust region algorithm with distributed activity detection	61
2.2.1.1 Algorithm formulation	61

2.2.1.2	Step-sizes computation in the activity detection phase	66
2.2.1.3	Distributed computations in the refinement step	67
2.2.2	Convergence analysis	71
2.2.2.1	Global convergence to first-order critical points	72
2.2.2.2	Active-set identification	77
2.2.2.3	Local convergence rate	80
2.2.3	Numerical experiments	84
2.2.3.1	Optimal AC power flow in polar coordinates	86
2.2.3.2	Optimal AC power flow on distribution networks	90
2.2.3.2.1	On the 56-bus AC-OPF:	90
2.2.3.2.2	On the 47-bus AC-OPF:	91
3	A Parametric Decomposition Algorithm for Nonconvex Programs	95
3.1	A Parametric Augmented Lagrangian Algorithm	98
3.1.1	Problem formulation and algorithm description	98
3.1.1.1	A descent-based scheme for parametric nonconvex programs	99
3.1.1.2	Projected gradient on the parametric augmented Lagrangian	101
3.1.1.3	Trust region methods on the parametric augmented Lagrangian	101
3.1.1.4	Alternating projected gradients in parametric augmented Lagrangian	104
3.1.1.5	TRAP on the parametric augmented Lagrangian (pTRAP)	106
3.2	Local Analysis and Contraction Properties	109
3.2.1	Convergence and local analysis of the primal descent methods	109
3.2.1.1	Convergence of Algorithm 8	113
3.2.1.2	Convergence of Algorithm 10	116
3.2.1.3	Local analysis of Algorithm 9	117
3.2.1.4	Local analysis of Algorithm 11	124
3.2.2	Local primal-dual contraction	127
3.2.2.1	An auxiliary generalised equation	130
3.2.2.2	Derivation of the contraction inequality	132
3.2.2.3	Improved contraction via continuation	139
3.3	Computational Considerations and Numerical Experiments	143
3.3.1	DC motor	144
3.3.2	Collaborative tracking of unicycles	151
4	Applications in Optimal Control	155
4.1	A Direct Optimal Control Algorithm Based on Augmented Lagrangian	156

4.1.1	The optimal control problem and its multiple shooting discretisation	157
4.1.2	The augmented Lagrangian algorithm	160
4.1.2.1	Dual updates	160
4.1.2.2	Primal updates	161
4.1.2.3	Solving the trust region subproblem	165
4.1.2.4	Gradient generation	167
4.1.3	Software description	173
4.1.4	Numerical experiments	175
4.1.4.1	Inverted pendulum	175
4.1.4.2	Real-time economic NMPC on a bioreactor	184
4.2	Multi-stage Optimal AC Power Flow Problems	188
	Conclusions	196
	Bibliography	199
A	Network data	214
B	Constrained Spectrum Control	216
B.1	Introduction	216
B.2	Notation	217
B.3	Spectrum constrained NMPC	217
B.3.1	Problem formulation	217
B.3.2	Properties of the closed-loop spectrum	221
B.3.3	Tractability of spectrum constraints	221
B.4	Recursive feasibility of spectrum constrained NMPC	222
B.5	Numerical Example	227
C	Parametric Polytope Reconstruction, an Application to Crystal Shape Estimation	231
C.1	Introduction	231
C.2	Vision methods for crystal shape estimation	232
C.2.1	State of the art	232
C.2.2	Proposed approach	233
C.3	Notation	235
C.4	Basic definitions in polyhedral geometry	235
C.5	Description of the shape parameter estimation technique	236
C.5.1	Choice of the re-projection error	238

C.5.2	A nonlinear least-squares problem for parametric polytope shape estimation	240
C.6	A partition of the parameter polytope	242
C.6.1	Preliminaries	242
C.6.2	Basic results	243
C.6.3	Computational geometry aspects	246
C.6.3.1	Enumerating p -faces of C	246
C.6.3.2	Check emptiness of polyhedra T_I	247
C.7	Rotation parameterisation	247
C.8	Application to crystal shape estimation	248
C.8.1	Pre-processing: Extracting matching contours	248
C.8.2	Numerical results	250
C.8.3	Computational aspects of the shape parameter estimation procedure	251
C.8.4	Comparison to existing in-situ stereological methods	253

List of Figures

2.1	Dependency graph of NLP (2.44).	53
2.2	Euclidean norm of linear coupling constraints along outer iterations of Algorithm 1 for varying numbers of inner alternating minimisations (Algorithm 2), a multiplicative coefficient $\beta_\varrho = 2$ and an initial penalty $\varrho^{(0)} = 5$	54
2.3	Euclidean norm of linear coupling constraints along outer iterations of Algorithm 1 for different maximum number of inner alternating minimisations (Algorithm 2), a multiplicative coefficient $\beta_\varrho = 15$ and an initial penalty $\varrho^{(0)} = 5$	55
2.4	Total number of alternating projected gradient steps for reaching a feasibility of 10^{-4} in the linear coupling constraints along maximum number of primal iterations (Algorithm 2) per dual iteration (Algorithm 1). The initial penalty is $\varrho^{(0)} = 5$ and $\beta_\varrho \in \{2, 5, 15\}$	55
2.5	Number of dual iterations in Algorithm 4 to reach a feasibility 10^{-4} in the linear coupling constraints along maximum number of primal iterations per dual iteration. The initial penalty is $\varrho^{(0)} = 5$ and $\beta_\varrho \in \{2, 5, 15\}$	56
2.6	58
2.7	Workflow at node j in terms of local computations, communications with the set of neighbours \mathcal{N}_j and a central node. Note that we use the index j for a node, and not k , which corresponds to a group of nodes, in which computations are performed in parallel. Thus, the nodes in the set \mathcal{N}_j are not in the same group as node j . Thick arrows represent communications involving vectors, whereas thin arrows stand for communications of scalars. Matrix E_j is defined at Eq. (2.66).	71
2.8	The 9-bus transmission network from http://www.maths.ed.ac.uk/optenergy/LocalOpt/	87
2.9	KKT satisfaction vs iteration count in the fourth LANCELOT subproblem formed on the AC-OPF with 56 buses. When using a centralised projected search as activity detector (dotted grey) and TRAP (full black). Curves obtained with a preconditioned sCG are highlighted with triangle markers.	90

2.10	Active-set history in the first LANCELOT iteration for the 56-bus AC-OPF. Activity detection in TRAP: TRAP (full black), centralised projected search (dashed grey with triangles).	91
2.11	Norm of power flow constraints on the 56-bus network against dual iterations of a LANCELOT outer loop with TRAP as primal solver. Inner solver: TRAP (full black), centralised trust region method (dashed grey with cross markers).	92
2.12	Cumulative sCG iterations vs iteration count in the first LANCELOT subproblem formed on the AC-OPF with 56 buses. Results obtained with TRAP as inner solver (full black), with a centralised trust region method (dashed grey). Results obtained with a preconditioned refinement stage are highlighted with cross markers.	92
3.1	Angular speed against time for increasing sampling periods Δt and a fixed computational power $2 \cdot 10^3 \text{proj/sec}$: 0.004 sec (top), 0.018 sec (middle) and 0.04 sec (bottom). The sub-optimal trajectory obtained with Algorithm 7 is plotted in dashed red, while the full NMPC trajectory obtained using IPOPT (for the same Δt) is in blue.	146
3.2	Angular speed against time for increasing penalty parameters ρ and a fixed computation power $2 \cdot 10^3 \text{proj/sec}$: 20 (top), 100 (middle) and 1000 (bottom). The sub-optimal trajectory obtained with Algorithm 7 is plotted in dashed red, while the full NMPC trajectory obtained using IPOPT (for the same Δt) is in blue.	147
3.3	Optimality of bound constrained augmented Lagrangian program for different sampling periods Δt and a fixed computation power $2 \cdot 10^3 \text{proj/sec}$: 0.004 sec (black), 0.018 sec (red) and 0.04 sec (blue).	148
3.4	Norm of equality constraints $\ G(\bar{z}_k, s_k)\ _2$ for different sampling periods Δt and a fixed computation power $2 \cdot 10^3 \text{proj/sec}$: 0.004 sec (black), 0.018 sec (red) and 0.04 sec (blue).	149
3.5	Evolution of the optimality tracking error E against sampling period for different computation power: $1 \cdot 10^3$ primal iterations per sec.(red), $2 \cdot 10^3$ (black), $3 \cdot 10^3$ (blue) and $4 \cdot 10^3$ (green).	150
3.6	Evolution of the optimality tracking error E against penalty parameter ρ for $2 \cdot 10^3 \text{proj/sec}$ and $\Delta t = 0.018 \text{ sec}$	151
3.7	Evolution of the optimality tracking error E against sampling period Δt . Algorithm 7 for $3 \cdot 10^3 \text{proj/sec}$ in black, for $4 \cdot 10^3 \text{proj/sec}$ in blue. Algorithm 12 with 3 homotopy steps for $3 \cdot 10^3 \text{proj/sec}$ in dashed red, with 4 homotopy steps for $4 \cdot 10^3 \text{proj/sec}$ in red.	152

3.8	Trajectories of the three-unicycles formation for 300 proj/sec , $\Delta t = 0.20 \text{ sec}$ and $\rho = 3 \cdot 10^3$	153
3.9	Evolution of the formation error between unicycles 1 and 2 for Algorithm 7 (blue), compared with the formation error obtained with the full NMPC (IPOPT, black). . .	153
4.1	Scheme of alternating minimisation in the augmented Lagrangian subproblem resulting from the multiple-shooting discretisation. Two sets of parallel steps are required for updating all shooting nodes and inputs.	162
4.2	Sparsity structure of the hessian of the augmented Lagrangian.	163
4.3	Structured product against reduced quasi-Newton hessian approximation. The size of each block depends of the number of free variables at the corresponding shooting group.	166
4.4	Construction of the banded preconditioner.	167
4.5	Class diagram of the C++ software for solving continuous-time OCPs via multiple-shooting and augmented Lagrangian. The box T stands for C++ template class. . .	174
4.6	Schematic illustrating the inverted pendulum.	176
4.7	Angular, horizontal position and control input of inverted pendulum using MUTRAL with $T = 1 \text{ sec}$ and $N = 20$. The sub-optimal trajectories obtained with Algorithm 9 are plotted in dashed red, while the full NMPC trajectories obtained using a complete augmented Lagrangian dual loop are plotted in blue.	178
4.8	Euclidean norm of shooting constraints using MUTRAL with $T = 1 \text{ sec}$ and $N = 20$	179
4.9	KKT satisfaction on augmented Lagrangian subproblem using MUTRAL with $T = 1 \text{ sec}$ and $N = 20$	179
4.10	Solving times using MUTRAL with $T = 1 \text{ sec}$ and $N = 20$	180
4.11	Cumulative PCG iterations using MUTRAL with $T = 1 \text{ sec}$ and $N = 20$	180
4.12	Angular, horizontal position and control input of inverted pendulum using MUTRAL with $T = 3 \text{ sec}$ and $N = 60$. The sub-optimal trajectories obtained with Algorithm 9 are plotted in dashed red, while the full NMPC trajectories obtained using a complete augmented Lagrangian dual loop are plotted in blue.	181
4.13	Euclidean norm of shooting constraints using MUTRAL with horizon $T = 3 \text{ sec}$ and $N = 60$ shooting intervals.	181
4.14	KKT satisfaction on augmented Lagrangian subproblem using MUTRAL with horizon $T = 3 \text{ sec}$ and $N = 60$ shooting intervals.	182
4.15	Solving times using MUTRAL with horizon $T = 3 \text{ sec}$ and $N = 60$ shooting intervals.	182
4.16	Cumulative PCG iterations using MUTRAL with horizon $T = 3 \text{ sec}$ and $N = 60$ shooting intervals.	183

4.17	Closed-loop state and input tracking errors versus number of trust region iterations.	183
4.18	Closed-loop trajectories for the bioreactor example.	186
4.19	Solving times and cumulative PCG iterations for the bioreactor example.	187
4.20	Topology of the 47-bus network [55].	188
4.21	Topology of the 7-bus network.	189
4.22	Total predicted and actual demand curves over the 7-bus distribution network.	191
4.23	Total predicted and actual demand curves over the 47-bus distribution network.	192
4.24	Total generation curves for the 7-bus distribution network. Comparison between the generation obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP with batteries.	192
4.25	Total state of charge over the 7-bus network. Comparison between the storage obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP.	193
4.26	Euclidean norm of AC power flow constraints of the 7-bus network obtained by means of Algorithm 7 coupled with 150 and 300 iterations of pTRAP.	193
4.27	Total generation curves for the 47-bus distribution network. Comparison between the generation obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP with batteries.	194
4.28	Total state of charge over the 47-bus network. Comparison between the storage obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP.	194
4.29	Euclidean norm of AC power flow constraints of the 47-bus network obtained by means of IPOPT and Algorithm 7 coupled with 100 iterations of pTRAP.	195
B.1	Closed-loop output trajectories: Without spectrum constraint (a), with spectrum constraints at 10.5 Hz (b), with spectrum constraints at 12.1 Hz (c), and with spectrum constraints at both frequencies (d).	229
B.2	Spectrograms of the output signal of the closed-loop system: No spectrum constraint (a), spectrum constraint at 10.5 Hz (b), at 12.1 Hz (c), and at both resonances (d).	230
C.1	Schematic drawing of the flow through cell. The light from the xenon flash lamps passes through the cell. Orthogonal projections of crystals in suspension are captured by the two cameras. A pre-processing is applied to each pair of images so as to extract pairs of single crystals. Hence crystals are analysed one by one.	234

C.2	Illustration of the parametric polytope reconstruction problem: Estimate the shape parameter \tilde{t} from projections onto hyperplanes \mathcal{H}_1 , \mathcal{H}_2 and \mathcal{H}_3 . The polytope projections $\pi_{\mathcal{H}_1}P\left(A\tilde{R}^\top, B\tilde{t}\right)$, $\pi_{\mathcal{H}_2}P\left(A\tilde{R}^\top, B\tilde{t}\right)$ and $\pi_{\mathcal{H}_3}P\left(A\tilde{R}^\top, B\tilde{t}\right)$ are plotted as dark areas.	237
C.3	Pair of images of two Ibuprofen crystal in water and extracted contours, as a blue lines.	249
C.4	Output of the Douglas-Peucker applied to a pair of images of Ibuprofen. The points extracted from the images appear as black dots. The black squares correspond to the vertices of the polygonal line produced by the Douglas-Peucker algorithm. The convex polygon corresponds to the convex hull of the output points of the Douglas-Peucker procedure.	250
C.5	Polyhedral partitions obtained for Acetaminophen, L-glutamic acid β and Ascorbic acid.	250
C.6	Examples of partition polytopes for Acetaminophen and associated parametric polytopes. The shape parameter t_C^l is taken as the Chebychev center of the partition polytope for the grey polytopes, and a random vector around the Chebychev center for the transparent ones.	251
C.7	Fitting of different organic crystals. (a) Photographs with extracted contours (white) and fitted projections (dashed, red). (b) Reconstructed 3D polytope. (c) calculated scaling vector \hat{t} . The scalar s is a multiplying constant. (d) re-projection error. . . .	252
C.8	Statistics for all simulated particles. (a) scaling error ϵ_t and (b) re-projection error ϵ_{Π}	253
C.9	Statistics for the re-projection error ϵ_{Π} for all photographed particles.	254

List of Tables

2.1	Results of Algorithm 4 applied to problems of the form (2.44) for varying N . The outer loop of Algorithm 4 is aborted when the 2-norm of the linear coupling constraints reaches 10^{-4}	59
2.2	Results for the 9-bus AC-OPF (Fig. 2.8) using a standard augmented Lagrangian outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since the refinement stage was not preconditioned.	89
2.3	Results for the 9-bus AC-OPF (Fig. 2.8) using a LANCELOT outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since no preconditioner was applied in the refinement step.	89
2.4	Results for the 56-bus AC-OPF of [64] using a (local) augmented Lagrangian outer loop with TRAP as primal solver.	93
2.5	Results for the 56-bus AC-OPF of [64] using a LANCELOT outer loop with TRAP as primal solver.	93
2.6	Results for the 47-bus AC-OPF of [64] using an augmented Lagrangian outer loop with TRAP as primal solver.	93
2.7	Results for the 47-bus AC-OPF of [64] using a LANCELOT outer loop with TRAP as primal solver.	94
3.1	Comparison of Algorithm 8, 9, 10 and 11.	108
4.1	Worst-case complexity estimates of the main phases in the primal loop of the multiple-shooting augmented Lagrangian algorithm.	172
4.2	Computation times of different online NMPC software on the inverted pendulum NMPC problem with horizon $T = 1$ sec and $N = 20$ shooting intervals.	179
4.3	Computation times of different online NMPC softwares on the inverted pendulum NMPC problem with horizon 3 sec and 60 shooting intervals.	182
A.1	Line data of 47-bus network [55].	215

Chapter 1

Introduction

We begin with a motivational chapter that previews a few questions and challenges that will arise later in this thesis.

1.1 Overview and motivation

Parametric problems are ubiquitous in optimisation and engineering. Indeed, optimisation problems are frequently constructed from a mathematical model of a real-world set-up and contain quantities that are likely to vary independently from the problem formulation. These are called *parameters*. For instance, an economic equilibrium can be represented as a solution of a parametric utility maximisation program with time-varying initial endowments at the market agents [50]. In the field of power systems, the well-known Optimal Power Flow (OPF) problem has a fixed structure defined by the network topology and data with a varying power demand [93]. In signal processing, the estimation of a time-varying signal from a sensor network can be cast as a distributed a posteriori probability maximisation problem, in which the sensor measurements enter as parameters [95]. A case in point is also the Optimal Control Problem (OCP), which consists in deriving a control input to a physical system that minimises a user-defined cost while satisfying the model's dynamics as well as state or input constraints. An OCP is typically parametric in the initial state of the system or reference trajectory. Moreover, the quality of the control input depends very much on the accuracy of the data provided to the optimal control problem. This thesis focuses on a particular family of optimal control problems, namely Nonlinear Model Predictive Control (NMPC) problems [113]. In this setting, a dynamical system is controlled by repeatedly solving an OCP at regular time instants as the system's state changes. The NMPC problem contains a prediction of the future system's behaviour based on a dynamical model. Hence, an relevant question is the following: under which conditions can an exact solution of the NMPC problem at a given time instant be considered as an approximate solution of the NMPC problems at the next time instants ?

This is especially important if one intends to develop efficient computational methods for solving the NMPC problem in real-time.

Theoretical properties regarding the sensitivity of the solution of a Nonlinear Program (NLP) to small changes in the parameters have been established by means of the fixed point theorem for smooth as well as non-smooth problems [52, 137]. From an algorithmic perspective, a fundamental question in parametric optimisation, to which most of the problems mentioned in the previous paragraph can be reduced, is how to track the solution trajectory $x^*(s) \in \mathbb{R}^n$ of a parametric generalised equation

$$0 \in f(x, s) + F(x) \quad ,$$

where $s \in \mathbb{R}^p$ is a parameter, $f : \mathbb{R}^n \times \mathbb{R}^p \rightarrow \mathbb{R}^n$ is a function and F is a point-to-set map from \mathbb{R}^n to subsets of \mathbb{R}^n . Tracking refers to deriving an approximate solution $\bar{x}(s)$ that remains close to a solution $x^*(s)$ as the parameter s varies. A general concern is that the approximate solution $\bar{x}(s)$ should be generated efficiently. Various optimality-tracking techniques have been developed and are referred to as *continuation, homotopy or path-following methods*. The most well-known strategy is the Euler-Newton continuation method for solving parametric nonlinear equations [3], which has been extended to variational inequalities by [51] and achieves fourth-order accuracy in the parameter difference. In the context of NMPC with least-squares objective, a path-following technique has been proposed and analysed by [46]. It consists in solving a convex Quadratic Programming (QP) problem per parameter update. Convexity is enforced via a Gauss-Newton approximation of the hessian, which leads to local linear convergence on nonlinear least-squares problems [119]. Thus, the method of [46] seems to be limited to least-squares NMPC programs, as it is difficult to enforce convexity of the QP objective in more general instances of parametric programs.

Due to their low computational footprint, continuation methods are tailored to real-time applications [91]. Local nonlinear optimisation techniques generally require an infinite number of iterations to reach full feasibility, and are computationally expensive. On the contrary, homotopy methods reduce to solving a convex QP [46] or a Linear Complementarity Problem (LCP) per parameter change [51, 157]. If the problem is small, it is practically reasonable to do so. However, in the case of large- or huge-scale programs, one may be interested in truncating the iterative process involved in solving the subproblem at an early stage in order to satisfy hard time constraints. This may also be required by a particular small-scale application, such as embedded optimisation for instance [134]. Another important aspect in large-scale optimisation is parallelism, which aims at reducing the computational time [161]. Except in very specific cases [161], parallelising or distributing the computations involved in solving an optimisation problem is generally performed by means of *decomposition* methods [14, 23, 43, 61]. These techniques proceed by partitioning

a large-scale NLP into small subproblems that can be solved separately, and by iteratively coordinating the subproblems' solutions. Distributed optimisation methods fall under the umbrella of Lagrangian decomposition, which hinges upon convexity [17]. This limitation is due to the fact that Lagrangian decomposition consists in solving the dual of an NLP. It is well-known that a duality gap may appear in the presence of *nonconvexity* [138]. Several nonconvex decomposition strategies have been proposed to address this issue [79], but they all seem to be limited to specific problem instances, suffer from slow local convergence and have not been analysed in a dynamic setting. In conclusion, nonconvexity remains a challenge from the point of view of parametric as well as distributed optimisation.

1.2 Contributions of this thesis

This thesis attempts to bridge the gap between the three concepts discussed in the previous Section, namely *continuation*, *decomposition* and *nonconvexity*. It is structured as follows:

- **Chapter 2:** A novel nonconvex decomposition scheme is proposed and analysed. It combines an augmented Lagrangian coordination with a decomposition phase based on alternating gradient projections. To the author's knowledge, such a decomposition scheme has not appeared in the literature with this level of generality. A proof of local convergence is derived under standard regularity and optimality conditions. Moreover, the effect of different stopping criteria on the augmented Lagrangian subproblem is investigated. As a second step, a novel alternating trust region algorithm, called TRAP (Trust Region with Alternating Projections) is presented. By using alternating projected gradients as activity detectors in the Cauchy phase of a trust region method, we construct a distributed nonconvex algorithm with local almost superlinear convergence. The efficacy of the method is demonstrated by solving augmented Lagrangian subproblems constructed from real-world OPF problems. This chapter is based on the following papers that were written in collaboration with Colin N. Jones:
 - Hours, J.-H. and Jones, C.N. An alternating trust region algorithm for distributed linearly constrained nonlinear programs, application to the AC optimal power flow. *Journal of Optimization Theory and Applications*, 2015. Accepted
 - Hours, J.-H. and Jones, C.N. An augmented Lagrangian coordination-decomposition algorithm for solving distributed nonconvex programs. In *Proceedings of the 2014 American Control Conference*, pages 4312–4317, Portland, Oregon, USA, June 2014

- **Chapter 3:** A distributed continuation algorithm applicable to nonconvex programs is presented. It consists of a fixed number of steps of a descent method applied to a parametric augmented Lagrangian subproblem, and a dual update per parameter change. We establish a local contraction inequality and prove stability of the continuation scheme under mild assumptions. Our analysis builds upon *strong regularity* for quantifying sensitivity to parameter changes, and the *Kurdyka-Lojasiewicz inequality* to derive local convergence rates of the descent methods. Four different descent schemes are investigated. Depending of the point of view one may say that there are two first-order methods and two active-set strategies, or that there are two distributed techniques and two centralised schemes. A key novelty of this chapter is the derivation of a local convergence rate for trust region Newton methods that does not rely on a finite activity detection property. This is relevant in a parametric setting, as one cannot reasonably assume that the active-set at a warm-starting point is equal to the active-set at the critical point, to which the iterates converge. Finally, a sensitivity analysis with respect to the magnitude of the parameter difference, the number of descent steps and the augmented Lagrangian penalty is presented. This chapter is based on the following papers that were written in collaboration with Colin N. Jones:
 - Hours, J.-H. and Jones, C.N. A parametric nonconvex decomposition algorithm for real-time and distributed NMPC. *IEEE Transactions on Automatic Control*, 61(2), February 2016. To appear
 - Hours, J.-H. and Jones, C.N. A parametric multi-convex technique with application to real-time NMPC. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 5052–5057, Los Angeles, CA, USA, December 2014
- **Chapter 4:** Two applications of the nonconvex continuation strategy introduced in Chapter 3 in optimal control are presented. The first example is a real-time NMPC algorithm based on a multiple-shooting discretisation and a trust region algorithm from Chapter 3. A C++ software package has been implemented and is successfully demonstrated on two challenging nonlinear control problems, namely a tracking NMPC example and an economic NMPC problem. The second example is a real-world multi-stage OPF problem. It is demonstrated that the pTRAP algorithm introduced in Chapter 3 has good scalability and warm-starting properties. Some parts of this chapter have been submitted for publication.
 - Hours, J.-H. and Shukla, H. and Jones, C.N. A parametric augmented Lagrangian algorithm for real-time economic NMPC. 2015. Submitted to the 2016 European Control Conference

Finally, two topics, which are related to NMPC and parametric optimisation but not directly linked to the core of this thesis, are presented.

- **Appendix B:** A novel NMPC scheme to shape the output spectrum of nonlinear systems is presented and its theoretical properties are investigated. This chapter is based on the following papers that were written in collaboration with Ravi Gondhalekar, Melanie N. Zeilinger, Thomas Besselmann, Mehmet Mercangöz and Colin N. Jones:
 - Hours, J.-H. and Zeilinger, M.N. and Gondhalekar, R. and Jones, C.N. Constrained spectrum control. *IEEE Transactions on Automatic Control*, 60(7):1969–1974, July 2015
 - Hours, J.-H. and Zeilinger, M.N. and Gondhalekar, R. and Jones, C.N. Spectrogram-MPC: Enforcing hard constraints on system’s output spectra. In *Proceedings of the American Control Conference*, pages 2010–2017, Montreal, CA, 2012
 - Gondhalekar, R. and Jones, C.N. and Besselmann, T. and Hours, J.-H. and Mercangöz, M. Constrained spectrum control using MPC. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1219–1226, Orlando, USA, 2011
- **Appendix C:** This chapter is based on the following papers that were written in collaboration with Stefan Schorsch, Thomas Vetter, Marco Mazzotti and Colin N. Jones:
 - Hours, J.-H. and Schorsch, S. and Jones, C.N. Parametric polytope reconstruction, an application to crystal shape estimation. *IEEE Transactions on Image Processing*, 23(10):4474–4485, October 2014
 - Schorsch, S. and Hours, J.-H. and Vetter, T. and Mazzotti, M. and Jones, C.N. An optimization-based approach to extract faceted crystal shapes from stereoscopic images. *Computers and Chemical Engineering*, 75:171–183, 2015

1.3 Background

We briefly recall some basic notions in variational analysis and optimisation.

1.3.1 Normal and tangent cones

The euclidean distance of a point $x \in \mathbb{R}^n$ to a set Σ in \mathbb{R}^n is defined by

$$d(x, \Sigma) := \inf_{y \in \Sigma} \|x - y\|_2 .$$

Given a closed and convex set Ω , the euclidean projection operator onto Ω is denoted by P_Ω and is defined by

$$P_\Omega(x) := \operatorname{argmin}_{y \in \Omega} \|x - y\|_2 ,$$

where $\|\cdot\|_2$ is the euclidean norm associated with the inner product $\langle \cdot, \cdot \rangle$. The indicator function of Ω is defined by

$$\iota_\Omega(x) := \begin{cases} 0 , & \text{if } x \in \Omega , \\ +\infty , & \text{if } x \notin \Omega . \end{cases} \quad (1.1)$$

We define the normal cone to Ω at $x \in \Omega$ as

$$\mathcal{N}_\Omega(x) := \{v \in \mathbb{R}^d : \forall y \in \Omega, \langle v, y - x \rangle \leq 0\} . \quad (1.2)$$

Definition 1.1 (Sub-differential of indicator function [138]). *Given a convex set Ω , for all $x \in \Omega$,*

$$\partial \iota_\Omega(x) = \mathcal{N}_\Omega(x) .$$

The tangent cone to Ω at x is defined as the closure of feasible directions at x [138]. Both $\mathcal{N}_\Omega(x)$ and $\mathcal{T}_\Omega(x)$ are closed and convex cones. As Ω is convex, for all $x \in \Omega$, $\mathcal{N}_\Omega(x)$ and $\mathcal{T}_\Omega(x)$ are polar to each other [138].

Theorem 1.1 (Moreau's decomposition [117]). *Let \mathcal{K} be a closed convex cone in \mathbb{R}^d and \mathcal{K}° its polar cone. For all $x, y, z \in \mathbb{R}^d$, the following two statements are equivalent:*

1. $z = x + y$ with $x \in \mathcal{K}$, $y \in \mathcal{K}^\circ$ and $\langle x, y \rangle = 0$,
2. $x = P_{\mathcal{K}}(z)$ and $y = P_{\mathcal{K}^\circ}(z)$.

When the set Ω is nonconvex, the normal cone can still be defined, but some modifications are needed in comparison to the convex case.

Definition 1.2 (Definition 6.3 in [138]). *Let $x \in \Omega$. The regular normal cone to Ω at x is the set*

$$\widehat{\mathcal{N}}_{\Omega}(x) := \{v \in \mathbb{R}^n : \forall y \in \Omega, \langle v, y - x \rangle \leq o(\|y - x\|_2)\} .$$

The normal cone to Ω at x is the set of vectors $v \in \mathbb{R}^n$ such that there exists a sequence $\{x_n\} \subset \Omega$ converging to x and a sequence $\{v_n\}$ that converges to v , such that $v_n \in \widehat{\mathcal{N}}_{\Omega}(x_n)$.

All sets that appear in the remainder are assumed to be Clarke regular, that is

$$\widehat{\mathcal{N}}_{\Omega}(x) = \mathcal{N}_{\Omega}(x) ,$$

for all $x \in \Omega$. It is worth noting that the definition of the normal cone 1.2 and the definition in Lemma 1.2 coincide when the set Ω is convex.

The box-shaped set $\{x \in \mathbb{R}^n : \forall i \in \{1, \dots, n\}, l_i \leq x_i \leq u_i\}$ is denoted by $\mathbb{B}(l, u)$. For $x \in \mathbb{R}^n$ and $r > 0$, the open ball of radius r centered around x is denoted by $\mathcal{B}(x, r)$. Given $x \in \Omega$, where the set Ω is defined as

$$\Omega := \{x \in \mathbb{R}^n : g_1(x) \leq 0, \dots, g_m(x) \leq 0\} ,$$

with the functions g_j being continuous, the set of active constraints at x is

$$\mathcal{A}_{\Omega}(x) := \{j \in \{1, \dots, m\} : g_j(x) = 0\} .$$

1.3.2 Differentiable functions, critical points and descent Lemma

Given a differentiable function f of several variables x_1, \dots, x_n , its gradient with respect to variable x_i is denoted by $\nabla_i f$ or $\nabla_{x_i} f$ without distinction.

Lemma 1.1 (Critical point). *Let f be a proper lower semicontinuous function. A necessary condition for x^* to be a minimiser of f is that*

$$0 \in \partial f(x^*) , \tag{1.3}$$

where $\partial f(x^*)$ is the sub-differential of f at x^* [138].

Points satisfying (1.3) are called *critical points*. A critical point x^* of the function $f + \iota_{\Omega}$ with

f differentiable, is said to be non-degenerate if

$$-\nabla f(x^*) \in \text{ri}(\mathcal{N}_\Omega(x^*)) \quad ,$$

where given a set $S \subseteq \mathbb{R}^n$, $\text{ri}(S)$ is its relative interior, which is defined as the interior of S within its affine hull.

Lemma 1.2 (Descent lemma [16]). *Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ a continuously differentiable function such that its gradient ∇f is ℓ_L -Lipschitz continuous. For all $x, y \in \mathbb{R}^n$,*

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\ell_L}{2} \|y - x\|_2^2 \quad .$$

Given a polynomial function f , its degree is denoted by $\deg(f)$. A semi-algebraic function is a function whose graph can be expressed as a union of intersections of level sets of polynomials.

1.3.3 Convergent sequences

A sequence $\{x^l\}$ converges to x^* at a Q-linear rate $\varrho \in]0, 1[$ if, for l large enough,

$$\frac{\|x^{l+1} - x^*\|_2}{\|x^l - x^*\|_2} \leq \varrho \quad .$$

The convergence rate is said to be Q-superlinear if the above ratio tends to zero as l goes to infinity.

1.3.4 Matrix notation

Given a matrix $M \in \mathbb{R}^{m \times n}$, its (i, j) element is denoted by $M_{i,j}$.

1.3.5 Fundamental results

The following Lemma is a reformulation of Theorem 6.14 in [138].

Lemma 1.3 (Expression of a normal cone under constraint qualification). *Let*

$$\mathcal{C} = \{x \in \mathcal{X} : F(x) = 0\} \quad ,$$

with \mathcal{X} a closed set in \mathbb{R}^n and F a continuously differentiable mapping from \mathbb{R}^n to \mathbb{R}^m , written as

$$F(x) = (f_1(x), \dots, f_m(x))^\top \quad .$$

Given $\bar{x} \in \mathcal{C}$, if the only vector $v \in \mathbb{R}^m$ such that

$$-\sum_{i=1}^m v_i \nabla f_i(\bar{x}) \in \mathcal{N}_{\mathcal{X}}(\bar{x})$$

is 0 and if \mathcal{X} is regular at \bar{x} , then the normal cone to \mathcal{C} at \bar{x} is

$$\mathcal{N}_{\mathcal{C}}(\bar{x}) = \left\{ w + \sum_{i=1}^m \nu_i \nabla f_i(\bar{x}) \ : \ w \in \mathcal{N}_{\mathcal{X}}(\bar{x}), \nu \in \mathbb{R}^m \right\} .$$

The following Lemma is particularly useful in the analysis of augmented Lagrangian algorithms. Its proof can be found in [135] for instance.

Lemma 1.4 (Debreu Lemma). *Let H denote a symmetric $n \times n$ matrix, and let J denote an $m \times n$ matrix. The matrix H is positive definite on the null-space of J if and only if there exists a positive $\bar{\rho}$ such that for all $\rho > \bar{\rho}$, the matrix $H + \rho J^T J$ is positive definite.*

The following property, which characterises a generalised equation, plays an important role in the thesis.

Definition 1.3 (Strong regularity of a generalised equation [137]). *Let Ω be a closed and convex set in \mathbb{R}^n and $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ a differentiable mapping. The generalised equation $0 \in f(x) + \mathcal{N}_{\Omega}(x)$ is said to be strongly regular at a solution $x^* \in \Omega$ if there exists radii $\eta > 0$ and $\kappa > 0$ such that for all $r \in \mathcal{B}(0, \eta)$, there exists a unique $x_r \in \mathcal{B}(x^*, \kappa)$ such that*

$$r \in f(x^*) + \nabla f(x^*)(x_r - x^*) + \mathcal{N}_{\Omega}(x_r) \ ,$$

and the inverse mapping $r \mapsto x_r$ from $\mathcal{B}(0, \eta)$ to $\mathcal{B}(x^*, \kappa)$ is Lipschitz continuous.

The strong regularity property 1.3 implies the following Theorem, which can be regarded as a version of the implicit function theorem for generalised equations. Its proof, which is given in [137], relies on a fixed point argument, like the implicit function theorem.

Theorem 1.2. *Let Ω be a closed and convex set in \mathbb{R}^n and $f : \mathbb{R}^n \times \mathbb{R}^p \rightarrow \mathbb{R}^n$ be continuously differentiable in both variables. Given $s_0 \in \mathbb{R}^p$, assume that x_0 is a solution of the parametric generalised equation*

$$0 \in f(x, s_0) + \mathcal{N}_{\Omega}(x) \ . \tag{1.4}$$

If the generalised equation (1.4) is strongly regular at x_0 with Lipschitz constant ℓ_0 , then for any $\epsilon > 0$, there exists $\delta > 0$ and $\kappa > 0$ such that for all $s \in \mathcal{B}(s_0, \kappa)$, there exists a unique $x(s)$ in

$\mathcal{B}(x_0, \delta)$ such that

$$0 \in f(x(s), s) + \mathcal{N}_\Omega(x(s)) \ .$$

Moreover, given $s_1, s_2 \in \mathcal{B}(s_0, \kappa)$, one has

$$\|x(s_1) - x(s_2)\|_2 \leq (\ell_0 + \epsilon) \|f(x(s_2), s_1) - f(x(s_2), s_2)\|_2 \ .$$

1.3.6 Dependency graph, colouring and parallel updates

An iterative method, which generates a sequence $\{x^l\} \subset \mathbb{R}^n$ can be represented by

$$x^{l+1} = F(x^l) \ , \tag{1.5}$$

where F is a mapping from \mathbb{R}^n to \mathbb{R}^n . Let x_i denote the i th component of the vector x and F_i denote the i th component of the mapping F . The execution of an iteration (1.5) can be represented by a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, which is called the *dependency graph* of F . The set of nodes $\mathcal{N} = \{1, \dots, n\}$ corresponds to the components of x . Given two distinct nodes i and j , (i, j) is an arc of the dependency graph \mathcal{G} if and only if the component F_j depends on component x_i . An important question in distributed optimisation is whether it is possible to maximise the number of parallel updates in (1.5). It can be shown that this is actually equivalent to finding an optimal colouring of the dependency graph \mathcal{G} [17].

Lemma 1.5 (Proposition 2.5 in [17]). *Assume that the mapping F and the dependency graph \mathcal{G} correspond to a Gauss-Seidel sweep. The following two statements are equivalent:*

- *There exists an ordering of the variables x_1, \dots, x_n such that the update (1.5) can be computed in K parallel steps.*
- *There exists a colouring of the dependency graph \mathcal{G} that uses K colours, and with the property that there exists no positive cycle with all nodes in the cycle having the same colour.*

It is worth noting that the optimal colouring problem is NP-complete [68]. However, many problems are structured and it is then possible to find an optimal colouring by inspection, as shown later in the thesis.

Chapter 2

Decomposition Strategies for Nonlinear Programs

Minimising a separable, smooth and nonconvex function subject to partially separable equality constraints [73, 74] and separable constraints

$$\begin{aligned} & \underset{x_1, \dots, x_N}{\text{minimise}} \quad \sum_{i=1}^N f_i(x_i) \\ & \text{s. t.} \quad C(x_1, \dots, x_N) = 0 \\ & \quad \quad x_1 \in \mathcal{X}_1, \dots, x_N \in \mathcal{X}_N, \end{aligned}$$

appears in many engineering problems such as Distributed Nonlinear Model Predictive Control (DNMPC) [118], power systems [100] and wireless networking [29]. For such problems involving a large number of agents, which result in large-scale nonconvex Nonlinear Programs (NLP), it may be desirable to perform computations in a distributed manner, meaning that all operations are not carried out on one single node, but on multiple nodes spread over a network and that information is exchanged during the optimisation process. Such a strategy may prove useful to reduce the computational burden in the case of extremely large-scale problems. Moreover, autonomy of the agents may be hampered by a purely centralised algorithm. Case in points are cooperative tracking using DNMPC [86] or the Optimal Power Flow problem (OPF) over a distribution network [64], into which generating entities may be plugged or unplugged. Moreover, it has been shown in a number of studies that distributing and parallelising computations can lead to significant speed-up in solving large-scale NLPs [161]. Splitting operations can be done on distributed memory parallel environments such as clusters [161], or on parallel computing architectures such as Graphical Processing Units (GPU) [56].

Our objective is to develop nonlinear programming methods in which most of the computations can be distributed and even parallelised. Some of the key features of a distributed optimisation strategy are the following:

- (i) *Distributed memory*. Vectors and matrices involved in the optimisation process are stored in different physical memories on different nodes. This requirement makes direct linear algebra methods more difficult to apply, as they generally require the assembly of matrices on a central unit.
- (ii) *Concurrency*. A high level of parallel computations is obtained at every iteration.
- (iii) *Cheap exchange*. Global communications of agents with a central node are cheap (scalars). More costly communications (vectors) remain local between neighbouring agents. In general, the amount of communication should be kept as low as possible. It is already clear that globalisation strategies based on line-search do not fit with the distributed framework [56], as these entail evaluating a ‘central’ merit function multiple times per iteration, thus significantly increasing communications.
- (iv) *Inexactness*. Convergence is ‘robust’ to inexact solutions of the subproblems, since it may be necessary to truncate the number of sub-iterations due to communication costs.
- (v) *Fast convergence*. The sequence of iterates converges at a fast (at least linear) local rate. Slow convergence generally results in a prohibitively high number of communications.

As we are interested in applications such as DNMPC, which require solving distributed parametric NLPs within a limited amount of time [86], a desirable feature of our algorithm should also be

- (vi) *Warm-start and activity detection*. The algorithm detects the optimal active-set quickly and enables warm-starting.

Whereas a fair number of well-established algorithms exist for solving distributed convex NLPs [17], there is, as yet, no consensus around a set of practical methods applicable to distributed nonconvex programs. Some work [161] exists on the parallelisation of linear algebra operations involved in solving nonconvex NLPs with IPOPT [152], but the approach is limited to very specific problem structures and the globalisation phase of IPOPT (filter line-search) is not suitable for fully distributed implementations (requirements (iii), (iv) and (vi) are not met). Among existing strategies capable of addressing a broader class of distributed nonconvex programs, one can make a clear distinction between Sequential Convex Programming (SCP) approaches and augmented Lagrangian techniques.

An SCP method consists in iteratively solving distributed convex NLPs, which are local approximations of the original nonconvex NLP. To date, some of the most efficient algorithms for solving distributed convex NLPs combine dual decomposition with smoothing techniques [118, 149]. On the contrary, an augmented Lagrangian method aims at decomposing a nonconvex auxiliary problem inside an augmented Lagrangian loop [30, 79, 84]. While convergence guarantees

can be derived in both frameworks, computational drawbacks also exist on both sides. For instance, it is not clear how to preserve the convergence properties of SCP schemes when every subproblem is solved to a low level of accuracy. Hence, (iv) is not satisfied immediately. Nevertheless, for some recent work in this direction, one may refer to [147]. The convergence rate of the algorithm analysed in [147] is sublinear, thus not fulfilling (v). On the contrary, the inexactness issue can be easily handled inside an augmented Lagrangian algorithm, as global and fast local convergence is guaranteed even though the subproblems are not solved to a high level of accuracy [34, 57]. However, in practice, poor initial estimates of the dual variables can drive the iterative process to infeasible points. Moreover, it is still not clear how the primal nonconvex subproblems should be decomposed and solved efficiently in a distributed context. The quadratic penalty term of an augmented Lagrangian does not allow for the same level of parallelism as a (convex) dual decomposition. Thus, requirement (ii) is not completely satisfied. To address this issue, we propose applying Proximal Alternating Linearised Minimisations (PALM) [21] to solve the auxiliary augmented Lagrangian subproblems [84, 86]. The resulting algorithm inherits the slow convergence properties of proximal gradient methods and does not readily allow for preconditioning. In this chapter, a novel mechanism for handling the augmented Lagrangian subproblems in a more efficient manner is proposed and analysed. The key idea is to use alternating gradient projections to compute a Cauchy point in a trust region Newton method [36].

When looking at practical trust region methods for solving bound-constrained problems [158], one may notice that the safeguarded Conjugate Gradient (sCG) algorithm is well-suited to distributed implementations, as the main computational tasks are structured matrix-vector and vector-vector multiplications, which do not require the assembly of a matrix on a central node. Moreover, the global communications involved in an sCG algorithm are cheap. Thus, sCG satisfies requirements (i), (ii) and (iii). The implementation of CG on distributed architectures has been extensively explored [45, 56, 151]. Furthermore, a trust region update requires only one centralised objective evaluation per iteration. From a computational perspective, it is thus comparable to a dual update, which requires evaluating the constraints functional and is ubiquitous in distributed optimisation algorithms. However, computing the Cauchy point in a trust region loop is generally done by means of a projected line-search [158] or sequential search based on a sorting algorithm [32]. Whereas it is broadly admitted that the Cauchy point computation is cheap, this operation requires a significant amount of global communications in distributed memory parallel environments, and is thus hardly amenable to such applications [56]. This hampers the implementability of trust region methods with good convergence guarantees on distributed computing platforms, whereas many parts of the algorithm are attractive for such implementations. The aim of this chapter is to bridge the gap by proposing a novel way of computing the Cauchy point that is more tailored to the distributed framework. Coordinate gradient descent methods such as PALM, are known to be parallelisable

for some partial separability structures [17]. Moreover, in practice, the number of backtracking iterations necessary to select a block step-size, can be bounded, making the approach suitable for ‘Same Instruction Multiple Data’ architectures. Therefore, we propose using one sweep of block-coordinate gradient descent to compute a Cauchy point. As shown in paragraph 2.2.2, such a strategy turns out to be efficient at identifying the optimal active-set. It can then be accelerated by means of an inexact Newton method. As our algorithm differs from the usual trust region Newton method, we provide a detailed convergence analysis in paragraph 2.2.2. Finally, one should mention a recent paper [154], in which a trust region method is combined with alternating minimisations, namely the Alternating Directions Method of Multipliers (ADMM) [17], but in a very different way from the strategy described next. The approach of [154] relies on a filter [59] and contains centralised safeguarding mechanisms in addition to the trust region update. Moreover, a local convergence rate is not established in [154].

2.1 A Method of Multipliers with Alternating Proximal Gradients

In this section, we propose a first-order strategy to decompose the augmented Lagrangian subproblem resulting from the relaxation of coupling constraints. Taking inspiration from ADMM, we suggest using an alternating direction method to compute an approximate first-order critical point of the augmented Lagrangian subproblem. Alternating direction methods have recently experienced a revival of interest with the work of [11, 21]. Our strategy is related to the PALM algorithm presented in [21]. However, in its current formulation, PALM requires knowledge of block-wise Lipschitz constants, which makes it unpractical for many problems of interest. Therefore, we propose resorting to a block-coordinate backtracking procedure. Similarly to PALM [21], the salient ingredient of our algorithm is the projection onto nonconvex sets, which can be computed in closed form in several cases. To make this effective, the separable nonconvex constraints should be kept as such in the augmented Lagrangian subproblem. From a theoretical perspective, this creates two challenges.

First, the convergence properties of the augmented Lagrangian dual loop are unclear when nonconvex constraints appear in the subproblems. Most of the existing work on augmented Lagrangian methods for nonlinear programs deals with convex or linear constraint sets [34, 31]. To the author's knowledge, only one approach in the literature is applicable with nonconvex constraints in the subproblem [8]. Its advantage and limitations are discussed next.

Secondly, alternating minimisations may fail to converge in a nonconvex setting, as a zigzagging behaviour can be observed in some cases [123]. However, as shown later, for guaranteeing convergence of the outer loop, subsequence convergence of the inner iterates to a first-order critical point is sufficient. Thus, our algorithm borrows the proximal regularisation mechanism of [11, 21] in order to ensure that all limit points of the primal sequence are critical points.

We consider the following class of nonconvex programs

$$\begin{aligned}
 & \underset{z_1, z_2}{\text{minimise}} && J(z_1, z_2) && (2.1) \\
 & \text{s. t.} && C(z_1, z_2) = 0 \\
 & && G_1(z_1) = 0, G_2(z_2) = 0 \\
 & && z_1 \in \Omega_1, z_2 \in \Omega_2,
 \end{aligned}$$

where $z_1 \in \mathbb{R}^{n_1}$, $z_2 \in \mathbb{R}^{n_2}$. We define $\mathcal{Z}_1, \mathcal{Z}_2$, subsets of \mathbb{R}^{n_1} and \mathbb{R}^{n_2} respectively, and \mathcal{Z} subset of

\mathbb{R}^n , as follows

$$\mathcal{Z}_1 := \{z_1 \in \Omega_1 : G_1(z_1) = 0\}, \mathcal{Z}_2 := \{z_2 \in \Omega_2 : G_2(z_2) = 0\}, \mathcal{Z} := \mathcal{Z}_1 \times \mathcal{Z}_2. \quad (2.2)$$

The nonlinear program (2.1) is written in terms of two blocks of variables z_1 and z_2 only. However, the algorithm and analysis that follow readily extend to multiple blocks of variables. In a distributed framework, the equality constraint $C(z_1, z_2) = 0$ models a coupling between agents associated to variables z_1 and z_2 . The separable equality constraints $G_1(z_1) = 0$ and $G_2(z_2) = 0$ may model discretised dynamics of each agent, for instance.

Assumption 2.1 (Polyhedral constraints). *The sets Ω_1 and Ω_2 are non-empty and polyhedral. More precisely, for every $i \in \{1, 2\}$, there exists a matrix $H_i \in \mathbb{R}^{m_i \times n_i}$ and a vector $\omega_i \in \mathbb{R}^{m_i}$ such that*

$$\Omega_i = \{z_i \in \mathbb{R}^{n_i} : H_i z_i \leq \omega_i\}, \quad (2.3)$$

with $m_i \geq 1$.

After stating the augmented Lagrangian algorithm, we provide a proof of local convergence to a KKT point under standard assumptions (Theorem 2.1). Similarly to the mechanism of [16], we show that the augmented Lagrangian subproblem has a unique solution under some conditions on the penalty and the Lagrange multiplier. For our specific setting, strong regularity [137] is needed. The main difference compared to the existing proofs is that the nonlinear constraints in the subproblem need special care. Then, the primal alternating minimizations algorithm is introduced. It is proven that all limit points of the sequence generated by this algorithm are critical point of the augmented Lagrangian, which is sufficient to ensure convergence of the outer augmented Lagrangian loop.

2.1.1 Augmented Lagrangian with relaxation of coupling constraints

In this paragraph, we propose and analyse a special form of the method of multipliers for computing first-order critical points of the following nonconvex program

$$\begin{aligned} & \underset{z}{\text{minimise}} \quad J(z) & (2.4) \\ & \text{s. t.} \quad C(z) = 0 \\ & \quad \quad G(z) = 0 \\ & \quad \quad z \in \Omega, \end{aligned}$$

where $z \in \mathbb{R}^n$, $J : \mathbb{R}^n \rightarrow \mathbb{R}$, $C : \mathbb{R}^n \rightarrow \mathbb{R}^p$ and $G : \mathbb{R}^n \rightarrow \mathbb{R}^q$ with $n, p, q \geq 1$. Problem (2.4) corresponds to a reformulation of NLP (2.1) with $z = (z_1^\top, z_2^\top)^\top$, $n = n_1 + n_2$ and

$$C(z) = C(z_1, z_2), \quad G(z) = (G_1(z_1)^\top, G_2(z_2)^\top)^\top, \quad \Omega = \Omega_1 \times \Omega_2 .$$

From Assumption 2.11, the constraint set Ω can be written

$$\Omega = \{z \in \mathbb{R}^n : Hz \leq \omega\} ,$$

with

$$H := [H_1^\top, H_2^\top]^\top, \quad \omega := [\omega_1^\top, \omega_2^\top]^\top . \quad (2.5)$$

The row vector i of matrix H is denoted by $h_i \in \mathbb{R}^n$.

2.1.1.1 Algorithm description

In order to compute a critical point of NLP (2.4), we propose a method of multipliers with relaxation of the equality constraint $C(z) = 0$, which models a coupling between variables z_1 and z_2 , as shown in (2.1). At every iteration of the proposed procedure, the main computational task is the derivation of an approximate critical point of the partial augmented Lagrangian subproblem

$$\begin{aligned} & \underset{z \in \Omega}{\text{minimise}} \quad L_\varrho(z, \mu) & (2.6) \\ & \text{s. t.} \quad G(z) = 0 , \end{aligned}$$

with the partially augmented Lagrangian function defined as

$$L_\varrho(z, \mu) := J(z) + \left\langle \left(\mu + \frac{\varrho}{2} C(z) \right), C(z) \right\rangle , \quad (2.7)$$

where $\varrho > 0$ is a penalty coefficient. In NLP (2.6), only the coupling constraint $C(z)$ is penalised. The polyhedral constraint $z \in \Omega$ and equality constraint $G(z) = 0$ are kept as such in the augmented Lagrangian subproblem (2.7). This algorithmic choice is relevant, as in a large number of practical cases of interest, the augmented Lagrangian subproblem (2.6) is easier to solve than NLP (2.4). In particular, in this Section, we focus on the particular case in which the proximal operator of the indicator function of the set \mathcal{Z} defined in (2.2) is computationally cheap to evaluate. Such ‘prox-friendly’ examples are the following:

- Box constraints, $\Omega = \{z \in \mathbb{R}^n \mid \underline{z} \leq z \leq \bar{z}\}$ and $G = 0$.

- Binary constraints, $\Omega = \mathbb{R}^n$ and

$$G(z) = \begin{pmatrix} z_1(1-z_1) \\ \vdots \\ z_n(1-z_n) \end{pmatrix}$$

- Euclidean norm constraints, $\Omega = \mathbb{R}^n$ and

$$G(z) = \begin{pmatrix} \|z_1\|_2^2 - R_1 \\ \|z_2\|_2^2 - R_2 \end{pmatrix}, \quad (2.8)$$

with $R_1, R_2 > 0$.

It worth noting that the last two instances of ‘prox-friendly’ operators are nonconvex. We intend to solve the augmented Lagrangian subproblem by means of a decomposition approach, as detailed later in paragraph 2.1.2.

The phases of our partial augmented Lagrangian method are stated in Algorithm 1 below.

Algorithm 1 Method of multipliers with partial constraint relaxation

- 1: **Input:** initial guess $((z^0)^\top, (\mu^0)^\top)^\top \in \mathbb{R}^{n+p}$,
 - 2: initial tolerance on optimality of augmented Lagrangian subproblem $\epsilon^{(0)} > 0$,
 - 3: multiplicative coefficients $\beta_\rho > 1$ and $\beta_\epsilon < 1$, initial penalty $\rho^{(0)} > 1$.
 - 4: **Initialization:** $z \leftarrow z^0, \mu \leftarrow \mu^0, \rho \leftarrow \rho^{(0)}, \epsilon \leftarrow \epsilon^{(0)}, k \leftarrow 0$.
 - 5: Find $z \in \mathcal{Z}$ such that $d(0, \nabla L_\rho(z, \mu) + \mathcal{N}_{\mathcal{Z}}(z)) \leq \epsilon$
 - 6: Update multiplier estimate $\mu \leftarrow \mu + \rho C(z)$ and penalty coefficient $\rho \leftarrow \beta_\rho \rho$
 - 7: Shrink tolerance $\epsilon \leftarrow \beta_\epsilon \epsilon$
 - 8: Set $k \leftarrow k + 1$, go to 5.
-

The procedure starts from a primal-dual initial guess $((z^0)^\top, (\mu^0)^\top)^\top$, an initial tolerance $\epsilon^{(0)}$ on the satisfaction of the optimality conditions in the augmented Lagrangian subproblem (2.6) and an initial penalty $\rho^{(0)}$. Given a positive tolerance ϵ , we define an ϵ -critical point of the function

$$L_\rho(\cdot, \mu) + \iota_{\mathcal{Z}}, \quad (2.9)$$

where the indicator function $\iota_{\mathcal{Z}}$ is defined in (1.1), by

$$d(0, \nabla L_\rho(z, \mu) + \mathcal{N}_{\mathcal{Z}}(z)) \leq \epsilon. \quad (2.10)$$

At every iteration of Algorithm 1 (line 5), an ϵ -critical point of the function (2.9) is computed by means of an iterative procedure, which is the alternating minimisation technique later described

in paragraph 2.1.2. Once the stopping criterion (2.10) is satisfied, the multiplier estimate μ is updated in a first-order fashion, the penalty coefficient ϱ is increased and the criticality tolerance ϵ is shrunk. In practice, the method proceeds until a specified level of feasibility, measured in terms of the euclidean norm, of the coupling function C is obtained. In the next paragraph, under standard assumptions, we show that the iterative procedure converges to a KKT point of the nonconvex program (2.1), and thus Algorithm 1 terminates.

2.1.1.2 Convergence analysis

Under standard assumptions on the nonconvex programs (2.4) and (2.6), we prove local convergence of Algorithm 1 to a first-order critical point of (2.4). Our analysis is along the lines of [16]. However, the mechanism for handling inequality constraints differs from [16], in which squared slack variables are introduced. In [8], a proof of global convergence of a method of multipliers close to Algorithm 1 is given under weak constraint qualification. Yet, the updates of the Lagrange multipliers estimates and the penalty coefficient differ from Algorithm 1. In particular, the dual estimates are projected onto a sufficiently large box at every outer iteration. This ensures boundedness of the sequence of multipliers. Instead, we assume that the sequence of dual estimates is bounded. In fact, one should point out that the projection mechanism of [8] could be directly applied in Algorithm 1. Moreover, the analysis of [8] relies on the fact that the augmented Lagrangian subproblem has an approximate KKT point for all dual iterates and penalty parameters, which is case dependent. If this is not the case, the algorithm proposed in [8] is aborted. In fact, it is worth noting that without this tweak, global convergence could not be guaranteed. In conclusion, Algorithm 1 should be turned into Algorithm 3.1 in [8] in order to obtain theoretical guarantees of global convergence. However, our local analysis of Algorithm 1 brings up important concepts, such as strong regularity [137], which is a cornerstone of Chapter 3. Therefore, we have chosen not to modify Algorithm 1 using the tweaks of [8].

We first require the problem functions to be sufficiently smooth.

Assumption 2.2 (Smoothness). *The functions J , C and G are twice continuously differentiable in an open set containing Ω .*

The Lagrangian of NLP (2.4) is a twice continuously differentiable function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}_+^m \rightarrow \mathbb{R}$ defined as

$$\mathcal{L}(z, \mu, \nu, \lambda) := J(z) + \langle \mu, C(z) \rangle + \langle \nu, G(z) \rangle + \langle \lambda, (Hz - h) \rangle \quad .$$

Under the Linear Independence Constraint Qualification (LICQ), it can be proven that a first-order critical point of (2.4) satisfies the well-known Karush-Kuhn-Tucker (KKT) conditions [119].

Assumption 2.3 (Linear Independence Constraint Qualification). *For all first-order critical points z^* of problem (2.4), the matrix*

$$\left[\nabla C(z^*)^\top, \nabla G(z^*)^\top, H_{\mathcal{A}}^\top \right]^\top$$

is full row-rank, with $H_{\mathcal{A}}$ the submatrix of H whose rows are the rows of H associated with the indices $i \in \{1, \dots, m\}$ such that $\langle h_i, z^ \rangle - \omega_i = 0$.*

Thus, if z^* is a critical point of (2.4), then it satisfies the KKT conditions, that is there exists multipliers $\mu^* \in \mathbb{R}^p$, $\nu^* \in \mathbb{R}^q$ and $\lambda^* \in \mathbb{R}^m$ such that

$$\begin{cases} \nabla_z \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) = 0 \\ C(z^*) = 0, G(z^*) = 0 \\ 0 \leq \lambda^* \perp (\omega - Hz^*) \geq 0 \end{cases} \quad (2.11)$$

The KKT relation (2.11) can be rewritten as a generalised equation in the primal-dual space $\mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q$,

$$0 \in \mathcal{F}(w^*) + \mathcal{N}_{\Omega \times \mathbb{R}^p \times \mathbb{R}^q}(w^*) \quad , \quad (2.12)$$

with $w^* = ((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top)^\top$ and where

$$\mathcal{F}(w) := \begin{pmatrix} \nabla J(z) + \nabla C(z)^\top \mu + \nabla G(z)^\top \nu \\ C(z) \\ G(z) \end{pmatrix} \quad ,$$

with $w = (z^\top, \mu^\top, \nu^\top)^\top$. In the remainder, we call a primal-dual point a first-order critical point or a KKT point without distinction. The analysis that follows strongly relies on the strong second-order optimality conditions [119].

Assumption 2.4 (Strong second-order optimality condition). *Problem (2.4) has a KKT point*

$$((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$$

such that

$$\langle p, \nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) p \rangle > 0 \quad , \quad (2.13)$$

for all $p \in \mathbb{R}^n \setminus \{0\}$ such that

$$\nabla C(z^*)p = 0, \quad \nabla G(z^*)p = 0, \quad H_{A_+}p = 0, \quad ,$$

with H_{A_+} the submatrix of H whose rows are the rows of H associated with the indices i in $\{1, \dots, m\}$ such that $\langle h_i, z^* \rangle - \omega_i = 0$ and $\lambda_i^* > 0$.

We first prove that for a sufficiently large penalty coefficient ϱ and for an appropriate multiplier μ , the augmented Lagrangian subproblem (2.6) has a unique solution in the neighbourhood of a KKT point of NLP (2.4). For this, we require constraint qualification of the set \mathcal{Z} , which is the constraint set of the augmented Lagrangian subproblem (2.6).

Assumption 2.5 (Mangasarian-Fromowitz constraint qualification). *At all points $z \in \mathcal{Z}$, the matrix $\nabla G(z)$ is full-row rank and there exists a vector $w \in \mathbb{R}^n \setminus \{0\}$ such that*

$$\begin{cases} \nabla G(z)w = 0, \\ H_A w < 0, \end{cases} \quad (2.14)$$

where H_A is the submatrix of H , whose rows correspond to the active constraints at z .

Lemma 2.1. *Assume that problem (2.4) satisfies Assumptions 2.2, 2.3, 2.4 and 2.5. Let*

$$((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$$

be a KKT point of NLP (2.4). There exists a positive scalar $\bar{\varrho}$, radii $\kappa > 0$ and $\delta > 0$ such that for all $\varrho \geq \bar{\varrho}$ and all $\bar{\mu} \in \mathbb{R}^p$ such that $\|\bar{\mu} - \mu^*\|_2 < \delta\varrho$, the subproblem

$$\underset{z \in \mathcal{Z}}{\text{minimise}} \quad L_\varrho(z, \bar{\mu}), \quad (2.15)$$

has a unique critical point in $\mathcal{B}(z^*, \kappa)$.

Proof. Let $\bar{\mu} \in \mathbb{R}^p$ and $\varrho > 0$. The optimality condition of (2.15) is

$$0 \in \nabla_z L_\varrho(z, \bar{\mu}) + \mathcal{N}_{\mathcal{Z}}(z), \quad (2.16)$$

However, by Assumption 2.5, Farkas' lemma and Lemma 1.3,

$$\mathcal{N}_{\mathcal{Z}}(z) = \{ \nabla G(z)^\top \nu + H^\top \lambda : \nu \in \mathbb{R}^q, \lambda \in \mathbb{R}_+^m \}.$$

Hence, the first-order optimality condition (2.16) is equivalent to the existence of $\nu \in \mathbb{R}^q$ and

$\lambda \in \mathbb{R}_+^m$ such that

$$\begin{cases} 0 \in \nabla_z L_\varrho(z, \bar{\mu}) + \nabla G(z)^\top \nu + H^\top \lambda \\ 0 = G(z) \\ 0 \leq \omega - Hz \end{cases} \quad (2.17)$$

Let $\mathcal{G}_\varrho : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^m \times \mathbb{R}^p \rightarrow \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^m$ be defined as

$$\mathcal{G}_\varrho(z, \mu, \nu, \pi) := \begin{pmatrix} \nabla_z \mathcal{L}(z, \mu, \nu, \lambda) \\ \frac{\mu - \mu^*}{\varrho} - C(z) - \pi \\ -G(z) \\ \omega - Hz \end{pmatrix},$$

where $\pi \in \mathbb{R}^p$ should be interpreted as a parameter. Consider the parametric generalised equation

$$0 \in \mathcal{G}_\varrho(z, \mu, \nu, \lambda, \pi) + \mathcal{N}_{\mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}_+^m}(z, \mu, \nu, \lambda) \quad (2.18)$$

It is easy to see that $(z^\top, \nu^\top, \lambda^\top)^\top$ satisfies the optimality condition (2.17) if and only if

$$(z^\top, \bar{\mu}^\top + \varrho C(z)^\top, \nu^\top, \lambda^\top)^\top$$

satisfies (2.18) with $\pi = \bar{\mu} - \mu^*/\varrho$. Moreover, $((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$ is a solution of (2.18) for $\pi = 0$. Thus, by Theorem 1.2, the result follows if we can show that

$$0 \in \mathcal{G}_\varrho(z, \mu, \nu, \lambda, 0) + \mathcal{N}_{\mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}_+^m}(z, \mu, \nu, \lambda) \quad (2.19)$$

is strongly regular at $((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$. Consider the matrix

$$\begin{bmatrix} \nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) & \nabla C(z^*)^\top & \nabla G(z^*)^\top & H_{\mathcal{A}_+}^\top \\ -\nabla C(z^*) & I_p/\varrho & 0 & 0 \\ -\nabla G(z^*) & 0 & 0 & 0 \\ -H_{\mathcal{A}_+} & 0 & 0 & 0 \end{bmatrix}, \quad (2.20)$$

where $H_{\mathcal{A}_+}$ is the submatrix of H , whose rows correspond to the indices i of the active constraints at z^* such that $\lambda_i^* > 0$. It readily follows from Assumptions 2.3 and 2.4, and Lemma 1.4, that the matrix (2.20) is nonsingular for any ϱ sufficiently large. To show this, take vectors u_z, u_μ, u_ν and

u_λ of appropriate dimensions such that

$$\begin{cases} \nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) u_z + \nabla C(z^*)^\top u_\mu + \nabla G(z^*)^\top u_\nu + H_{\mathcal{A}_+}^\top u_\lambda = 0 \\ -\nabla C(z^*) u_z + u_\mu / \varrho = 0 \\ -\nabla G(z^*) u_z = 0 \\ -H_{\mathcal{A}_+} u_z = 0 \end{cases} \quad (2.21)$$

From the strong second-order optimality Assumption 2.4 and Lemma 1.4, it follows that there exists a positive scalar $\bar{\varrho}$ such that for any $\varrho > \bar{\varrho}$, the matrix

$$\nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) + \varrho \nabla C(z^*)^\top \nabla C(z^*) + \varrho \nabla G(z^*)^\top \nabla G(z^*) + \varrho H_{\mathcal{A}_+}^\top H_{\mathcal{A}_+}$$

is positive definite. Then, it can be deduced from (2.21) that

$$\begin{bmatrix} \nabla G(z^*) \\ H_{\mathcal{A}_+} \end{bmatrix} \left(\nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) + \varrho \nabla C(z^*)^\top \nabla C(z^*) + \varrho \nabla G(z^*)^\top \nabla G(z^*) + \varrho H_{\mathcal{A}_+}^\top H_{\mathcal{A}_+} \right)^{-1} \begin{bmatrix} \nabla G(z^*) \\ H_{\mathcal{A}_+} \end{bmatrix}^\top \begin{pmatrix} u_\nu \\ u_\lambda \end{pmatrix} = 0 ,$$

which implies that

$$\begin{bmatrix} \nabla G(z^*) \\ H_{\mathcal{A}_+} \end{bmatrix}^\top \begin{pmatrix} u_\nu \\ u_\lambda \end{pmatrix} = 0 .$$

By means of Assumption 2.3, one can conclude that $u_\nu = 0$ and $u_\lambda = 0$. Using (2.21), it is then easy to show that $u_z = 0$ and $u_\mu = 0$.

Now consider the Schur complement

$$\begin{bmatrix} H_{\mathcal{A}_0}^\top \\ 0 \\ 0 \\ 0 \end{bmatrix}^\top \begin{bmatrix} \nabla_{z,z}^2 \mathcal{L}(z^*, \mu^*, \nu^*, \lambda^*) & \nabla C(z^*)^\top & \nabla G(z^*)^\top & H_{\mathcal{A}_+}^\top \\ -\nabla C(z^*) & I_p / \varrho & 0 & 0 \\ -\nabla G(z^*) & 0 & 0 & 0 \\ -H_{\mathcal{A}_+} & 0 & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} H_{\mathcal{A}_0}^\top \\ 0 \\ 0 \\ 0 \end{bmatrix} , \quad (2.22)$$

where $H_{\mathcal{A}_0}$ is the submatrix of H , whose rows correspond to the indices i of the active constraints at z^* such that $\lambda_i^* = 0$. Following the same argument as to show that (2.20) is nonsingular, we can prove that the Schur complement (2.22) is positive definite for $\varrho > \bar{\varrho}$. Hence, by the necessary and sufficient conditions of Section 4 in [137], we obtain that the generalised equation (2.19) is strongly regular at $((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$ if $\varrho > \bar{\varrho}$. This yields radii $\delta > 0$ and $\kappa > 0$ such that for

all $\bar{\mu} \in \mathbb{R}^p$ satisfying $\|\bar{\mu} - \mu^*\|_2 / \varrho < \delta$, there exists a unique $z \in \mathcal{B}(z^*, \kappa) \cap \mathcal{Z}$ satisfying (2.16). \square

We are now assured that if the sequence of multipliers and penalty coefficients satisfy the conditions of Lemma 2.1, then Algorithm 1 is well-defined, that is, at every iteration k , there exists a primal point z^k satisfying

$$d(0, \nabla_z L_{\varrho^{(k)}}(z^k, \bar{\mu}^k) + \mathcal{N}_{\mathcal{Z}}(z^k)) \leq \epsilon ,$$

given $\epsilon > 0$. Local convergence to the KKT point

$$((z^*)^\top, (\mu^*)^\top, (\nu^*)^\top, (\lambda^*)^\top)^\top$$

defined in Lemma 2.1, follows using a limit point argument and assuming that the iterates stay within the right neighbourhood of the KKT point. Such a localisation condition is difficult to enforce in practice, as it requires knowledge of the radius $\kappa > 0$. Before stating the convergence result, we need the following instrumental Lemma. It is also worth noting that the constraint set \mathcal{Z} is closed, since G is continuous and Ω is polyhedral.

Lemma 2.2. *Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ denote a continuously differentiable function, and Ω denote a closed set in \mathbb{R}^n . Given $\epsilon > 0$ and $z \in \Omega$, the following holds*

$$d(0, \nabla f(z) + \mathcal{N}_{\Omega}(z)) \leq \epsilon \iff \exists v \in \mathbb{R}^n \text{ such that } \|v\|_2 \leq \epsilon \text{ and } -\nabla f(z) \in \mathcal{N}_{\Omega}(z) + v .$$

Proof. This is a direct consequence of the definition of the distance to a set as an infimum, and the basic properties of the infimum. \square

Theorem 2.1 (Local convergence to a KKT point). *Assume that Assumptions 2.3, 2.5 and 2.4 hold. Assume that $\varrho^{(0)} > \bar{\varrho}$ and that for all $k \geq 0$,*

$$\|z^k - z^*\|_2 < \kappa \text{ and } \|\mu^k - \mu^*\|_2 < \delta \varrho^{(k)} . \quad (2.23)$$

If the dual sequence $\{\mu^k\}$ is bounded and all limit points of the primal sequence $\{z^k\}$ generated by Algorithm 1 are regular, which means that they satisfy Assumption 2.3, then $\{z^k\}$ converges to z^ and $\{\mu^k + \varrho^{(k)} C(z^k)\}$ converges to μ^* .*

Proof. As the sequence $\{z^k\}$ is bounded, by Weierstrass' theorem, it possesses a subsequence $\{z^{k_l}\}$ such that

$$z^{k_l} \rightarrow \tilde{z} ,$$

where $\tilde{z} \in \Omega \cap \mathcal{B}(z^*, \kappa)$. From Algorithm 1 and Lemma 2.2, for all $l \geq 1$,

$$0 = \nabla_z L_{\varrho^{(k_l)}}(z^{k_l}, \mu^{k_l}) + y^{k_l} + v^{k_l} ,$$

where $y^{k_l} \in \mathcal{N}_{\mathcal{Z}}(z^{k_l})$ and $\|v^{k_l}\|_2 \leq \epsilon^{k_l}$. However, by Assumption 2.5, Farkas' lemma and Lemma 1.3, there exists multipliers $\nu^{k_l} \in \mathbb{R}^q$ and $\lambda^{k_l} \in \mathbb{R}_+^m$ such that

$$0 = \nabla_z L_{\varrho^{(k_l)}}(z^{k_l}, \mu^{k_l}) + \nabla G(z^{k_l})^\top \nu^{k_l} + \sum_{i \in \mathcal{A}(z^{k_l})} [\lambda^{k_l}]_i h_i + v^{k_l} . \quad (2.24)$$

For all $k \geq 1$, define the multiplier $\hat{\mu}^k$ as

$$\hat{\mu}^k := \mu^k + \varrho^{(k)} C(z^k) .$$

We first show that the sequences of multipliers $\{\nu^{k_l}\}$ and $\{\lambda^{k_l}\}$ are bounded. For the sake of contradiction, assume that the sequence $\{S^{(k)}\}$ defined by

$$S^{(k)} := \max \{ \|\hat{\mu}^k\|_\infty, \|\nu^k\|_\infty, \|\lambda^k\|_\infty \}$$

has a subsequence that tends to infinity. Note that from the expression of $\hat{\mu}^k$, $S^{(k)} \neq 0$. We have that

$$-\frac{\nabla J(z^{k_l}) + v^{k_l}}{S^{(k_l)}} = \nabla C(z^{k_l})^\top \frac{\hat{\mu}^{k_l}}{S^{(k_l)}} + \nabla G(z^{(k_l)})^\top \frac{\nu^{k_l}}{S^{(k_l)}} + \sum_{i \in \mathcal{A}(z^{k_l})} \frac{[\lambda^{k_l}]_i}{S^{(k_l)}} h_i , \quad (2.25)$$

with $[\lambda^{k_l}]_i = 0$ if $i \notin \mathcal{A}(z^{k_l})$. However, as the subsequence $\{z^{k_l}\}$ tends to \tilde{z} , by definition of the active-set $\mathcal{A}(z^{k_l})$, there exists $l_0 \geq 1$ such that for $l \geq l_0$,

$$\mathcal{A}(z^{k_l}) \subset \mathcal{A}(\tilde{z}) . \quad (2.26)$$

Moreover, as the sequences $\{\hat{\mu}^{k_l}/S^{(k_l)}\}$, $\{\nu^{k_l}/S^{(k_l)}\}$ and $\{\lambda^{k_l}/S^{(k_l)}\}$ are bounded, the functions J , C and G are continuously differentiable, and v^{k_l} goes to zero, by extracting an appropriate subsequence, one obtains the existence of $\hat{\mu} \in \mathbb{R}^p$, $\hat{\nu} \in \mathbb{R}^q$ and $\hat{\lambda} \in \mathbb{R}_+^m$, among which at least one of them is nonzero, such that

$$0 = \nabla C(\tilde{z})^\top \hat{\mu} + \nabla G(\tilde{z})^\top \hat{\nu} + \sum_{i \in \mathcal{A}(\tilde{z})} [\hat{\lambda}]_i h_i ,$$

which contradicts the assumption that \tilde{z} is regular. Subsequently, the sequences $\{\nu^{k_l}\}$ and $\{\lambda^{k_l}\}$

are bounded. Thus, without loss of generality, as an appropriate subsequence can always be extracted, there exists $\tilde{\nu} \in \mathbb{R}^q$ and $\tilde{\lambda} \in \mathbb{R}_+^m$ such that

$$\nu^{k_l} \rightarrow \tilde{\nu} \text{ and } \lambda^{k_l} \rightarrow \tilde{\lambda} .$$

However, as \tilde{z} is regular, there exists $l_1 \geq 1$ such that for all $l \geq 1$, the matrix

$$\nabla C(z^{k_l}) \nabla C(z^{k_l})^\top$$

is invertible, and subsequently,

$$\hat{\mu}^{k_l} = - \left(\nabla C(z^{k_l}) \nabla C(z^{k_l})^\top \right)^{-1} \nabla C(z^{k_l}) \left(\nabla J(z^{k_l}) + \nabla G(z^{k_l})^\top \nu^{k_l} + \sum_{i \in \mathcal{A}(z^{k_l})} [\lambda^{k_l}]_i h_i + v^{k_l} \right) .$$

By continuity of ∇J and ∇G , by convergence of $\{\nu^{k_l}\}$ and $\{\lambda^{k_l}\}$ and since v^{k_l} tends to zero, one can conclude that $\{\hat{\mu}^{k_l}\}$ converges to $\tilde{\mu}$ such that

$$\tilde{\mu} := - \left(\nabla C(\tilde{z}) \nabla C(\tilde{z})^\top \right)^{-1} \nabla C(\tilde{z}) \left(\nabla J(\tilde{z}) + \nabla G(\tilde{z})^\top \tilde{\nu} + H^\top \tilde{\lambda} \right) .$$

By taking limit in (2.24), one obtains that

$$\nabla J(\tilde{z}) + \nabla C(\tilde{z})^\top \tilde{\mu} + \nabla G(\tilde{z})^\top \tilde{\nu} + \sum_{i \in \mathcal{A}(\tilde{z})} [\tilde{\lambda}]_i h_i = 0 . \quad (2.27)$$

The sequences $\{\hat{\mu}^{k_l}\}$ and $\{\mu^{k_l}\}$ are bounded. Hence, as $\rho^{(k_l)} \rightarrow +\infty$, the limit point \tilde{z} is feasible, that is $C(\tilde{z}) = 0$. Together with (2.27), this implies that $(\tilde{z}^\top, \tilde{\mu}^\top, \tilde{\nu}^\top, \tilde{\lambda}^\top)^\top$ is a KKT point of NLP (2.4). However, by the strong second-order optimality condition (Assumption 2.4), z^* is the unique critical point of NLP (2.4) in $\mathcal{B}(z^*, \kappa)$. Subsequently, $\tilde{z} = z^*$, and by independence of the constraint gradients at z^* (Assumption 2.3), $\tilde{\mu} = \mu^*$, $\tilde{\nu} = \nu^*$ and $\tilde{\lambda} = \lambda^*$. This concludes the proof. \square

Remark 2.1. *As mentioned earlier, ensuring the localisation condition $\|z^k - z^*\|_2 < \kappa$ is not obvious. However, warm-starting an inner method for solving an augmented Lagrangian subproblem on the output of the previous subproblem tends to make this requirement satisfied.*

Remark 2.2. *Clearly, a limitation of our analysis is the assumption that the dual sequence $\{\mu^k\}$ is bounded. However, this can be guaranteed algorithmically by projecting the dual iterate μ^k onto a sufficiently large box containing the origin at every outer iteration [8]. It is also worth noting the*

the Mangasarian-Fromowitz constraints qualification imply that the set of Lagrange multipliers is bounded at a critical point. Hence, an unbounded dual sequence is a sign that these conditions are most likely violated.

Thus, only local convergence of Algorithm 1 can be derived. This is due to conditions (2.23). However, it is worth noting that when all constraints of the augmented Lagrangian subproblem are linear, global convergence can be guaranteed by modifying the dual and penalty updates [34, 31]. Thus, it appears that all globally convergent augmented Lagrangian methods are based either on a different dual update than the classical first-order update in Algorithm 1, or a different update of the penalty coefficient.

Remark 2.3. *The local convergence analysis presented above allowed us to introduce generalised equations (2.18), which turns out to be a key ingredient in order to analyse parametric properties of the solution of a nonlinear program, as in Chapter 3.*

2.1.2 Splitting the augmented Lagrangian subproblem

From a distributed optimization perspective, the dual update in Algorithm 1 acts as a coordinator between the subvariables z_1 and z_2 . In practice, it requires a local exchange of information between computing nodes associated with variables z_1 and z_2 . Increasing the penalty coefficient ρ and shrinking the criticality tolerance ϵ does not need to be performed on a central unit, but requires synchronisation between the computing nodes by means of a global clock. However, it remains to compute an ϵ -critical point of the nonconvex augmented Lagrangian subproblem (2.6) in a distributed manner. This is far from obvious and has not yet been addressed in the literature in a nonconvex setting. To cope with this problem, we propose an alternating projected gradient method. Alternating minimisation techniques, also known as Gauss-Seidel or Block-Coordinated Descent (BCD) schemes, are a method of choice in distributed optimisation, as computations can be readily parallelised under some structural assumptions on the coupling between subvariables [17]. In a nonconvex setting, alternating minimisation methods suffer from a lack of popularity, partly due to their ambiguous convergence properties. Indeed, examples can be found, in which a Gauss-Seidel procedure cycles infinitely without approaching a critical point [123]. Fortunately, conditions on the problem structure as well as algorithmic refinements can be derived to guarantee global convergence to first-order critical points. In [150], the subsequence convergence of the iterates generated by a Gauss-Seidel scheme applied to a nonconvex function that consists of the sum of nonseparable differentiable and separable nondifferentiable summands is proven under some pseudoconvexity and quasiconvexity assumptions. A stronger convergence result has been recently derived in [11]. Under a more general assumption on the problem structure and by means of blockwise proximal regularisations, global convergence of an alternating minimisation procedure

to a critical point is proven [11]. The results of [11] have been extended to more general descent methods in [12]. The results presented next build upon some ideas of [11, 12]. It is worth mentioning that a well-known problem with Gauss-Seidel methods is their frequent non-convergence when the number of blocks is larger than two [76]. This issue is solved by enforcing a proximal regularisation on each block at every iteration [76, 11]. We use the same idea for the Algorithm described in the next paragraphs.

2.1.2.1 Algorithm formulation

Aiming at distributing computations in order to compute an iterate satisfying the stopping criterion (2.10), we apply Algorithm 2 below. It is essentially a special form of the very generic Proximal Alternating Linearised Minimisations (PALM) algorithm proposed by [21].

Algorithm 2 Projected Alternating Gradients

1: **Input:** Primal initial points $z_1^0 \in \mathbb{R}^{n_1}$ and $z_2^0 \in \mathbb{R}^{n_2}$, tolerance on criticality $\epsilon > 0$, initial curvature estimates ξ_1, ξ_2 .

2: $k \leftarrow 0$

3: **while** $d(0, \nabla_z L_\varrho(z^k, \mu) + \mathcal{N}_{\mathcal{Z}}(z^k)) > \epsilon$ **do**

4: Find $z_1^{k+1} \in P_{\mathcal{Z}_1} \left(z_1^k - \frac{1}{c_1^k} \nabla_{z_1} L_\varrho(z_1^k, z_2^k, \mu) \right)$, where $c_1^k > 0$ satisfies

$$L_\varrho(z_1^{k+1}, z_2^k, \mu) + \frac{\alpha_1^k}{2} \|z_1^{k+1} - z_1^k\|_2^2 \leq L_\varrho(z_1^k, z_2^k, \mu) + \langle \nabla_{z_1} L_\varrho(z_1^k, z_2^k, \mu), z_1^{k+1} - z_1^k \rangle + \frac{c_1^k}{2} \|z_1^{k+1} - z_1^k\|_2^2. \quad (2.28)$$

5: Find $z_2^{k+1} \in P_{\mathcal{Z}_2} \left(z_2^k - \frac{1}{c_2^k} \nabla_{z_2} L_\varrho(z_1^{k+1}, z_2^k, \mu) \right)$, where $c_2^k > 0$ satisfies

$$L_\varrho(z_1^{k+1}, z_2^{k+1}, \mu) + \frac{\alpha_2^k}{2} \|z_2^{k+1} - z_2^k\|_2^2 \leq L_\varrho(z_1^{k+1}, z_2^k, \mu) + \langle \nabla_{z_2} L_\varrho(z_1^{k+1}, z_2^k, \mu), z_2^{k+1} - z_2^k \rangle + \frac{c_2^k}{2} \|z_2^{k+1} - z_2^k\|_2^2. \quad (2.29)$$

6: $k \leftarrow k + 1$

7: **end while**

8: **Output:** z_1, z_2

The procedure starts from initial points $z_1^0 \in \mathcal{Z}_1$ and $z_2^0 \in \mathcal{Z}_2$. At every iteration of Algorithm 2, the subvariable z_1 is updated by means of a gradient projection step, for which the step-size $1/c_1$ sat-

ifies the coordinatewise sufficient decrease condition (2.28). Using the updated subvariable z_1 , the subvariable z_2 is modified in a similar manner, with a step-size $1/c_2$ satisfying the coordinatewise sufficient decrease condition (2.29). The sequences $\{\alpha_1^k\}$ and $\{\alpha_2^k\}$ are such that for all $k \geq 0$,

$$\alpha_1^k \in]\underline{\alpha}, \tilde{\alpha}[\text{ and } \alpha_2^k \in]\underline{\alpha}, \tilde{\alpha}[, \quad (2.30)$$

with $\tilde{\alpha} > \underline{\alpha} > 0$. They need to be specified in advance and act as tuning parameters. Nevertheless, we have observed that the best practical performance of Algorithm 2 is obtained with very small values of these coefficients. The sufficient decrease conditions (2.28) and (2.29) are obtained via a backtracking procedure, which is described below for the subvariable z_2 .

Algorithm 3 Backtracking procedure at subvariable z_2 and iteration k of Algorithm 2

Input: Variables z_1^{k+1} and z_2^k , initial curvature estimate ξ_2 .

Parameters: Multiplicative coefficient $\beta > 1$, regularisation coefficient α_2^k .

$c_2 \leftarrow \xi_2$

$$z_2 \in P_{\mathcal{Z}_2} \left(z_2^k - \frac{1}{c_2} \nabla_{z_2} L_\varrho (z_1^{k+1}, z_2^k, \mu) \right)$$

while $L_\varrho (z_1^{k+1}, z_2, \mu) + \frac{\alpha_2^k}{2} \|z_2 - z_2^k\|_2^2 > L_\varrho (z_1^{k+1}, z_2^k, \mu) + \langle \nabla_{z_2} L_\varrho (z_1^{k+1}, z_2^k, \mu), z_2 - z_2^k \rangle + \frac{c_2}{2} \|z_2 - z_2^k\|_2^2$ **do**

$c_2 \leftarrow \beta \cdot c_2$

$$z_2 \in P_{\mathcal{Z}_2} \left(z_2^k - \frac{1}{c_2} \nabla_{z_2} L_\varrho (z_1^{k+1}, z_2^k, \mu) \right)$$

end while

Output: $c_2^k \leftarrow c_2$

The backtracking procedure is similar for variable z_1 and takes variables z_1^k and z_2^k on input. For Algorithm 2 to be well-defined, one needs to make sure that Algorithm 3 terminates at every iteration of Algorithm 2. This is guaranteed under the condition that the gradient of the augmented Lagrangian is Lipschitz continuous on the subvariables z_1 and z_2 respectively.

Assumption 2.6 (Coordinate-wise Lipschitz continuity). *Given $z_1 \in \mathcal{Z}_1$, the coordinate gradient*

$$z_2 \mapsto \nabla_{z_2} L_\varrho (z_1, z_2, \mu)$$

is Lipschitz continuous on \mathcal{Z}_2 with modulus $\ell_2(z_1, \mu, \varrho)$. Lipschitz continuity also holds for the function

$$z_1 \mapsto \nabla_{z_1} L_\varrho (z_1, z_2, \mu)$$

with z_2 fixed in \mathcal{Z}_2 , and a modulus $\ell_1(z_2, \mu, \varrho)$.

We require the coordinate-wise Lipschitz constants defined in 2.6 to be upper bounded for all points of the sequence $\{z^k\}$.

Assumption 2.7 (Upper bounds on coordinate Lipschitz constants). *There exists scalars $\bar{\ell}_1(\mu, \varrho) > 0$ and $\bar{\ell}_2(\mu, \varrho) > 0$ such that for all $k \geq 0$,*

$$\ell_1(z_2^k, \mu, \varrho) < \bar{\ell}_1(\mu, \varrho) \text{ and } \ell_2(z_1^k, \mu, \varrho) < \bar{\ell}_2(\mu, \varrho) .$$

We also need to assume that the gradient of the augmented Lagrangian is Lipschitz continuous on all bounded subsets of \mathbb{R}^n .

Assumption 2.8 (Lipschitz continuity of gradient on bounded subsets). *Given any bounded subset \mathcal{S} of \mathbb{R}^n , the gradient $\nabla_z L_\varrho$ is Lipschitz continuous on \mathcal{S} with a Lipschitz constant denoted by $\ell_{\mathcal{S}}(\nabla_z L_\varrho)$.*

Lemma 2.3. *Assume that Assumption 2.6 holds. Let k denote an iteration index of Algorithm 2. There exists an integer $j_2^k \geq 0$ such that*

$$c_2^k = \beta^{j_2^k} \xi_2 ,$$

where $\xi_2 > 0$ is an initial estimate of the curvature coefficient c_2 , as shown in Algorithm 3.

Proof. As the function $\nabla_{z_2} L_\varrho(z_1^{k+1}, \cdot, \mu)$ is Lipschitz continuous by Assumption 2.6, the descent Lemma [17] yields

$$L_\varrho(z_1^{k+1}, z_2, \mu) \leq L_\varrho(z_1^{k+1}, z_2^k, \mu) + \langle \nabla_{z_2} L_\varrho(z_1^{k+1}, z_2^k, \mu), z_2 - z_2^k \rangle + \frac{\ell_2(z_1^{k+1}, \mu, \varrho)}{2} \|z_2 - z_2^k\|_2^2$$

for all $z_2 \in \mathcal{Z}_2$. Hence, by taking c_2 such that

$$c_2 > \ell_2(z_1^{k+1}, \mu, \varrho) + \alpha_2^k \tag{2.31}$$

the sufficient decrease condition (2.29) is fulfilled with

$$z_2^{k+1} \in P_{\mathcal{Z}_2} \left(z_2^k - \frac{1}{c_2} \nabla_{z_2} L_\varrho(z_1^{k+1}, z_2^k, \mu) \right) .$$

Condition (2.31) is met after at most

$$j_2^k := \left\lceil \frac{\log(\alpha_2^k + \ell_2(z_1^{k+1}, \mu, \varrho) / \xi_2)}{\log \beta} \right\rceil \tag{2.32}$$

iterations of the backtracking loop, which proves that Algorithm 3 terminates. \square

Remark 2.4. *A similar property holds for the backtracking procedure on subvariable z_1 with at most*

$$j_1^k := \left\lceil \frac{\log(\alpha_1^{k+\ell_1}(z_2^k, \mu, \rho)/\xi_1)}{\log \beta} \right\rceil \quad (2.33)$$

backtracking iterations, where ξ_1 is an initial guess of the curvature estimate c_1 .

Algorithm 2 iterates until the stopping criterion (2.10) is met. Next, using the results of [12], we show that Algorithm 2 terminates, and thus yields an ϵ -critical point of the partial augmented Lagrangian subproblem (2.6).

2.1.2.2 Convergence analysis

Our analysis consists in showing that the sequence of augmented Lagrangian values

$$\{L_\varrho(z_1^k, z_2^k, \mu) + \iota_{\mathcal{Z}}(z^k)\}$$

decreases of at least a fraction of $\|z^k - z^{k-1}\|_2^2$ at every iteration, and that a subgradient of $L_\varrho(\cdot, \mu) + \iota_{\mathcal{Z}}$ is bounded by $\|z^k - z^{k-1}\|_2$. It is worth noting that these two properties imply that if the sequence of iterates $\{z^k\}$ is bounded and the sequence of objectives $\{L_\varrho(z^k, \mu)\}$ is bounded below, then there exists a subsequence of $\{z^k\}$ that converges to a critical point of the function $L_\varrho(\cdot, \mu) + \iota_{\mathcal{Z}}$.

In the remainder of the proof, we assume that the Lagrange multiplier μ and the penalty coefficient ϱ are fixed. We first state the sufficient decrease property, which is fulfilled by the sequence of iterates generated by Algorithm 2.

Lemma 2.4 (Sufficient decrease in the augmented Lagrangian). *For all $k \geq 1$,*

$$L_\varrho(z_1^k, z_2^k, \mu) + \iota_{\mathcal{Z}}(z^k) + \frac{\alpha}{2} \|z^k - z^{k-1}\|_2^2 \leq L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu) + \iota_{\mathcal{Z}}(z^{k-1}) \quad (2.34)$$

Proof. This is a direct consequence of the definition of z_1^k and z_2^k as projected gradient steps onto the sets \mathcal{Z}_1 and \mathcal{Z}_2 respectively. Indeed, this implies that

$$\langle \nabla_{z_1} L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu), z_1^k - z_1^{k-1} \rangle + \frac{c_1^{k-1}}{2} \|z_1^k - z_1^{k-1}\|_2^2 \leq 0 ,$$

and that $\iota_{\mathcal{Z}_1}(z_1^k) = 0$. Putting this together with the sufficient decrease (2.28), this yields

$$L_\varrho(z_1^k, z_2^{k-1}, \mu) + \iota_{\mathcal{Z}_1}(z_1^k) + \frac{\alpha_1^{k-1}}{2} \|z_1^k - z_1^{k-1}\|_2^2 \leq L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu) + \iota_{\mathcal{Z}_1}(z_1^{k-1}) ,$$

as $\iota_{\mathcal{Z}_1}(z_1^{k-1}) = 0$. A similar reasoning for z_2^k gives

$$L_\varrho(z_1^k, z_2^k, \mu) + \iota_{\mathcal{Z}_2}(z_2^k) + \frac{\alpha_2^{k-1}}{2} \|z_2^k - z_2^{k-1}\|_2^2 \leq L_\varrho(z_1^k, z_2^{k-1}, \mu) + \iota_{\mathcal{Z}_2}(z_2^{k-1}) ,$$

and thus (2.34) immediately follows, as $\alpha_1^{k-1}, \alpha_2^{k-1} \geq \underline{\alpha}$. \square

We now derive an upper-bound on a subgradient vector of $L_\varrho(\cdot, \mu) + \iota_{\mathcal{Z}}$.

Lemma 2.5 (Relative error on subgradient of augmented Lagrangian). *Assume that the sequence $\{z^k\}$ generated by Algorithm 2 is bounded. Under Assumptions 2.6, 2.7 and 2.8, there exists a scalar $\Gamma(\mu, \varrho) > 0$ such that for all $k \geq 1$, there exists $v^k \in \mathcal{N}_{\mathcal{Z}}(z^k)$ such that*

$$\|v^k + \nabla_z L_\varrho(z^k, \mu)\|_2 \leq \Gamma(\mu, \varrho) \|z^k - z^{k-1}\|_2 . \quad (2.35)$$

Proof. From the definition of z_1^k and z_2^k in Algorithm 2,

$$z_1^k \in \operatorname{argmin}_{z_1 \in \mathcal{Z}_1} \langle \nabla_{z_1} L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu), z_1 - z_1^{k-1} \rangle + \frac{c_1^{k-1}}{2} \|z_1 - z_1^{k-1}\|_2^2$$

and

$$z_2^k \in \operatorname{argmin}_{z_2 \in \mathcal{Z}_2} \langle \nabla_{z_2} L_\varrho(z_1^k, z_2^{k-1}, \mu), z_2 - z_2^{k-1} \rangle + \frac{c_2^{k-1}}{2} \|z_2 - z_2^{k-1}\|_2^2 ,$$

which implies that there exists $v_1^k \in \mathcal{N}_{\mathcal{Z}_1}(z_1^k)$ and $v_2^k \in \mathcal{N}_{\mathcal{Z}_2}(z_2^k)$ such that

$$0 = v_1^k + \nabla_{z_1} L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu) + c_1^{k-1} (z_1^k - z_1^{k-1})$$

and

$$0 = v_2^k + \nabla_{z_2} L_\varrho(z_1^k, z_2^{k-1}, \mu) + c_2^{k-1} (z_2^k - z_2^{k-1}) .$$

Posing $v^k := \left((v_1^k)^\top, (v_2^k)^\top \right)^\top$,

$$\begin{aligned} \|v^k + \nabla_z L_\varrho(z^k, \mu)\|_2 &\leq (c_1^{k-1} + c_2^{k-1}) \|z^k - z^{k-1}\|_2 + \|\nabla_{z_1} L_\varrho(z_1^k, z_2^k, \mu) - \nabla_{z_1} L_\varrho(z_1^{k-1}, z_2^{k-1}, \mu)\|_2 \\ &\quad + \|\nabla_{z_2} L_\varrho(z_1^k, z_2^k, \mu) - \nabla_{z_2} L_\varrho(z_1^k, z_2^{k-1}, \mu)\|_2 \end{aligned}$$

However,

$$c_1^{k-1} = \beta^{j_1^{k-1}} \xi_1 \text{ and } c_2^{k-1} = \beta^{j_2^{k-1}} \xi_2 ,$$

where j_1^{k-1} and j_2^{k-1} are defined by (2.33) and (2.32) respectively. As the regularisation coefficients α_1^{k-1} and α_2^{k-1} are upper bounded by $\tilde{\alpha}$, and as $\ell_2(z_1^k, \mu, \varrho)$ is upper bounded by $\bar{\ell}_2(\mu, \varrho)$ (Assumption 2.7), one can conclude from the expressions of j_1^{k-1} and j_2^{k-1} that there exists $\bar{c}_1 > 0$ and $\bar{c}_2 > 0$ such that

$$c_1^{k-1} \leq \bar{c}_1 \text{ and } c_2^{k-1} \leq \bar{c}_2 ,$$

for all $k \geq 1$. However, the sequence $\{z^k\}$ is bounded, hence there exists a scalar $R > 0$ such that

$$z^k \in \mathcal{B}(0, R) ,$$

for all $k \geq 0$. Subsequently,

$$\|v^k + \nabla_z L_\varrho(z^k, \mu)\|_2 \leq (\bar{c}_1 + \bar{c}_2 + \ell_{\mathcal{B}(0,R)}(\nabla_z L_\varrho) + \bar{\ell}_2(\mu, \varrho)) \|z^k - z^{k-1}\|_2 ,$$

which concludes the proof. \square

We can now show that Algorithm 2 terminates for any criticality tolerance $\epsilon > 0$.

Theorem 2.2. *Assume that the augmented Lagrangian function $L_\varrho(\cdot, \mu)$ is bounded below on bounded subsets of \mathbb{R}^n . Assume that the sequence $\{z^k\}$ is bounded and that Assumptions 2.6, 2.7 and 2.8 hold. Given an arbitrary tolerance $\epsilon > 0$, Algorithm 2 terminates with a point z^k satisfying*

$$d(0, \nabla_z L_\varrho(z^k, \mu) + \mathcal{N}_{\mathcal{Z}}(z^k)) \leq \epsilon .$$

Proof. Combining Lemma 2.4 and Lemma 2.5, for all $k \geq 1$,

$$\|v^k + \nabla_z L_\varrho(z^k, \mu)\|_2 \leq \Gamma(\mu, \varrho) \sqrt{\frac{2(L_\varrho(z^{k-1}, \mu) - L_\varrho(z^k, \mu))}{\underline{\alpha}}} , \quad (2.36)$$

where $v^k \in \mathcal{N}_{\mathcal{Z}}(z^k)$. However, the series of nonnegative terms $\sum_{k \geq 1} (L_{\varrho}(z^{k-1}, \mu) - L_{\varrho}(z^k, \mu))$ is bounded, as the sequence $\{L_{\varrho}(z^k, \mu)\}$ converges, since it decreases and is bounded below by assumption. Hence,

$$(L_{\varrho}(z^{k-1}, \mu) - L_{\varrho}(z^k, \mu)) \rightarrow 0, \quad (2.37)$$

and then, by inequality (2.36),

$$d(0, \nabla_z L_{\varrho}(z^k, \mu) + \mathcal{N}_{\mathcal{Z}}(z^k)) \rightarrow 0,$$

which yields the result. \square

In conclusion, Algorithm 2 can be used as an inner solver in Algorithm 1, as the stopping criterion at line 5 of Algorithm 1 is guaranteed to be satisfied after a sufficiently large number of iterations in Algorithm 2. Recall that the purpose of Algorithm 2 is to decompose the augmented Lagrangian subproblem and thus allow for distributed computations. To clarify this point, we present the steps of the coupling between Algorithm 1 (coordination) and Algorithm 2 (decomposition) on a distributed problem with network constraints.

2.1.3 A coordination-decomposition algorithm

We now formulate our coordination-decomposition algorithm (Algorithms 1 and 2) in the case of several sub-variables coupled by network constraints. It is implicitly assumed that each sub-variable is associated with a computing node. In this paragraph, our goal is to show when local computations and communications between nodes happen as the algorithm proceeds. More precisely, we aim at solving the following class of distributed nonconvex programs with separable objective subject to partially separable equality constraints and separable constraints

$$\begin{aligned} & \underset{z_1, \dots, z_N}{\text{minimise}} \quad \sum_{i=1}^N J_i(z_i) & (2.38) \\ & \text{s. t. } C_i(z_i, z_{\mathcal{V}_i}) = 0, \quad i \in \{1, \dots, N\} \\ & \quad z_i \in \mathcal{Z}_i, \end{aligned}$$

with $N \geq 2$. For each $i \in \{1, \dots, N\}$, $z_{\mathcal{V}_i}$ denotes the variables z_j which are coupled with variable z_i and are called its *neighbours*. The Lagrange multiplier associated with the equality constraint $C_i(z_i, z_{\mathcal{V}_i}) = 0$ is denoted by μ_i . Without loss of generality, we assume that all coupling constraints involving the sub-variable z_i are gathered in $C_i(z_i, z_{\mathcal{V}_i})$. The sets \mathcal{Z}_i are closed and ‘prox-friendly’, that is the euclidean projection onto the set \mathcal{Z}_i can be computed in closed-form. The augmented

Lagrangian subproblem associated with program (2.38) is

$$\underset{z_1 \in \mathcal{Z}_1, \dots, z_N \in \mathcal{Z}_N}{\text{minimise}} \sum_{i=1}^N J_i(z_i) + \left\langle \left(\mu_i + \frac{\rho}{2} C_i(z_i, z_{\mathcal{V}_i}) \right), C_i(z_i, z_{\mathcal{V}_i}) \right\rangle . \quad (2.39)$$

The coordination-decomposition method is described in Algorithm 4 below. Algorithm 4 does

Algorithm 4 Coordination-decomposition algorithm

```

1: Input: Initial guesses  $\left( \left( z_1^{(0)} \right)^\top, \left( \mu_1^{(0)} \right)^\top \right)^\top, \dots, \left( \left( z_N^{(0)} \right)^\top, \left( \mu_N^{(0)} \right)^\top \right)^\top$ 
2:     multiplicative coefficient  $\beta_\rho > 1$ , initial penalty  $\rho^{(0)} > 1$ .
3: Initialisation:  $\left( (z_i)^\top, (\mu_i)^\top \right)^\top \leftarrow \left( (z_i^0)^\top, (\mu_i^0)^\top \right)^\top$  for all  $i \in \{1, \dots, N\}$ 
4:  $k \leftarrow 0$ 
5: while Outer stopping criterion not met do            $\triangleright$  Central communication or synchronisation
6:     Decomposition
7:     while Inner stopping criterion not met do        $\triangleright$  Central communication or synchronisation
8:         For all nodes  $i \in \{1, \dots, N\}$ ,
9:         Gather  $z_{\mathcal{V}_i}$  from neighbours of node  $i$             $\triangleright$  Local communications
10:        Compute  $z_i \in P_{\mathcal{Z}_i} \left( z_i - \frac{1}{c_i} \left( \nabla J_i(z_i) + \nabla_{z_i} C_i(z_i, z_{\mathcal{V}_i})^\top (\mu_i + \rho C_i(z_i, z_{\mathcal{V}_i})) \right) \right)$ 
11:    end while
12:    Coordination
13:    Gather  $z_{\mathcal{V}_i}$  from neighbours of node  $i$             $\triangleright$  Local communications
14:    Compute  $\mu_i \leftarrow \mu_i + \rho C_i(z_i, z_{\mathcal{V}_i})$ 
15:    Update
16:     $\rho \leftarrow \beta_\rho \rho$ 
17:     $k \leftarrow k + 1$ 
18: end while
    
```

not involve any matrix factorisation, which makes it a *matrix free* method. This feature is highly suitable for distributed computations. From this perspective, the most expensive steps are at lines 9 and 13, as they require local exchange of data between neighbouring nodes. Communications with a central unit or synchronisation of the computing nodes may be required by the stopping criteria for the dual (outer) loop (line 5) and the primal (inner) loop (line 7). To clarify this point, we discuss the following three stopping criteria:

1. Criticality-based criterion, $d(0, \nabla_z L_\rho(z, \mu) + \mathcal{N}_{\mathcal{Z}}(z)) \leq \epsilon$: With this condition, local convergence of the dual loop is guaranteed, as proven earlier in this chapter. It is equivalent to the existence of a vector v in $\mathcal{N}_{\mathcal{Z}}(z)$ such that

$$\|v + \nabla_z L_\rho(z, \mu)\|_2 \leq \epsilon . \quad (2.40)$$

In practice, one can solve

$$\underset{v \in \mathcal{N}_{\mathcal{Z}}(z)}{\text{minimise}} \quad \|v + \nabla_z L_\varrho(z, \mu)\|_2 \quad ,$$

which is equivalent to

$$\underset{v_1 \in \mathcal{N}_{\mathcal{Z}_1}(z_1), \dots, v_N \in \mathcal{N}_{\mathcal{Z}_N}(z_N)}{\text{minimise}} \quad \sum_{i=1}^N \|v_i + \nabla_{z_i} L_\varrho(z, \mu)\|_2^2 \quad ,$$

by Proposition 6.41 in [138]. This task amounts to finding a solution to the following problems locally on each node

$$\underset{z_i \in \mathcal{N}_{\mathcal{Z}_i}(z_i)}{\text{minimise}} \quad \|v_i + \nabla_{z_i} L_\varrho(z, \mu)\|_2$$

and adding up the local objectives on a central unit at every primal iteration. A drawback of this stopping condition is that one needs to specify a sequence of criticality tolerances ϵ for each outer iteration. This is problem dependent and requires tuning. A more systematic way of updating the criticality tolerance on the augmented Lagrangian subproblem is proposed by [34], which is a cornerstone of the well-known LANCELOT software. However, the adaptation of the criticality tolerance in [34] is based on the level of satisfaction of the equality constraint that are relaxed in the augmented Lagrangian subproblem, which requires an extra global summation per outer iteration.

2. Eckstein and Silva's criterion [54]: At every dual iteration, the primal alternating minimisation (Algorithm 2) stops at inner iteration k if a vector v^k is found in the normal cone $\mathcal{N}_{\mathcal{Z}}(z^k)$, which satisfies

$$\frac{2}{\varrho} |\langle w - z^k, \nabla_z L_\varrho(z^k, \mu) + v^k \rangle| + \|\nabla_z L_\varrho(z^k, \mu) + v^k\|_2^2 \leq \sigma \|C(z^k)\|_2^2 \quad , \quad (2.41)$$

with $\sigma \in [0, 1)$ and w is an auxiliary vector updated at every dual iteration l as follows

$$w^l = w^{l-1} - \varrho^l (v^l + \nabla_z L_\varrho(z^l, \mu)) \quad .$$

Convergence to the dual loop is guaranteed in the convex case [54]. In the nonconvex case, theoretical guarantees have not been published, although good performance has been reported [54]. In particular, Eckstein and Silva's stopping criterion shows superior performance to the LANCELOT criterion in terms of gradient evaluations [54]. With respect to distributed

computations, Eckstein and Silva's stopping condition requires a global summation at every iteration of Algorithm 2. The update of the vector w can be distributed.

3. Heuristic criterion: The two stopping conditions described above involve global summations, which may be costly on a distributed architecture. Therefore, one may also stop the inner loop of Algorithm 4 after a fixed number iterations. Theoretical convergence guarantees of the outer loop are obviously lost when doing this. Besides, this strategy requires synchronisation between the computing units. One could also abort primal iterations when the innovation $z^k - z^{k-1}$ becomes too small, which can be checked in a distributed manner. Similarly, convergence guarantees are also lost.

2.1.4 Numerical experiments

We report numerical experiments to illustrate the theoretical results of the previous paragraphs. We consider nonconvex quadratic programs subject to linear coupling constraints and separable nonconvex constraints

$$\begin{aligned}
 & \underset{z_0, \dots, z_N}{\text{minimise}} \quad \sum_{i=0}^N \langle z_i, H_i z_i \rangle + \langle g_i, z_i \rangle & (2.42) \\
 & \text{s. t. } \quad A_i z_i + B_i z_{i+1} = b_i, \quad i \in \{0, \dots, N-1\} \\
 & \quad \quad \quad \|z_i\|_2 = r_i, \quad i \in \{0, \dots, N\} \quad ,
 \end{aligned}$$

where the matrices $H_i \in \mathbb{R}^{d \times d}$ are symmetric, $r_i > 0$, $z_i \in \mathbb{R}^d$, $A_i, B_i \in \mathbb{R}^{m \times d}$, $g_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}^m$ with $N \geq 1$, $d \geq 1$ and $m \geq 1$. As mentioned earlier in this chapter, the euclidean projection onto the separable nonconvex constraint sets in problem (2.42) can be evaluated in closed form, more precisely if $z \neq 0$,

$$P_{S(0,r)}(z) = r \frac{z}{\|z\|_2} \quad , \quad (2.43)$$

where $S(0, r)$ denote the sphere of radius r centred at the origin. If $z = 0$, the projection operator is multi-valued. Following Algorithm 2, one can take any point on the sphere $S(0, r)$. The problem data is randomly generated. In order to prevent too ill-conditioned problems, on which a first-order method such as Algorithm 4 can show very poor performance, the eigenvalues of matrices H_i , A_i and B_i are kept within the interval $[-1, 1]$.

After introducing Lagrange multipliers μ_i and a penalty ρ , the augmented Lagrangian subprob-

lem resulting from the relaxation of the linear constraints in (2.42) is written as

$$\begin{aligned} \underset{z_0, \dots, z_N}{\text{minimise}} \quad & \langle z_N, H_N z_N \rangle + \langle g_N, z_N \rangle + \sum_{i=0}^{N-1} \langle z_i, H_i z_i \rangle + \langle g_i, z_i \rangle \\ & + \left\langle \left(\mu_i + \frac{\rho}{2} (A_i z_i + B_i z_{i+1} - b_i) \right), (A_i z_i + B_i z_{i+1} - b_i) \right\rangle \\ \text{s. t.} \quad & \|z_i\|_2 = r_i, \quad i \in \{0, \dots, N\} \quad . \end{aligned} \tag{2.44}$$

Importantly, in the case of NLP (2.44), an alternating minimisation procedure such as the inner loop of Algorithm 4 can be parallelised. At every iteration, the sub-variables z_i with i even, are held constant and the sub-variables with odd indices are updated in parallel. Then, the sub-variables with an odd index are fixed to their current value and the sub-variables with even indices are updated in parallel. This fact stands out when looking at the dependency graph, as defined in [17], of the augmented Lagrangian subproblem (2.44) in Figure 2.1. To perform the update of variable z_i , one only needs to know variables z_{i-1} and z_{i+1} . In conclusion, a significant level of concurrency



Figure 2.1: Dependency graph of NLP (2.44).

is obtained when applying Algorithm 4 to NLP (2.42), as its inner loop then consists of two cyclic groups of parallel gradient projections. An important aspect of Algorithm 4 is that a finite number of primal iterations is often enough to guarantee convergence of the outer loop. This number can be obtained via the criticality-based stopping criterion or Eckstein’s stopping criterion, which yield theoretical convergence guarantees. One can also fix a priori the maximum number of primal alternating minimisations and analyse its effect on the convergence of the dual iterates. The final target is to have the smallest amount of primal iterations, resulting in an ADMM-like algorithm. Of course, one should keep in mind that theoretical guarantees of convergence are lost in this case. The results of this analysis are presented in Figures 2.2 and 2.3 below, and are obtained by running our coordination-decomposition scheme on the same random problem. The proposed algorithm has been implemented in MATLAB.

From the numerical results of Figure 2.2, a larger number of primal iterations results in faster convergence of the dual iterates. This is not difficult to understand as better satisfaction of the criticality condition on the augmented Lagrangian subproblem is obtained. However, the convergence

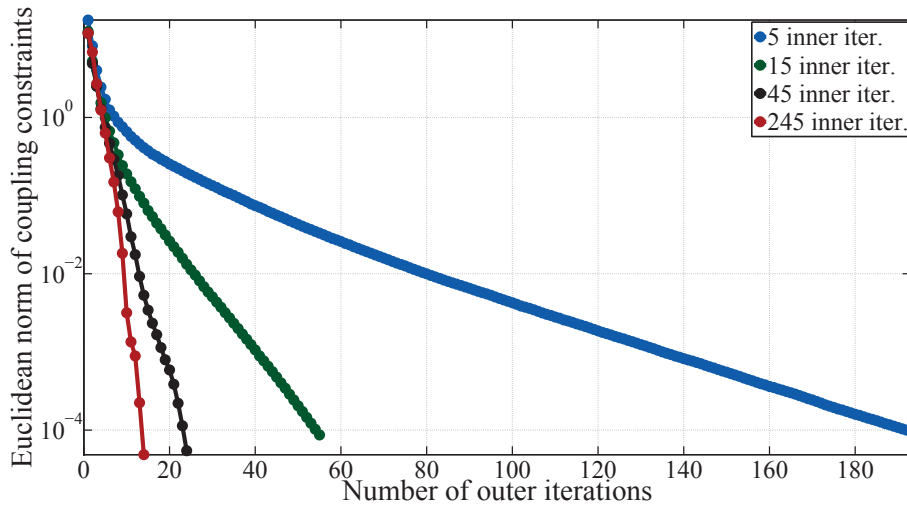


Figure 2.2: Euclidean norm of linear coupling constraints along outer iterations of Algorithm 1 for varying numbers of inner alternating minimisations (Algorithm 2), a multiplicative coefficient $\beta_\varrho = 2$ and an initial penalty $\varrho^{(0)} = 5$.

analysis of paragraphs 2.1.2 and 2.1.1 does not provide a convergence rate of the dual variables that depends on the number of primal iterations. Such a result is provided in Chapter 3 in a parametric context. From Figures 2.2 and 2.3, it appears that the rate of increase β_ϱ plays an important role on the convergence speed of the outer loop. A larger β_ϱ results in slower convergence. At this point, this is just an empirical observation, but the analysis of Chapter 3 gives insights into this phenomenon. Figure 2.4 shows that for a small rate of increase β_ϱ , the smallest total number of projected gradient steps is not necessarily obtained for the smallest possible number of primal iterations per dual iteration. As the total number of gradient projections is proportional to the computational time of Algorithm 4, one can conclude that the overall performance of Algorithm 4 is very sensitive to the choice of parameters, which are the number of primal iterations per dual step and the rate of increase β_ϱ . More precisely, the curves in Figure 2.5 show that the best performance is obtained with 25 alternating minimizations per dual step and $\beta_\varrho = 2$. Again, Chapter 3 will help clarifying this observation. One can reasonably expect that a larger amount of primal iterations per augmented Lagrangian subproblem results in a smaller overall number of dual steps. An important question is how fast the total number of outer iterations decreases. Figure 2.5 demonstrates that this decrease can actually be quick. For $\beta_\varrho = 2$, the number of outer iterations drops from around 200 for 5 alternating gradient projections per augmented Lagrangian subproblem to less than 15 for 50 inner iterations. Then, the number of outer iterations remains almost constant as the number of primal iterations per dual step increases. In conclusion, from a certain point, increasing

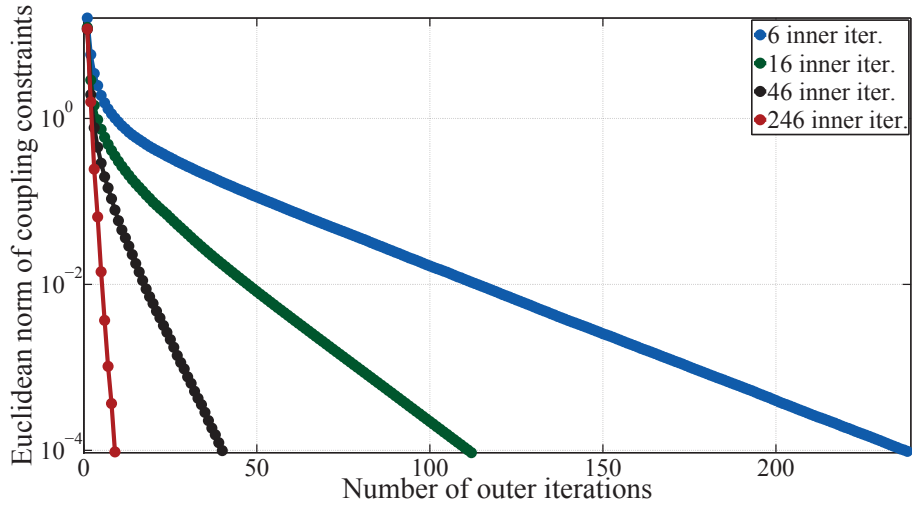


Figure 2.3: Euclidean norm of linear coupling constraints along outer iterations of Algorithm 1 for different maximum number of inner alternating minimisations (Algorithm 2), a multiplicative coefficient $\beta_\varrho = 15$ and an initial penalty $\varrho^{(0)} = 5$.

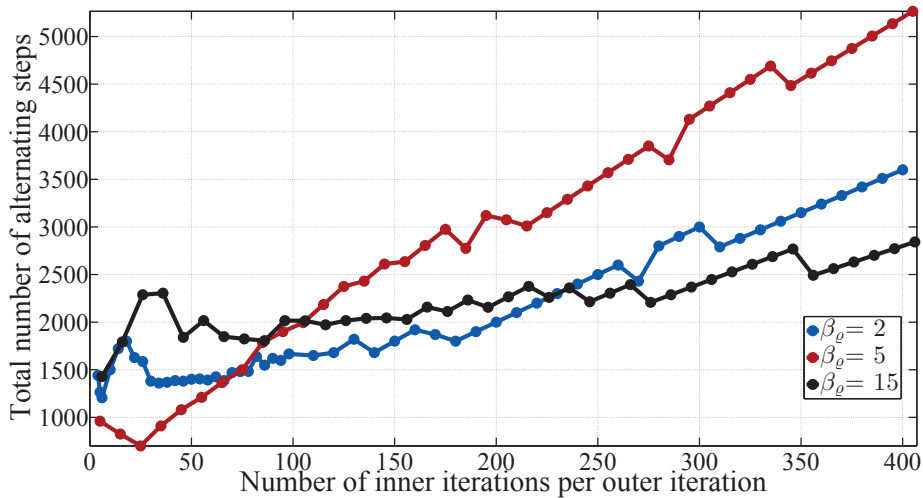


Figure 2.4: Total number of alternating projected gradient steps for reaching a feasibility of 10^{-4} in the linear coupling constraints along maximum number of primal iterations (Algorithm 2) per dual iteration (Algorithm 1). The initial penalty is $\varrho^{(0)} = 5$ and $\beta_\varrho \in \{2, 5, 15\}$.

the number of primal iterations per dual step does not help to improve convergence of Algorithm 4.

Next, we test our coordination-decomposition strategy on randomly generated nonconvex quadratic programs of the form (2.42) with varying number of elements N of fixed dimension d and fixed

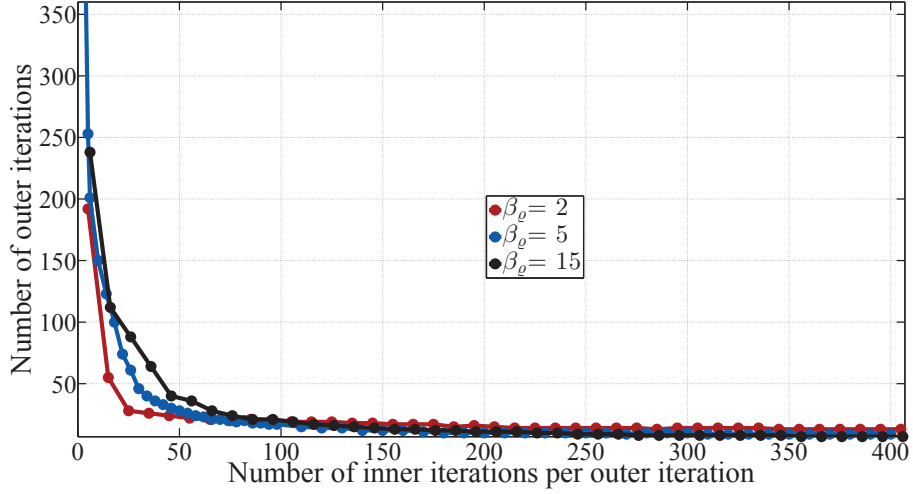


Figure 2.5: Number of dual iterations in Algorithm 4 to reach a feasibility 10^{-4} in the linear coupling constraints along maximum number of primal iterations per dual iteration. The initial penalty is $\varrho^{(0)} = 5$ and $\beta_\varrho \in \{2, 5, 15\}$.

number of constraints. We first start with $d = 10$, $m = 5$ and $N \in \{100, 200, 300, 400, 500, 600\}$. The output of Algorithm 4 is compared with IPOPT in terms of objective value and feasibility of the coupling constraints. As the problems we consider are medium to large-scale nonlinear programs, it is critical to stop the alternating minimisations at an early stage when the dual iterate is a bad estimate of an optimal Lagrange multiplier. Thus, we resort to Eckstein and Silva's relative error criterion [54] and fix the maximum number of inner steps per augmented Lagrangian subproblem to 100. An important point here is that although the constraint set of the augmented Lagrangian subproblem is nonconvex, it is easy to generate a vector in its normal cone. More precisely, the constraint set is

$$\Omega := S(0, r_1) \times \dots \times S(0, r_N) \quad . \quad (2.45)$$

Given $z \in \Omega$, the normal cone to Ω at z is

$$\mathcal{N}_\Omega(z) := \mathcal{N}_{S(0, r_1)}(z_1) \times \dots \times \mathcal{N}_{S(0, r_N)}(z_N) \quad , \quad (2.46)$$

where for $i \in \{1, \dots, N\}$,

$$\mathcal{N}_{S(0, r_i)}(z_i) = \{\alpha z_i, \alpha > 0\} \quad . \quad (2.47)$$

The right-hand side of the relative error criterion (2.41) involves the term

$$\|v^k + \nabla_z L_\varrho(z^k, \mu^k)\|_2^2, \quad (2.48)$$

where v^k lies in the normal cone $\mathcal{N}_\Omega(z^k)$. Thus, in order to make the relative error criterion (2.41) efficient in practice, one needs to choose v^k such that the summand (2.48) is minimum. However, by (2.46), one has

$$\|v^k + \nabla_z L_\varrho(z^k, \mu^k)\|_2^2 = \sum_{i=1}^N \|v_i^k + \nabla_{z_i} L_\varrho(z^k, \mu^k)\|_2^2.$$

Hence, the elements $v_i^k \in \mathcal{N}_{S(0,r_i)}(z_i^k)$ should be such that each term in the sum above is minimum. For $i \in \{1, \dots, N\}$, the vector $v_i^k \in \mathcal{N}_{S(0,r_i)}(z_i^k)$ minimising the associated term above is given by the solution of the one-dimensional program

$$\underset{\alpha_i > 0}{\text{minimise}} \quad \|\alpha_i z_i^k + \nabla_{z_i} L_\varrho(z^k, \mu^k)\|_2^2,$$

which is

$$\alpha_i^* = -\frac{\langle \nabla_{z_i} L_\varrho(z^k, \mu^k), z_i^k \rangle}{\|z_i^k\|_2^2}. \quad (2.49)$$

In conclusion, in the case of NLP (2.44), the relative error criterion (2.41) can be written

$$\begin{aligned} & \frac{2}{\varrho} \left| \sum_{i=1}^N \left\langle w_i - z_i^k, \nabla_{z_i} L_\varrho(z^k, \mu) - \frac{\langle \nabla_{z_i} L_\varrho(z^k, \mu), z_i^k \rangle}{\|z_i^k\|_2^2} z_i^k \right\rangle \right| \\ & + \sum_{i=1}^N \left\| \nabla_{z_i} L_\varrho(z^k, \mu) - \frac{\langle \nabla_{z_i} L_\varrho(z^k, \mu), z_i^k \rangle}{\|z_i^k\|_2^2} z_i^k \right\|_2^2 \leq \sigma \|C(z^k)\|_2^2. \end{aligned} \quad (2.50)$$

For the numerical experiments reported here, we chose $\sigma = 0.99$. It is thus easy to verify if the above inequality is satisfied at every inner iteration of Algorithm 4. One should keep in mind that theoretical convergence guarantees of Algorithm 4 are lost when applying this criterion, as NLP (2.44) is nonconvex. Nevertheless, it was observed that criterion (2.50) is able to significantly reduce the number of alternating minimizations in the first outer iterations of Algorithm 4 without compromising convergence, as shown in Fig. 2.6. In order to assess the performance of Algorithm 4, we generate large-scale nonconvex QPs, either by increasing the number of subvariables N or increasing their dimension d . Results are presented in Table 2.1. We compare against the nonlinear solver IPOPT [152] with the linear solver MA27. The stopping tolerance in IPOPT was

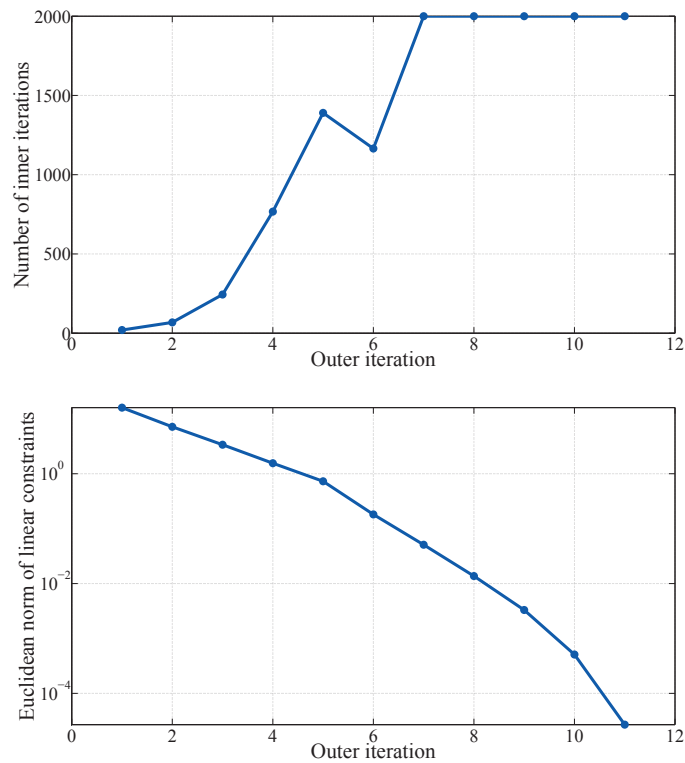


Figure 2.6

set to 10^{-7} . It appears that the number of inner iterations can be kept constant at a relatively small number (100 in our case) without hampering convergence of the outer loop. As the problem dimension increases, the number of outer iterations required to satisfy a specified feasibility level increases. One can also observe that the number of iterations taken by IPOPT varies significantly from one problem instance to another. The total number of iterations in IPOPT is smaller than the total number of alternating minimisations in Algorithm 4. However, each of these iterations is far more costly and harder to decompose, as it consists in solving a linear system via a direct method. From Table 2.1, one can observe that IPOPT provides a lower objective value and tighter satisfaction of the equality constraints upon convergence. Thus, our decomposition-coordination algorithm is advisable when a moderate level of optimality is required.

d	m	N	$\varrho^{(0)}$	β_q	Max. # prim. iter.	# du. iter.	Coupl. feas.	Obj.	IPOPT coupl. feas.	IPOPT obj.	IPOPT # iter.
10	5	100	5	2	100	20	$4.79 \cdot 10^{-5}$	$-5.70 \cdot 10^3$	$7.83 \cdot 10^{-15}$	$-5.82 \cdot 10^3$	172
10	5	200	5	2	100	22	$5.87 \cdot 10^{-5}$	$-1.14 \cdot 10^4$	$8.12 \cdot 10^{-15}$	$-1.16 \cdot 10^4$	194
10	5	300	5	2	100	22	$6.46 \cdot 10^{-5}$	$-1.74 \cdot 10^4$	$4.85 \cdot 10^{-11}$	$-1.73 \cdot 10^4$	300
10	5	400	5	2	100	32	$8.43 \cdot 10^{-5}$	$-8.98 \cdot 10^4$	$2.95 \cdot 10^{-14}$	$-9.49 \cdot 10^4$	461
10	5	500	5	5	100	17	$6.32 \cdot 10^{-5}$	$-1.07 \cdot 10^5$	$3.39 \cdot 10^{-14}$	$-1.11 \cdot 10^5$	807
10	5	600	5	5	100	45	$9.62 \cdot 10^{-5}$	$-1.35 \cdot 10^5$	$3.52 \cdot 10^{-14}$	$-1.43 \cdot 10^5$	322
20	5	100	5	5	100	9	$2.71 \cdot 10^{-5}$	$-3.15 \cdot 10^4$	$1.28 \cdot 10^{-14}$	$-3.23 \cdot 10^4$	124
20	5	200	5	5	100	20	$8.36 \cdot 10^{-5}$	$-6.36 \cdot 10^4$	$1.83 \cdot 10^{-14}$	$-6.56 \cdot 10^4$	335
20	5	300	5	5	100	20	$6.70 \cdot 10^{-5}$	$-9.56 \cdot 10^4$	$2.25 \cdot 10^{-14}$	$-9.80 \cdot 10^4$	323
20	5	400	5	5	100	12	$3.52 \cdot 10^{-5}$	$-1.27 \cdot 10^5$	$2.56 \cdot 10^{-14}$	$-1.31 \cdot 10^5$	372
20	15	100	5	5	100	26	$7.17 \cdot 10^{-5}$	$-1.46 \cdot 10^4$	$2.63 \cdot 10^{-14}$	$-1.73 \cdot 10^4$	195
20	15	200	5	5	100	66	$9.83 \cdot 10^{-5}$	$-2.95 \cdot 10^4$	$3.68 \cdot 10^{-14}$	$-3.49 \cdot 10^4$	345
20	15	300	5	5	100	60	$8.69 \cdot 10^{-5}$	$-4.60 \cdot 10^4$	$4.73 \cdot 10^{-14}$	$-5.32 \cdot 10^4$	381

Table 2.1: Results of Algorithm 4 applied to problems of the form (2.44) for varying N . The outer loop of Algorithm 4 is aborted when the 2-norm of the linear coupling constraints reaches 10^{-4} .

2.2 Trust Region with Alternating Projections (TRAP)

In this section, we introduce a novel decomposition technique that can be applied to the augmented Lagrangian subproblems. Like any first order method, the performance of the alternating direction method introduced in the previous section is generally sensitive to the problem conditioning. However, as the outer loop converges, the augmented Lagrangian subproblems are becoming increasingly ill-conditioned and a tighter satisfaction of the optimality conditions is required to guarantee convergence. Thus, a method with robust and fast local convergence properties is required as an inner solver. The alternating minimisations strategy described above does not provide such guarantees. In order to ensure fast local convergence, a Newton model is generally a first step. Yet, the distributed Newton subproblem should also be solved by means of a decomposition method. In principle, if the Newton subproblem is convex, Lagrangian dual decomposition strategies can be used [23]. A major drawback of this approach is that the local convergence rate of splitting techniques is at best linear [81, 142]. In practice, although the speed of convergence can be enhanced by means of well-chosen preconditioners [70] or smoothing techniques [149], arbitrarily bad performance can be obtained in terms of accuracy and speed, especially when the problem data changes at every iteration rendering preconditioning hard. As a result, decomposition strategies from the convex world should be considered as bad candidates for solving distributed Newton subproblems arising in an SCP scheme. Dual Newton strategies [103] are probably more suitable for this, but there is a lack of numerical experience regarding their performance when used in SQP schemes, as well as preconditioning. A better choice for a distributed inner algorithm seems to be the conjugate gradient [139], as it is based on sparse structured matrix-vector products, which can be easily implemented on distributed platforms, and is guaranteed to converge after a finite number of iterations. Moreover, with safeguarding mechanisms, it can be applied to solve indefinite linear systems in a trust region framework [145].

Another important point is that the Newton algorithm may fail to converge from remote starting point. Therefore, a globalisation mechanism, such as trust region or line search, is required. Line search requires repeated evaluations of a merit function along a descent direction, which can be cumbersome in a distributed context, as it requires multiple synchronisations per iteration. On the contrary, a trust region procedure necessitates one single evaluation of a merit function per iteration, which is likely to be more efficient in a distributed context where communication is expensive. These considerations lead us to consider trust region methods [145, 119] as a good starter for a distributed nonconvex solver.

Next, we present and analyse a novel trust region strategy applicable to linearly constrained nonlinear programs. Compared to related existing methods, the main novelty is in the Cauchy point computation. Instead of computing a centralised gradient step via projected search, as done

in a standard trust region approach, we resort to the alternating minimisation method described in the previous section to compute a Cauchy point.

2.2.1 A trust region algorithm with distributed activity detection

2.2.1.1 Algorithm formulation

The problem we consider is that of minimising a partially separable objective function subject to separable convex constraints.

$$\begin{aligned} & \underset{w}{\text{minimise}} \quad L(w_1, \dots, w_N) & (2.51) \\ & \text{s. t. } w_i \in \mathcal{W}_i, \forall i \in \{1, \dots, N\} \quad , \end{aligned}$$

where $w := (w_1^\top, \dots, w_N^\top)^\top \in \mathbb{R}^n$, with $n = \sum_{i=1}^N n_i$, and $\mathcal{W} := \mathcal{W}_1 \times \dots \times \mathcal{W}_N$, where the sets $\mathcal{W}_i \subset \mathbb{R}^{n_i}$ are closed and convex. The following Assumption is standard in distributed computations [17].

Assumption 2.9 (Colouring scheme). *There exists a colouring of the dependency graph of the Gauss-Seidel sweep on the objective function L with $K \ll N$ colours and no positive cycle with all variables w_1, \dots, w_N in the same colour.*

Consequently, by Lemma 1.5, the sub-variables w_1, \dots, w_N can be re-ordered and grouped together in such a way that a Gauss-Seidel minimisation sweep on the function L can be performed in parallel within $K \ll N$ groups, which are updated sequentially. In the sequel, the re-ordered variable is denoted by $x = (x_1^\top, \dots, x_K^\top)^\top$. The set \mathcal{W} is transformed accordingly into $\Omega = \Omega_1 \times \dots \times \Omega_K$. It is worth noting that each set Ω_k with $k \in \{1, \dots, K\}$ can then be decomposed further into sets \mathcal{W}_i with $i \in \{1, \dots, N\}$. Hence, NLP (2.51) is equivalent to

$$\begin{aligned} & \underset{x}{\text{minimise}} \quad L(x_1, \dots, x_K) \\ & \text{s. t. } x_k \in \Omega_k, \forall k \in \{1, \dots, K\} \quad . \end{aligned}$$

Remark 2.5. *Such a partially separable structure in the objective (Assumption 2.9) is encountered very often in practice, for instance when relaxing network coupling constraints via an augmented Lagrangian penalty. Thus, by relaxing the nonlinear coupling constraint $C(w_1, \dots, w_N) = 0$ and*

the local equality constraints $g_i(w_i) = 0$ of

$$\begin{aligned} & \underset{w_1, \dots, w_N}{\text{minimise}} \sum_{i=1}^N f_i(w_i) \\ & \text{s. t. } C(w_1, \dots, w_N) = 0 \\ & \quad g_i(w_i) = 0, \quad i \in \{1, \dots, N\} \\ & \quad w_i \in \mathcal{W}_i, \quad i \in \{1, \dots, N\} \quad , \end{aligned}$$

in a differentiable penalty function, one obtains an NLP of the form (2.51). In NLPs resulting from the direct transcription of optimal control problems, the objective is generally separable and the constraints are stage-wise with a coupling between the variables at a given time instant with the variables of the next time instant. In this particular case, the number of groups is $K = 2$. In Section 2.2.3, we illustrate this property by means of examples arising from various formulations of the Optimal Power Flow (OPF) problem. The number of colours K represents the level of parallelism that can be achieved in a Gauss-Seidel method for solving (2.51). Thus, in the case of a discretised OCP, an alternating projected gradient sweep can be applied in two steps during which all updates are parallel.

For the sake of exposition, in order to make the distributed nature of our algorithm apparent, we assume that every sub-variable w_i , with $i \in \{1, \dots, N\}$, is associated with a computing node insofar as possible. One should note that it may be difficult to make the association of the computing nodes respect the coupling topology. Two nodes are called *neighbours* if they are coupled in the objective L . Our goal is to find a first-order critical point of NLP (2.51) via an iterative procedure for which we are given an initial feasible point $x^0 \in \Omega$. The iterative method described next aims at computing every iterate in a distributed fashion, which requires communications between neighbouring nodes and leads to a significant level of concurrency.

Assumption 2.10. *The objective function L is bounded below on $\{x \in \Omega : L(x) \leq L(x^0)\}$.*

The algorithm formulation can be done for any convex set Ω , but some features are more suitable for linear inequality constraints.

Assumption 2.11 (Polyhedral constraints). *For all $k \in \{1, \dots, K\}$, the set Ω_k is a non-empty polyhedron, such that*

$$\Omega_k := \{x \in \mathbb{R}^{n_k} : \langle \omega_{k,i}, x \rangle \leq h_{k,i}, \quad i \in \{1, \dots, m_k\}\} \quad ,$$

with $\omega_{k,i} \in \mathbb{R}^{n_k}$, $h_{k,i} \in \mathbb{R}$ for all $i \in \{1, \dots, m_k\}$ and $n_k, m_k \geq 1$.

Assumption 2.12. *The objective function L is continuously differentiable in an open set containing Ω . Its gradient ∇L is uniformly continuous.*

It is well-known [36] that for problem (2.51), x^* being a critical point is equivalent to

$$P_{\Omega}(x^* - \nabla L(x^*)) = x^* . \quad (2.52)$$

Algorithm 5 below is designed to compute a critical point x^* of the function $L + \iota_{\Omega}$. It is essentially a two-phase approach, in which an active-set is first computed and then, a quadratic model is minimised approximately on the current active face. Standard two-phase methods compute the active-set by means of a centralised projected search, updating all variables centrally. More precisely, a model of the objective is minimised along the projected objective gradient, which yields the Cauchy point. The model decrease provided by the Cauchy point is then enhanced in a refinement stage. Similarly to a two-phase method, in order to globalise convergence, Algorithm 5 uses the standard trust region mechanism. At every iteration, a model m of the objective function L is constructed around the current iterate x as follows

$$m(x') := L(x) + \langle \nabla L(x), x' - x \rangle + \frac{1}{2} \langle x' - x, B(x)(x' - x) \rangle , \quad (2.53)$$

where $x' \in \mathbb{R}^n$ and $B(x)$ is a symmetric matrix.

Assumption 2.13 (Uniform bound on model hessian). *There exists $\hat{B} > 0$ such that*

$$\|B(x)\|_2 \leq \hat{B} ,$$

for all $x \in \Omega$.

The following Assumption is necessary to ensure distributed computations in Algorithm 5. It is specific to Algorithm 5 and does not appear in the standard trust region methods [26].

Assumption 2.14 (Structured model hessian). *For all $i, j \in \{1, \dots, n\}$, $B_{i,j}(x) = 0$ for all $x \in \Omega$ if and only if the gradient of the objective function L with respect to the group of variables indexed by i does not depend on group j for all $x \in \Omega$.*

Remark 2.6. *It is worth noting that the partial separability structure of the objective function L is transferred to the sparsity pattern of the model hessian B . Hence, a Gauss-Seidel sweep on the model function m can also be carried out in K parallel steps.*

The main characteristic of TRAP is the activity detection phase, which differs from the projected search in standard trust region methods [26]. At every iteration, TRAP updates the current

Algorithm 5 Trust Region Algorithm with Alternating Projections (TRAP)

```

1: Constants: Initial trust region radius  $\Delta$ , update parameters  $\sigma_1, \sigma_2$  and  $\sigma_3$  such that  $0 < \sigma_1 < \sigma_2 < 1 < \sigma_3$ , test ratios  $\eta_1$  and  $\eta_2$  such that  $0 < \eta_1 < \eta_2 < 1$ , coefficients  $\gamma_1 \in ]0, 1[$  and  $\gamma_2 > 0$ , termination tolerance  $\epsilon$ .
2: Input: Initial guess  $x$ , projection operators  $\{P_{\Omega_k}\}_{k=1}^K$ , objective function  $L$ , objective gradient  $\nabla L$ .
3: while  $\|P_{\Omega}(x - \nabla L(x)) - x\|_2 > \epsilon$  do
4:   Distributed activity detection (alternating gradient projections):
5:   for  $k = 1 \dots, K$  do
6:      $z_k \leftarrow P_{\Omega_k}(x_k - \alpha_k \nabla_k m(z_{[1, k-1]}, x_k, x_{[k+1, K]}))$ , ▷ In parallel in group  $k$ 
7:     where  $\alpha_k$  is computed according to requirements (2.55), (2.56) and (2.57).
8:   end for
9:   Distributed refinement (Algorithm 6):
10:  Find  $y \in \Omega$  such that
11:     $m(x) - m(y) \geq \gamma_1 (m(x) - m(z))$ 
12:     $\|y - x\|_{\infty} \leq \gamma_2 \Delta$ 
13:     $\mathcal{A}_{\Omega_k}(z_k) \subset \mathcal{A}_{\Omega_k}(y_k)$  for all  $k \in \{1, \dots, K\}$ .
14:  Trust region update:
15:   $\rho \leftarrow L(x) - L(y) / m(x) - m(y)$ 
16:  if  $\rho < \eta_1$  then ▷ Not successful
17:    (Do not update  $x$ )
18:    Take  $\Delta$  within  $[\sigma_1 \Delta, \sigma_2 \Delta]$ 
19:  else if  $\rho \in [\eta_1, \eta_2]$  then ▷ Successful
20:     $x \leftarrow y$ 
21:    Take  $\Delta$  within  $[\sigma_1 \Delta, \sigma_3 \Delta]$ 
22:    Update objective gradient  $\nabla L(x)$  and model hessian  $B(x)$ .
23:  else ▷ Very successful
24:     $x \leftarrow y$ 
25:    Take  $\Delta$  within  $[\Delta, \sigma_3 \Delta]$ 
26:    Update objective gradient  $\nabla L(x)$  and model hessian  $B(x)$ 
27:  end if
28: end while
    
```

active-set by computing iterates z_1, \dots, z_K (Lines 4 to 8). This is the main novelty of TRAP, compared to existing two-phase techniques, and allows for different step-sizes $\alpha_1, \dots, \alpha_K$ per block of variables, which is relevant in a distributed framework, as the current active-set can be split among nodes and does not need to be computed centrally. In the trust region literature, the point

$$z := (z_1^{\top}, \dots, z_K^{\top})^{\top}, \quad (2.54)$$

is often referred to as the *Cauchy point*. We keep this terminology in the remainder of the chapter. It is clear from its formulation that TRAP allows one to compute Cauchy points via independent

projected searches on every node. Once the Cauchy points z_1, \dots, z_K have been computed, they are used in the refinement step to compute a new iterate y that satisfies the requirements shown from Lines 10 to 13. The last step consists in checking if the model decrease $m(y) - m(x)$ is sufficiently close to the variation in the objective L (Lines 16 to 27). In this case, the iterate is updated and the trust region radius Δ increased, otherwise the radius is shrunk and the iterate frozen. This operation requires a global exchange of information between nodes.

In the remainder, the objective gradient $\nabla L(x)$ is denoted by $g(x)$. The model function m is an approximation of the objective function L around the current iterate x . The quality of the approximation is controlled by the trust region, defined as the box

$$\mathbb{B}(x - \Delta, x + \Delta) \quad ,$$

where Δ is the trust region radius.

In the rest of the chapter, we denote the Cauchy points by z_k or $z_k(\alpha_k)$ without distinction, where α_k are appropriately chosen step-sizes. More precisely, following Section 3 in [26], in TRAP, the block-coordinate step-sizes α_k are chosen so that for all $k \in \{1, \dots, K\}$, the Cauchy points z_k satisfy

$$\begin{cases} m(z_{[1,k-1]}, z_k, x_{[k+1,K]}) \leq m(z_{[1,k-1]}, x_k, x_{[k+1,K]}) \\ \quad + \nu_0 \langle \nabla_k m(z_{[1,k-1]}, x_k, x_{[k+1,K]}) , z_k - x_k \rangle \quad , \\ \|z_k - x_k\|_\infty \leq \nu_2 \Delta \quad , \end{cases} \quad (2.55)$$

with $\nu_0 \in]0, 1[$ and $\nu_2 > 0$, where $z_{[1,k-1]}$ stands for $(z_1^\top, \dots, z_{k-1}^\top)^\top$, along with the condition that there exists positive scalars $\nu_1 < \nu_2, \nu_3, \nu_4$ and ν_5 for all $k \in \{1, \dots, K\}$,

$$\alpha_k \in [\nu_4, \nu_5] \quad \text{or} \quad \alpha_k \in [\nu_3 \tilde{\alpha}_k, \nu_5] \quad (2.56)$$

where the step-sizes $\tilde{\alpha}_k$ are such that one of the following conditions hold for every $k \in \{1, \dots, K\}$,

$$\begin{aligned} m(z_{[1,k-1]}, z_k(\tilde{\alpha}_k), x_{[k+1,K]}) &> m(z_{[1,k-1]}, x_k, x_{[k+1,K]}) \\ &+ \nu_0 \langle \nabla_k m(z_{[1,k-1]}, x_k, x_{[k+1,K]}) , z_k(\tilde{\alpha}_k) - x_k \rangle \quad , \end{aligned} \quad (2.57)$$

or

$$\|z_k(\tilde{\alpha}_k) - x_k\|_\infty \geq \nu_1 \Delta \quad , \quad (2.58)$$

Conditions (2.55) ensure that the step-sizes α_k are sufficiently small to enforce a sufficient decrease

coordinate-wise, as well as containment within a scaled trust region. Conditions (2.56), (2.57) and (2.58) guarantee that the step-sizes α_k do not become arbitrarily small. All conditions (2.55), (2.56) and (2.57) can be tested in parallel in each of the K groups of variables. In the next two paragraphs, the choice of step-sizes α_k ensuring the sufficient decrease is clarified, as well as the distributed refinement step. In paragraph 2.2.2 next, the convergence properties of TRAP are analysed. Numerical examples are presented in paragraph 2.2.3.

2.2.1.2 Step-sizes computation in the activity detection phase

At a given iteration of TRAP, the step-sizes α_k are computed by backtracking to ensure a sufficient decrease at every block of variables and coordinate-wise containment in a scaled trust region as formalised by (2.55). It is worth noting that the coordinate-wise backtracking search can be run in parallel among the variables of group k , as they are decoupled from each other. As a result, there is one step-size per sub-variable w_i in group k . Yet, for simplicity, we write it as a single step-size α_k . The reasoning of paragraph 2.2.2 can be adapted accordingly. The following Lemma shows that a coordinate-wise step-size α_k can be computed that ensures conditions (2.55), (2.56), (2.57) and (2.58) on every block of coordinates $k \in \{1, \dots, K\}$.

Lemma 2.6. *Assume that Assumption 2.13 holds. For all $k \in \{1, \dots, K\}$, an iterate z_k satisfying conditions (2.55), (2.56), (2.57) and (2.58) can be found after a finite number of backtracking iterations.*

Proof. Let $k \in \{1, \dots, K\}$. We first show that for a sufficiently small α_k , conditions (2.55) are satisfied. By definition of the Cauchy point z_k ,

$$z_k = \operatorname{argmin}_{z \in \Omega_k} \langle \nabla_k m(z_{[1, k-1]}, x_k, x_{[k+1, K]}), z - x_k \rangle + \frac{1}{2\alpha_k} \|z - x_k\|_2^2 ,$$

which implies that

$$\langle \nabla_k m(z_{[1, k-1]}, x_k, x_{[k+1, K]}), z_k - x_k \rangle + \frac{1}{2\alpha_k} \|z_k - x_k\|_2^2 \leq 0 ,$$

Hence, as $\nu_0 \in]0, 1[$, it follows that

$$\begin{aligned} \langle \nabla_k m(z_{[1, k-1]}, x_k, x_{[k+1, K]}), z_k - x_k \rangle + \frac{1 - \nu_0}{2\alpha_k} \|z_k - x_k\|_2^2 \leq \\ \nu_0 \langle \nabla_k m(z_{[1, k-1]}, x_k, x_{[k+1, K]}), z_k - x_k \rangle . \end{aligned}$$

However, from the descent Lemma, which can be applied since the model gradient is Lipschitz

continuous by Assumption 2.13,

$$m(z_{[1,k-1]}, z_k, x_{[k+1,K]}) \leq m(z_{[1,k-1]}, x_k, x_{[k+1,K]}) + \langle \nabla_k m(z_{[1,k-1]}, x_k, x_{[k+1,K]}), z_k - x_k \rangle + \frac{\hat{B}}{2} \|z_k - x_k\|_2^2 .$$

By choosing

$$\alpha_k \leq \frac{1 - \nu_0}{\hat{B}} ,$$

condition (2.55) is satisfied after a finite number of backtracking iterations. Denoting by q_k the smallest integer such that requirement (2.55) is met, α_k can be written

$$\alpha_k = c^{q_k} \cdot \alpha^{(0)} ,$$

where $c \in]0, 1[$ and $\alpha^{(0)} > 0$. Then, condition (2.56) is satisfied with $\nu_4 = \alpha^{(0)}$ and $\nu_3 = c$. \square

Lemma 2.6 is very close to Theorem 4.2 in [115], but the argument regarding the existence of the step-sizes α_k is different.

2.2.1.3 Distributed computations in the refinement step

In Algorithm 5, the objective gradient $g(x)$ and model hessian $B(x)$ are updated after every successful iteration. This task requires exchanges of variables between neighbouring nodes, as the objective is partially separable (Ass. 2.9). Node i only needs to store the sub-part of the objective function L that combines its variable w_i and the variables associated to its neighbours. However, the refinement step (line 10 to 13 in Algorithm 5), in which one obtains a fraction of the model decrease yielded by the Cauchy points z_1, \dots, z_K , should also be computed in a distributed manner. As detailed next, this phase consists in solving the Newton problem on the subspace of free variables at the current iteration, which is defined as the set of free variables at the Cauchy points z_1, \dots, z_K . In order to achieve a reasonable level of efficiency in the trust region procedure, this step is generally performed via the Steihaug-Toint CG, or sCG [145]. The sCG algorithm is a CG procedure that is cut if a negative curvature direction is encountered or a problem bound is hit in the process. Another way of improving on the Cauchy point to obtain fast local convergence is the Dogleg strategy [119]. However, this technique requires the model hessian B to be positive definite [119]. This condition does not fit well with distributed computations, as positive definiteness is typically enforced by means of BFGS updates, which are known for not preserving the sparsity structure of the objective without non-trivial modifications and assumptions [155]. Com-

pared to direct methods, iterative methods such as the sCG procedure have clear advantages in a distributed framework, for they do not require assembling the hessian matrix on a central node. Furthermore, their convergence speed can be enhanced via block-diagonal preconditioning, which is suitable for distributed computations. In the sequel, we briefly show how a significant level of distributed operations can be obtained in the sCG procedure, mainly due to the sparsity structure of the model hessian that matches the partial separability structure of the objective function. More details on distributed implementations of the CG algorithm can be found in numerous research papers [151, 45, 56]. The sCG algorithm that is described next is a rearrangement of the standard sCG procedure following the idea of [45]. The two separate inner products that usually appear in the CG are grouped together at the same stage of the algorithm.

An important feature of the refinement step is the increase of the active set at every iteration. More precisely, in order to ensure finite detection of activity, the set of active constraints at the points y_1, \dots, y_K , obtained in the refinement phase, needs to contain the set of active constraints at the Cauchy points z_1, \dots, z_K , as formalised at line 13 of Algorithm 5. This requirement is very easy to fulfil when Ω is a bound constraint set, as it just requires enforcing the constraint

$$y_{k,i} = z_{k,i}, \quad i \in \{j \in \{1, \dots, n_k\} : z_{k,j} = \underline{x}_{k,j} \text{ or } \bar{x}_{k,j}\}$$

for all groups $k \in \{1, \dots, K\}$ in the trust region problem at the refinement step.

For the convergence analysis that follows in paragraph 2.2.2, the refinement step needs to be modified compared to existing trust region techniques. Instead of solving the standard refinement problem

$$\begin{aligned} & \underset{p}{\text{minimise}} \quad \langle g(x), p \rangle + \frac{1}{2} \langle p, B(x) p \rangle \\ & \text{s. t.} \quad \|p\|_\infty \leq \gamma_2 \Delta \\ & \quad \quad x + p \in \Omega \\ & \quad \quad \mathcal{A}_\Omega(z) \subseteq \mathcal{A}_\Omega(x + p) \quad , \end{aligned}$$

in which the variables corresponding to indices of active constraints at the Cauchy point z are fixed to zero, we solve a regularised version

$$\begin{aligned} & \underset{y \in \Omega}{\text{minimise}} \quad \langle g(x), y - x \rangle + \frac{1}{2} \langle y - x, B(x) (y - x) \rangle + \frac{\sigma}{2} \|y - z\|_2^2 \quad (2.59) \\ & \text{s. t.} \quad \|y - x\|_\infty \leq \gamma_2 \Delta \\ & \quad \quad \mathcal{A}_\Omega(z) \subseteq \mathcal{A}_\Omega(y) \quad , \end{aligned}$$

where $\sigma \in]\underline{\sigma}, \bar{\sigma}[$ with $\underline{\sigma} > 0$, and z is the Cauchy point yielded by the procedure described in the previous paragraph 2.2.1. The regularisation coefficient σ should not be chosen arbitrarily, as it may inhibit the fast local convergence properties of the Newton method. This point is made explicit in paragraph 2.2.2. The regularised trust region subproblem (2.59) can be equivalently written

$$\begin{aligned} & \underset{p}{\text{minimise}} \quad \langle g_\sigma(x), p \rangle + \frac{1}{2} \langle p, B_\sigma(x) p \rangle & (2.60) \\ & \text{s. t. } x + p \in \Omega \\ & \quad \|p\|_\infty \leq \gamma_2 \Delta \\ & \quad \mathcal{A}_\Omega(z) \subseteq \mathcal{A}_\Omega(x + p) \quad , \end{aligned}$$

with

$$g_\sigma(x) := g(x) - \sigma(z - x), \quad B_\sigma(x) := B(x) + \frac{\sigma}{2} I \quad . \quad (2.61)$$

As in standard trust region methods, we solve the refinement subproblem (2.60) by means of CG iterations, which can be distributed as a result of Assumption 2.14. In order to describe this stage in Algorithm 6, one needs to assume that Ω is a box constraint set. In the remainder, we denote by Z the matrix whose columns are an orthonormal basis of the subspace

$$V(z) := \{x \in \mathbb{R}^n : \langle \omega_{k,i}, x_k \rangle = 0, i \in \mathcal{A}_{\Omega_k}(z_k), k \in \{1, \dots, K\}\} \quad .$$

Remark 2.7. *It is worth noting that the requirement $m(x) - m(y) \geq \gamma_1 (m(x) - m(z))$, with $\gamma_1 < 1$, is satisfied after all iteration of Algorithm 6, as the initial guess is the Cauchy point z and the sCG iterations ensure monotonic decrease of the regularised model (Theorem 2.1 in [145]).*

Remark 2.8. *It is worth noting that the sparsity pattern of the reduced model hessian B_σ has the same structure as the sparsity pattern of the model hessian B , as the selection matrix Z has a block-diagonal structure. Moreover, the partial separability structure of the objective matches the sparsity patterns of both the hessian and the reduced hessian. For notational convenience, Algorithm 6 is written in terms of variables x_1, \dots, x_K , but it is effectively implementable in terms of variables w_1, \dots, w_N . The inner products (Lines 6 to 8) and updates (Lines 11 to 14, lines 20 to 22) can be computed in parallel at every node, as well as the structured matrix-vector product (Line 5).*

In Algorithm 6, the reduced model hessian \hat{B} can be evaluated when computing the product at line 5, which requires local exchanges of vectors between neighbouring nodes, since the sparsity

Algorithm 6 Distributed Safeguarded Conjugate Gradient (sCG)

```

1: Input: reduced model hessian  $\hat{B}_\sigma := Z^\top B_\sigma Z$ , reduced gradient  $\hat{g} := Z^\top g$ , initial guess  $\hat{z} := Z^\top z$ 
2: Parameters: stopping tolerance  $\hat{\epsilon} := \xi \|\hat{g}\|_2$  with  $\xi \in ]0, 1[$ 
3: Initialise  $\hat{x}, \hat{p}, \hat{v}, \hat{r}, \hat{t}$  and  $\hat{u}_{\text{prev}}$  via a standard sCG iteration using  $z, x, Z, B_\sigma, \hat{B}_\sigma$  and  $\hat{g}$ 
4: while  $\hat{u} > \hat{\epsilon}^2$  and  $\hat{t} > 0$  do
5:   Compute structured matrix-vector product  $\hat{s} \leftarrow \hat{B}_\sigma \hat{r}$  ▷ Local communications
6:   for  $k = 1 \dots K$  do ▷ In parallel among  $K$  groups
7:     Compute  $\langle \hat{r}_k, \hat{r}_k \rangle$  and  $\langle \hat{r}_k, \hat{s}_k \rangle$ 
8:   end for
9:    $\hat{u} \leftarrow \sum_{i=k}^K \langle \hat{r}_k, \hat{r}_k \rangle$ ,  $\hat{\delta} \leftarrow \sum_{k=1}^K \langle \hat{r}_k, \hat{s}_k \rangle$  ▷ Global summations
10:  Compute step-sizes  $\hat{\beta} \leftarrow \hat{u}/\hat{u}_{\text{prev}}$  and  $\hat{t} \leftarrow \hat{\delta} - \hat{\beta}^2 \hat{t}$ 
11:  for  $k = 1 \dots K$  do ▷ In parallel among  $K$  groups
12:    Update conjugate direction  $\hat{p}_k \leftarrow \hat{r}_k + \hat{\beta} \hat{p}_k$  and  $\hat{v}_k \leftarrow \hat{s}_k + \hat{\beta} \hat{v}_k$ 
13:    Compute smallest step-size  $a_k$  such that  $\hat{x}_k + a_k \hat{p}_k$  hits a bound  $\underline{x}_k, \bar{x}_k$  or the trust region boundary
14:  end for
15:  if  $\hat{t} \leq 0$  then ▷ Negative curvature check
16:    Compute step-size  $\hat{a} \leftarrow \min \{a_1, \dots, a_K\}$  to hit boundary of  $\mathbb{B}(x - \Delta, x + \Delta) \cap \Omega$ 
17:  else
18:    Compute standard CG step-size  $\hat{a} \leftarrow \hat{u}/\hat{t}$ 
19:  end if
20:  for  $k = 1 \dots K$  do ▷ In parallel among  $K$  groups
21:    Update iterate  $\hat{x}_k \leftarrow \hat{x}_k + \hat{a} \hat{p}_k$  and residual  $\hat{r}_k \leftarrow \hat{r}_k - \hat{a} \hat{v}_k$ 
22:  end for
23:   $\hat{u}_{\text{prev}} \leftarrow \hat{u}$ 
24: end while
25: Output:  $y \leftarrow z + Z(\hat{x} - \hat{z})$ 

```

pattern of \hat{B} represents the coupling structure in the objective L . From a distributed implementation perspective, the more costly parts of the refinement procedure 6 are at line 9 and line 16. These operations consist in summing up the inner products from all nodes and a minimum search over the step-sizes that ensure constraint satisfaction and containment in the trust region. They need to be performed on a central node that has access to all data from other nodes, or via a consensus algorithm. Therefore, lines 9 and 16 come with a communication cost, although the amount of transmitted data is very small (one scalar per node). In the end, one should notice that the information that is required to be known globally by all nodes $\{1, \dots, N\}$ is fairly limited at every iteration of TRAP. It only consists of the trust region radius Δ and the step-sizes \hat{a} and $\hat{\beta}$ in the refinement step 6. Finally, at every iteration, all nodes need to be informed of the success or failure of the iteration so as to update or freeze their local variables. This is the result of the trust region test,

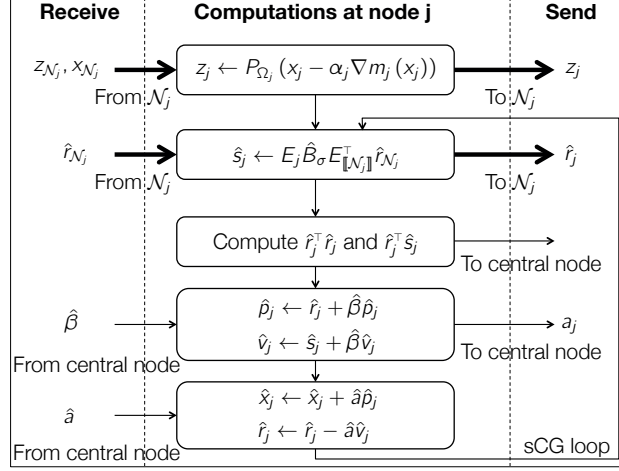


Figure 2.7: Workflow at node j in terms of local computations, communications with the set of neighbours \mathcal{N}_j and a central node. Note that we use the index j for a node, and not k , which corresponds to a group of nodes, in which computations are performed in parallel. Thus, the nodes in the set \mathcal{N}_j are not in the same group as node j . Thick arrows represent communications involving vectors, whereas thin arrows stand for communications of scalars. Matrix E_j is defined at Eq. (2.66).

which needs to be carried out on a central node. In Figure 2.7, we give a sketch of the workflow at a generic node j . One can notice that, in terms of local computations, TRAP behaves as a standard two-phase approach on every node.

2.2.2 Convergence analysis

The analysis of TRAP that follows is along the lines of the convergence proof of trust region methods in [26], where the Cauchy point is computed via a projected search, which involves a sequence of evaluations of the model function on a central node. However, for TRAP, the fact that the Cauchy point is yielded by a distributed projected gradient step on the model function requires some modifications in the analysis. Namely, the lower bound on the decrease in the model and the upper bound on criticality at the Cauchy point are expressed in a rather different way. However, the arguments behind the global convergence proof are essentially the same as in [26].

In this section, for theoretical purposes only, another first-order criticality measure different from (2.52) is used. We utilise the condition that $x^* \in \Omega$ is a first-order critical point if the projected gradient at x^* is zero,

$$\nabla_{\Omega} L(x^*) = 0 \quad , \quad (2.62)$$

where, given $x \in \Omega$, the projected gradient is defined as

$$\nabla_{\Omega} L(x) := P_{\mathcal{T}_{\Omega}(x)}(-g(x)) \ .$$

Discussions on this first-order criticality measure can be found in [36]. It is equivalent to the standard optimality condition

$$\langle g(x^*), x - x^* \rangle \geq 0, \text{ for all } x \in \Omega \ . \quad (2.63)$$

It follows from Moreau's decomposition that a point x^* satisfying (2.62) automatically satisfies (2.52). Consequently, it is perfectly valid to use (2.52) for the convergence analysis of TRAP.

2.2.2.1 Global convergence to first-order critical points

We start with an estimate of the block-coordinate model decrease provided by the Cauchy points $z_k, k \in \{1, \dots, K\}$, of Algorithm 5. For this purpose, we define for all $k \in \{1, \dots, K\}$,

$$m_k(x') := m(z_{[1, k-1]}, x', x_{[k+1, K]}) \ , \quad (2.64)$$

where $x' \in \mathbb{R}^{n_k}$. This corresponds to the model function evaluated at x' with the block-coordinates 1 to $k-1$ being fixed to the associated Cauchy points z_1, \dots, z_{k-1} and the block-coordinates $k+1$ to K having values x_{k+1}, \dots, x_K . Note that by definition of the function m_k ,

$$m_k(z_k) = m_{k+1}(x_{k+1}) \ ,$$

for all $k \in \{1, \dots, K-1\}$.

Lemma 2.7. *There exists a constant $\chi > 0$ with respect to the iteration index k , such that, for all $k \in \{1, \dots, K\}$,*

$$m_k(x_k) - m_k(z_k) \geq \chi \frac{\|z_k - x_k\|_2}{\alpha_k} \min \left\{ \Delta, \frac{1}{1 + \|B(x)\|_2} \frac{\|z_k - x_k\|_2}{\alpha_k} \right\} \ . \quad (2.65)$$

Proof. The proof goes along the same lines as the one of Theorem 4.3 in [115]. Yet, some arguments differ, due to the alternating projections. We first assume that condition (2.55) is satisfied with

$$\alpha_k \geq \nu_4 \ .$$

Using the variational characterisation of the projection onto a closed and convex set (2.63), we

obtain

$$m_k(x_k) - m_k(z_k) \geq \nu_0 \nu_4 \frac{\|z_k - x_k\|_2^2}{\alpha_k^2} .$$

We then consider the second case in (2.56) when

$$\alpha_k \geq \nu_3 \tilde{\alpha}_k .$$

The first possibility (2.57) is then

$$m_k(z_k(\tilde{\alpha}_k)) - m_k(x_k) > \nu_0 \langle \nabla m_k(x_k), z_k(\tilde{\alpha}_k) - x_k \rangle .$$

However, by definition of the model function in Eq. (2.53), the left-hand side term in the above inequality is equal to

$$\begin{aligned} & \langle g_k(x), z_k(\tilde{\alpha}_k) - x_k \rangle + \frac{1}{2} \langle z_k(\tilde{\alpha}_k) - x_k, E_k B(x) E_k^\top (z_k(\tilde{\alpha}_k) - x_k) \rangle \\ & \quad + \langle \tilde{z}_{[1,k-1]} - x_{[1,k-1]}, E_{[1,k-1]} B(x) E_{[1,k-1]}^\top (\tilde{z}_k - x_k) \rangle \\ & = \frac{1}{2} \langle z_k(\tilde{\alpha}_k) - x_k, E_k B(x) E_k^\top (z_k(\tilde{\alpha}_k) - x_k) \rangle + \langle \nabla m_k(x_k), z_k(\tilde{\alpha}_k) - x_k \rangle , \end{aligned}$$

where, given $k \in \{1, \dots, K\}$, the matrix $E_k \in \mathbb{R}^{n_k \times n}$ is such that for $i \in \{1, \dots, n_k\}$,

$$E_k(i, n_1 + \dots + n_{k-1} + i) = 1 , \tag{2.66}$$

and all other entries are zero. This yields, by the Cauchy-Schwarz inequality

$$\begin{aligned} \frac{\|B(x)\|_2}{2} \|z_k(\tilde{\alpha}_k) - x_k\|_2^2 & > -(1 - \nu_0) \langle \nabla m_k(x_k), z_k(\tilde{\alpha}_k) - x_k \rangle \\ & \geq \frac{1 - \nu_0}{\tilde{\alpha}_k} \|z_k(\tilde{\alpha}_k) - x_k\|_2^2 \end{aligned}$$

Hence,

$$\tilde{\alpha}_k \geq \frac{2(1 - \nu_0)}{1 + \|B(x)\|_2} .$$

The second possibility (2.58) is

$$\|z_k(\tilde{\alpha}_k) - x_k\|_\infty \geq \nu_1 \Delta .$$

For this case, [115] provides the lower bound

$$\|z_k - x_k\|_2 \geq \nu_3 \nu_1 \Delta .$$

Finally, inequality (2.65) holds with

$$\chi := \nu_0 \min \{ \nu_4, 2(1 - \nu_0) \nu_3, \nu_3 \nu_1 \} .$$

□

From Lemma 2.7, an estimate of the decrease in the model provided by the Cauchy point z is derived.

Corollary 2.1 (Sufficient decrease). *The following inequality holds*

$$m(x) - m(z) \geq \chi \sum_{k=1}^K \frac{\|z_k - x_k\|_2}{\alpha_k} \min \left\{ \Delta, \frac{1}{1 + \|B(x)\|_2} \frac{\|z_k - x_k\|_2}{\alpha_k} \right\} . \quad (2.67)$$

Proof. This is a direct consequence of Lemma 2.7 above, as

$$m(x) - m(z) = \sum_{k=1}^K m_k(x_k) - m_k(z_k) .$$

from the definition of m_k in Eq. (2.64). □

In a similar manner to [26], the level of criticality reached by the Cauchy point z is measured by the norm of the projected gradient of the objective, which can be upper bounded by the difference between the current iterate x and the Cauchy point z .

Lemma 2.8 (Relative error condition). *The following inequality holds*

$$\|\nabla_{\Omega} L(z)\|_2 \leq K \|B(x)\|_2 \|z - x\|_2 + \sum_{k=1}^K \left(\frac{\|z_k - x_k\|_2}{\alpha_k} + \|g_k(z) - g_k(x)\|_2 \right) , \quad (2.68)$$

where g_k stands for $\nabla_k L$.

Proof. From the definition of z_k as the projection of

$$x_k - \alpha_k \nabla m_k(x_k)$$

onto the closed convex set Ω_k , there exists $v_k \in \mathcal{N}_{\Omega_k}(z_k)$ such that

$$0 = v_k + \nabla m_k(x_k) + \frac{z_k - x_k}{\alpha_k} .$$

Hence,

$$\|v_k + g_k(z)\|_2 \leq \|g_k(z) - g_k(x)\|_2 + \|B(x)\|_2 \|z - x\|_2 + \frac{\|z_k - x_k\|_2}{\alpha_k}$$

However,

$$\left\| P_{\mathcal{N}_{\Omega_k}(z_k)}(-g_k(z)) + g_k(z) \right\|_2 \leq \|v_k + g_k(z)\|_2 ,$$

and by Moreau's decomposition theorem,

$$-g_k(z) = P_{\mathcal{N}_{\Omega_k}(z_k)}(-g_k(z)) + P_{\mathcal{T}_{\Omega_k}(z_k)}(-g_k(z)) .$$

Thus,

$$\left\| P_{\mathcal{T}_{\Omega_k}(z_k)}(-g_k(z)) \right\|_2 \leq \|g_k(z) - g_k(x)\|_2 + \|B(x)\|_2 \|z - x\|_2 + \frac{\|z_k - x_k\|_2}{\alpha_k} .$$

As the sets $\{\Omega_k\}_{k=1}^K$ are closed and convex,

$$\mathcal{T}_{\Omega}(z) = \mathcal{T}_{\Omega_1}(z_1) \times \dots \times \mathcal{T}_{\Omega_K}(z_K) .$$

Subsequently,

$$\|\nabla_{\Omega} L(z)\|_2 \leq \sum_{k=1}^K \left\| P_{\mathcal{T}_{\Omega_k}(z_k)}(-g_k(z)) \right\|_2$$

and inequality (2.68) follows. \square

Based on the estimate of the model decrease (2.67) and the relative error bound (2.68) at the Cauchy point z , one can follow the standard proof mechanism of trust region methods quite closely [26]. Most of the steps are proven by contradiction, assuming that criticality is not reached. The nature of the model decrease (2.67) is well-suited to this type of reasoning. Hence, most of the ideas of [26] can be adapted to our setting.

Lemma 2.9. *If Assumptions 2.10, 2.12 and 2.13 are satisfied, then the sequence of iterates yielded*

by Algorithm 5 satisfies that for all $k \in \{1, \dots, K\}$,

$$\liminf \frac{\|z_k - x_k\|_2}{\alpha_k} = 0, \quad (2.69)$$

Proof. For the sake of contradiction, assume that there exists a block index $k_0 \in \{1, \dots, K\}$ and $\epsilon > 0$ such that

$$\frac{\|z_{k_0}^l - x_{k_0}^l\|_2}{\alpha_{k_0}^l} \geq \epsilon$$

for all iteration indices $l \geq 1$. Using Corollary 2.1, the standard proof mechanism of trust region methods [26] can be easily adapted to obtain (2.69). \square

We are now ready to state the main Theorem of this section. It is claimed that all limit points of the sequence $\{x^l\}$ generated by TRAP are critical points of (2.51).

Theorem 2.3 (Limit points are critical points). *Assume that Assumptions 2.10, 2.12 and 2.13 hold. If x^* is a limit point of $\{x^l\}$, then there exists a subsequence $\{l_i\}$ such that*

$$\begin{cases} \lim_{i \rightarrow +\infty} \|\nabla_{\Omega} L(z^{l_i})\|_2 = 0 \\ z^{l_i} \rightarrow x^* \end{cases}. \quad (2.70)$$

Moreover, $\nabla_{\Omega} L(x^*) = 0$, meaning that x^* is a critical point of $L + \iota_{\Omega}$.

Proof. Let $\{x^{l_i}\}$ be a subsequence of $\{x^l\}$ such that $x^{l_i} \rightarrow x^*$. If for all $k \in \{1, \dots, K\}$

$$\frac{\|z_k^{l_i} - x_k^{l_i}\|_2}{\alpha_k^{l_i}} \rightarrow 0, \quad (2.71)$$

then the proof is complete, via Lemma 2.8 and the fact that the step-sizes α_k are upper bounded by ν_5 . In order to show (2.71), given $\epsilon > 0$ one can assume that there exists $k_0 \in \{1, \dots, K\}$ such that for all $i \geq 1$, $\|z_{k_0}^{l_i} - x_{k_0}^{l_i}\|_2 / \alpha_{k_0}^{l_i} \geq \epsilon$. One can then easily combine the arguments in the proof of Theorem 5.4 in [26] with Corollary 2.1 and Lemma 2.8 in order to obtain (2.70). \square

Theorem 2.3 above proves that all limit points of the sequence $\{x^l\}$ generated by TRAP are critical points. It does not actually claim convergence of $\{x^l\}$ to a single critical point. However, such a result can be obtained under standard regularity assumptions [119], which ensure that a critical point is an isolated local minimum.

Assumption 2.15 (Strong second-order optimality condition). *The objective function L is twice continuously differentiable and its hessian $\nabla^2 L(x)$ is denoted by $H(x)$. The sequence $\{x^l\}$ yielded by TRAP has a non-degenerate limit point x^* such that for all $v \in \mathcal{N}_\Omega(x^*)^\perp$, where*

$$\mathcal{N}_\Omega(x^*)^\perp := \{v \in \mathbb{R}^n : \forall w \in \mathcal{N}_\Omega(x^*), \langle w, v \rangle = 0\} , \quad (2.72)$$

one has

$$\langle v, H(x^*)v \rangle \geq \kappa \|v\|_2^2 , \quad (2.73)$$

where $\kappa > 0$.

Theorem 2.4 (Convergence to first-order critical points). *If Assumptions 2.13, 2.10, 2.12 and 2.15 are fulfilled, then the sequence $\{x^l\}$ generated by TRAP converges to a non-degenerate critical point x^* of $L + \iota_\Omega$.*

Proof. This is an immediate consequence of Corollary 6.7 in [26]. \square

2.2.2.2 Active-set identification

In most of the trust region algorithms for constrained optimisation, the Cauchy point acts as a predictor of the set of active constraints at a critical point. Therefore, a desirable feature of the novel Cauchy point computation in TRAP is finite detection of activity, meaning that the active set at the limit point is identified after a finite number of iterations. In this paragraph, we show that TRAP is equivalent to the standard projected search in terms of identifying the active set at the critical point x^* defined in Theorem 2.4.

Lemma 2.10. *As Ω is a polyhedral set, given one of its faces \mathcal{F} , there exists faces $\mathcal{F}_1, \dots, \mathcal{F}_K$ of $\Omega_1, \dots, \Omega_K$ respectively, such that $\mathcal{F} = \mathcal{F}_1 \times \dots \times \mathcal{F}_K$ [163].*

Remark 2.9. *Given a point $x \in \Omega$, there exists a face \mathcal{F} of Ω such that $x \in \text{ri}(\mathcal{F})$. The normal cone to Ω at x is the cone generated by the normal vectors to the active constraints at x . As the set of active constraints is constant on the relative interior of a face, one can write without distinction $\mathcal{N}_\Omega(x)$ or $\mathcal{N}(\mathcal{F})$.*

The following Lemma is similar in nature to Lemma 7.1 in [26], yet with an adaptation in order to account for the novel way of computing the Cauchy point. In particular, it is only valid for a sufficiently high iteration count, contrary to Lemma 7.1 of [26], which can be written independently of the iteration count. This is essentially due to the fact that the Cauchy point is computed via an alternating projected search, contrary to [26], where a centralised projected search is performed.

Lemma 2.11. *Assume that Assumptions 2.13, 2.10, 2.12 and 2.15 hold. Let x^* be a non-degenerate critical point of (2.51) that belongs to the relative interior of a face \mathcal{F}^* of Ω . Let $\{\mathcal{F}_k^*\}_{k=1}^K$ be faces of $\{\Omega_k\}_{k=1}^K$ such that $\mathcal{F}^* = \mathcal{F}_1^* \times \dots \times \mathcal{F}_K^*$ and thus $x_k^* \in \text{ri}(\mathcal{F}_k^*)$, for all $k \in \{1, \dots, K\}$.*

Assume that $x^l \rightarrow x^$. For l sufficiently large, for all $k \in \{1, \dots, K\}$ and all $\alpha_k > 0$, there exists $\epsilon_k > 0$ such that*

$$x_k^l \in \mathcal{B}(x_k^*, \epsilon_k) \cap \text{ri}(\mathcal{F}_k^*) \implies P_{\Omega_k}(x_k^l - t_k \nabla m_k(x_k^l)) \in \text{ri}(\mathcal{F}_k^*) \quad ,$$

for all $t_k \in]0, \alpha_k]$.

Proof. Similarly to the proof of Lemma 7.1 in [26], the idea is to show that there exists a neighbourhood of x_k^* such that if x_k^l lies in this neighbourhood, then

$$x_k^l - \alpha_k \nabla m_k(x_k^l) \in \text{ri}(\mathcal{F}_k^* + \mathcal{N}(\mathcal{F}_k^*)) \quad .$$

Lemma 2.11 then follows by using the properties of the projection operator onto a closed convex set and Theorem 2.3 in [26].

For simplicity, we prove the above relation for $k = 2$. It can be trivially extended to all indices k in $\{3, \dots, K\}$. Let $\alpha_2 > 0$ and $l \geq 1$.

$$x_2^l - \alpha_2 \nabla m_2(x_2^l) = x_2^l - \alpha_2 g_2(x^l) - \alpha_2 E_2 B(x^l) E_1^\top (z_1^l - x_1^l) \quad ,$$

where the matrix E_k is defined in (2.66). As x^* is non-degenerate,

$$x^* - \alpha_2 g(x^*) \in \text{ri}(\mathcal{F}^*) + \text{ri}(\mathcal{N}(\mathcal{F}^*)) \quad .$$

However, as the sets $\{\mathcal{F}_k^*\}_{k=1}^K$ are convex, one has [138]

$$\text{ri}(\mathcal{F}^*) = \text{ri}(\mathcal{F}_1^*) \times \dots \times \text{ri}(\mathcal{F}_K^*) \quad \text{and} \quad \mathcal{N}(\mathcal{F}^*) = \mathcal{N}(\mathcal{F}_1^*) \times \dots \times \mathcal{N}(\mathcal{F}_K^*) \quad .$$

Hence,

$$x_2^* - \alpha_2 g_2(x^*) \in \text{ri}(\mathcal{F}_2^*) + \text{ri}(\mathcal{N}(\mathcal{F}_2^*)) = \text{int}(\mathcal{F}_2^* + \mathcal{N}(\mathcal{F}_2^*)) \quad ,$$

by Theorem 2.3 in [26]. By continuity of the objective gradient g , there exists $\delta_2 > 0$ such that

$$\|x^l - x^*\|_2 < \delta_2 \implies x_2^l - \alpha_2 g_2(x^l) \in \text{int}(\mathcal{F}_2^* + \mathcal{N}(\mathcal{F}_2^*)) \quad .$$

However, as shown beforehand (Lemma 2.9),

$$\lim_{l \rightarrow +\infty} \|z_1^l - x_1^l\|_2 = 0 .$$

Moreover, $E_2 B(x^l) E_1^\top$ is bounded above (Ass. 2.13), subsequently for l sufficiently large,

$$x_2^l - \alpha_2 \nabla m_2(x_2^l) \in \text{int}(\mathcal{F}_2^* + \mathcal{N}(\mathcal{F}_2^*)) \subseteq \text{ri}(\mathcal{F}_2^* + \mathcal{N}(\mathcal{F}_2^*)) ,$$

by Theorem 2.3 in [26]. Then, Lemma 2.11 follows by properly choosing the radii ϵ_k so that

$$\sum_{k=1}^K \epsilon_k^2 = \left(\min \{ \delta_k \}_{k=1}^K \right)^2 .$$

□

We have just shown that, for a sufficiently large iteration count l , if the primal iterate x^l is sufficiently close to the critical point x^* and on the same face \mathcal{F}^* , then the set of active constraints at the Cauchy point z^l is the same as the set of active constraints at x^* .

Theorem 2.5. *If Assumptions 2.13, 2.10, 2.12 and 2.15 are fulfilled, then the following holds*

$$\lim_{l \rightarrow +\infty} \|\nabla_{\Omega} L(x^l)\|_2 = 0 .$$

Moreover, there exists l_0 such that for all $l \geq l_0$,

$$\mathcal{A}_{\Omega}(x^l) = \mathcal{A}_{\Omega}(x^*) .$$

Proof. The reasoning of the proof of Theorem 7.2 in [26] can be applied using Lemma 2.11 and line 13 in Algorithm 5. The first step is to show that the Cauchy point z identifies the optimal active set after a finite number of iterations. This is guaranteed by Theorem 2.2 in [26], since $\nabla_{\Omega} L(z^l) \rightarrow 0$ by Theorem 2.3, and the sequence $\{x^l\}$ converges to a non-degenerate critical point by Theorem 2.4. Lemma 2.11 is used to show that if x^l is sufficiently close to x^* , then the Cauchy point z^l remains in the relative interior of the same face, and thus the active constraints do not change after some point. □

Theorem 2.5 shows that the optimal active set is identified after a finite number of iterations, which corresponds to the behaviour of the gradient projection in standard trust region methods. This fact is crucial for the local convergence analysis of the sequence $\{x^l\}$, as a fast local convergence rate cannot be obtained if the dynamics of the active constraints does not settle down.

2.2.2.3 Local convergence rate

In this paragraph, we show that the local convergence rate of the sequence $\{x^l\}$ generated by TRAP is almost Q-superlinear, in the case where a Newton model is approximately minimised at every trust region iteration, that is

$$B = \nabla^2 L ,$$

in model (2.53). Similarly to (2.61), one can define

$$H_\sigma := H + \frac{\sigma}{2} I . \quad (2.74)$$

To establish fast local convergence, a key step is to prove that the trust region radius is ultimately bounded away from zero. It turns out that the regularisation of the trust region problem (2.59) plays an important role in this proof. As shown in the next Lemma 2.12, after a sufficiently large number of iterations, the trust region radius does not interfere with the iterates and an inexact Newton step is always taken at the refinement stage (Line 10 to 13), implying fast local convergence depending on the level of accuracy in the computation of the Newton direction. However, Theorem 7.4 in [26] cannot be applied here, since due to the alternating gradient projections, the model decrease at the Cauchy point cannot be expressed in terms of the projected gradient on the active face at the critical point.

Lemma 2.12. *If Assumptions 2.13, 2.10, 2.12 and 2.15 are fulfilled, then there exists an index $l_1 \geq 1$ and $\Delta^* > 0$ such that for all $l \geq l_1$, $\Delta^l \geq \Delta^*$.*

Proof. The idea is to show that the ratio ρ converges to one, which implies that all iterations are ultimately successful, and subsequently, by the mechanism of Algorithm 5, the trust region radius is bounded away from zero asymptotically. For all $l \geq 1$,

$$|\rho^l - 1| = \frac{\left| L(y^l) - L(x^l) - \langle g(x^l), y^l - x^l \rangle - \frac{1}{2} \langle y^l - x^l, H(x^l)(y^l - x^l) \rangle \right|}{m(x^l) - m(y^l)} . \quad (2.75)$$

However,

$$\begin{aligned}
 m^l(x^l) - m^l(y^l) &= m^l(x^l) - m^l(z^l) + m^l(z^l) - m^l(y^l) \\
 &\geq \frac{\eta}{2} \|z^l - x^l\|_2^2 + \frac{\sigma}{2} \|y^l - z^l\|_2^2 \\
 &\geq \frac{\min\{\eta, \sigma\}}{2} (\|z^l - x^l\|_2^2 + \|y^l - z^l\|_2^2) \\
 &\geq \frac{\min\{\eta, \sigma\}}{2} \max\{\|z^l - x^l\|_2^2, \|y^l - z^l\|_2^2\} ,
 \end{aligned}$$

and

$$\begin{aligned}
 \|p^l\|_2 &\leq \|y^l - z^l\|_2 + \|z^l - x^l\|_2 \\
 &\leq 2 \max\{\|y^l - z^l\|_2, \|z^l - x^l\|_2\} .
 \end{aligned}$$

Hence,

$$m^l(x^l) - m^l(y^l) \geq \frac{\min\{\eta, \sigma\}}{8} \|p^l\|_2^2 .$$

Moreover, using the mean-value theorem, one obtains that the numerator in (2.75) is smaller than

$$\frac{1}{2} \psi^l \|p^l\|_2^2 ,$$

where

$$\psi^l := \sup_{\tau \in [0,1]} \|H(x^l + \tau p^l) - H(x^l)\|_2 . \quad (2.76)$$

Subsequently, we have

$$|\rho^l - 1| \leq \frac{4}{\min\{\eta, \sigma\}} \psi^l ,$$

and the result follows by showing that p^l converges to zero. Take $l \geq l_0$, where l_0 is as in Theorem 2.5. Thus, $p^l \in \mathcal{N}(\mathcal{F}^*)^\perp$. However, from the model decrease, one obtains

$$\frac{1}{2} \langle p^l, H(x^l) p^l \rangle \leq \langle -g(x^l), p^l \rangle .$$

From Theorem 2.4, the sequence $\{x^l\}$ converges to x^* , which satisfies the strong second-order optimality condition 2.15. Hence, by continuity of the hessian $\nabla^2 L$ and the fact that $\mathcal{A}_\Omega(x^l) =$

$\mathcal{A}_\Omega(x^*)$, one can claim that there exists $l_1 \geq l_0$ such that for all $l \geq l_1$, for all $v \in \mathcal{N}_\Omega(x^l)^\perp = \mathcal{N}(\mathcal{F}^*)^\perp$,

$$\langle v, H(x^l)v \rangle \geq \kappa \|v\|_2^2 .$$

Thus, by Moreau's decomposition, it follows that

$$\begin{aligned} \frac{\kappa}{2} \|p^l\|_2^2 &\leq \left\langle P_{\mathcal{T}_\Omega(x^l)}(-g(x^l)) + P_{\mathcal{N}_\Omega(x^l)}(-g(x^l)), p^l \right\rangle \\ &\leq \left\| P_{\mathcal{T}_\Omega(x^l)}(-g(x^l)) \right\|_2 \|p^l\|_2 , \end{aligned}$$

since $p^l \in \mathcal{N}(\mathcal{F}^*)^\perp$. Finally, p^l converges to zero, as a consequence of Lemma 2.8 and the fact that $\|z^l - x^l\|_2$ converges to 0, by Lemma 2.8 and the fact that the step-sizes α_k are upper bounded for $k \in \{1, \dots, K\}$. \square

The refinement step in TRAP actually consists of a truncated Newton method, in which the Newton direction is generated by an iterative procedure, namely the distributed sCG described in Algorithm 6. The Newton iterations terminate when the residual \hat{s} is below a tolerance that depends on the norm of the projected gradient at the current iteration. In Algorithm 6, the stopping condition is set so that at every iteration $l \geq 1$, there exists $\xi^l \in]0, 1[$ satisfying

$$\left\| Z^l (Z^l)^\top (g_{\sigma^l}(x^l) + H_{\sigma^l}(x^l)p^l) \right\|_2 \leq \xi^l \left\| Z^l (Z^l)^\top g(x^l) \right\|_2 . \quad (2.77)$$

The local convergence rate of the sequence $\{x^l\}$ generated by TRAP is controlled by the sequences $\{\xi^l\}$ and $\{\sigma^l\}$, as shown in the following Theorem.

Theorem 2.6 (Local linear convergence). *Assume that the direction p yielded by Algorithm 6 satisfies (2.77) if $\|p\|_\infty \leq \gamma^* \Delta$ and $\mathcal{A}_\Omega(x) = \mathcal{A}_\Omega(x+p)$, given $\gamma^* \in]0, \gamma_2[$. Under Assumptions 2.13, 2.10, 2.12 and 2.15, for a sufficiently small $\bar{\sigma}$, the sequence $\{x^l\}$ generated by TRAP converges Q -linearly to x^* if $\xi^* < 1$ is sufficiently small, where*

$$\xi^* := \limsup_{l \rightarrow +\infty} \xi^l .$$

If $\xi^ = 0$, the Q -linear convergence ratio can be made arbitrarily small by properly choosing $\bar{\sigma}$, resulting in almost Q -superlinear convergence.*

Proof. Throughout the proof, we assume that l is sufficiently large so that the active-set is $\mathcal{A}_\Omega(x^*)$ and that p^l satisfies condition (2.77). This is ensured by Lemma 2.12 and Theorem 2.5, as the sequence $\{p^l\}$ converges to zero. Thus, we can write $Z^l = Z^*$. The orthogonal projection onto

the subspace $\mathcal{N}(\mathcal{F}^*)^\perp$ is represented by the matrix $Z^*(Z^*)^\top$. A first-order development yields a positive sequence $\{\delta^l\}$ converging to zero such that

$$\begin{aligned} \|Z^*(Z^*)^\top g(x^{l+1})\|_2 &\leq \|Z^*(Z^*)^\top (g(x^l) + H(x^l)p^l)\|_2 + \delta^l \|p^l\|_2 \\ &\leq \frac{2\delta^l}{\kappa} \|Z^*(Z^*)^\top g(x^l)\|_2 + \|Z^*(Z^*)^\top (g_{\sigma^l}(x^l) + H_{\sigma^l}(x^l)p^l)\|_2 \\ &\quad + \bar{\sigma} \left\| Z^*(Z^*)^\top \left(\frac{p^l}{2} + z^l - x^l \right) \right\|_2 \\ &\leq \left(\frac{2\delta^l}{\kappa} + \xi^l \right) \|Z^*(Z^*)^\top g(x^l)\|_2 \\ &\quad + \bar{\sigma} \left(\frac{1}{\kappa} + \frac{\|Z^*(Z^*)^\top (z^l - x^l)\|_2}{\|Z^*(Z^*)^\top g(x^l)\|_2} \right) \|Z^*(Z^*)^\top g(x^l)\|_2 . \end{aligned}$$

where the second inequality follows from the last inequality in Lemma 2.12, and the definition of g_σ in Eq. (2.61) and H_σ in Eq. (2.74). However, from the computation of the Cauchy point described in paragraph 2.2.1 and Assumption 2.13, the term

$$\frac{\|Z^*(Z^*)^\top (z^l - x^l)\|_2}{\|Z^*(Z^*)^\top g(x^l)\|_2}$$

is bounded by a constant $C > 0$. Hence,

$$\frac{\|Z^*(Z^*)^\top g(x^{l+1})\|_2}{\|Z^*(Z^*)^\top g(x^l)\|_2} \leq \frac{2\delta^l}{\kappa} + \xi^l + \bar{\sigma} \left(\frac{1}{\kappa} + C \right) .$$

Moreover, a first-order development provides us with a constant $\Upsilon > 0$ such that

$$\|Z^*(Z^*)^\top g(x^l)\|_2 \leq (\hat{B} + \Upsilon) \|x^l - x^*\|_2 .$$

There also exists a positive sequence $\{\epsilon^l\}$ converging to zero such that

$$\|Z^*(Z^*)^\top g(x^{l+1})\|_2 \geq \|Z^*(Z^*)^\top H(x^*) (x^{l+1} - x^*)\|_2 - \epsilon^l \|x^{l+1} - x^*\|_2 .$$

However, since $x^{l+1} - x^*$ lies in $\mathcal{N}(x^*)^\perp$, $Z^*(Z^*)^\top (x^{l+1} - x^*) = x^{l+1} - x^*$. Thus, by Assumption (2.73),

$$\|Z^*(Z^*)^\top \nabla L(x^{l+1})\|_2 \geq (\kappa - \epsilon^l) \|x^{l+1} - x^*\|_2 ,$$

which implies that, for l sufficiently large, there exists $\bar{\epsilon} \in]0, \kappa[$ such that

$$\|Z^* (Z^*)^\top g(x^{l+1})\|_2 \geq (\kappa - \bar{\epsilon}) \|x^{l+1} - x^*\|_2 .$$

Finally,

$$\frac{\|x^{l+1} - x^*\|_2}{\|x^l - x^*\|_2} \leq \frac{\hat{B} + \Upsilon}{\kappa - \bar{\epsilon}} \left(\frac{2\delta^l}{\kappa} + \xi^l + \bar{\sigma} \left(\frac{1}{\kappa} + C \right) \right) ,$$

which yields the result. \square

2.2.3 Numerical experiments

The optimal AC power flow constitutes a challenging class of nonconvex problems for benchmarking optimisation algorithms and software. It has been used very recently in the testing of a novel adaptive augmented Lagrangian technique [39]. The power flow equations form a set of nonlinear coupling constraints over a network. Some distributed optimisation strategies have already been explored for computing OPF solutions, either based on convex relaxations [104] or nonconvex heuristics [100]. As the convex relaxation may fail in a significant number of cases [25], it is also relevant to explore distributed strategies for solving the OPF in its general nonconvex formulation. Naturally, all that we can hope for with this approach is a local minimum of the OPF problem. Algorithm 5 is tested on the augmented Lagrangian subproblems obtained via a polar coordinates formulation of the OPF equations, as well as rectangular coordinates formulations. Our TRAP algorithm is run as an inner solver inside a standard augmented Lagrangian loop [16] and in the more sophisticated LANCELOT dual loop [34]. More precisely, if the OPF problem is written in the following form

$$\begin{aligned} & \underset{x}{\text{minimise}} f(x) & (2.78) \\ & \text{s. t. } g(x) = 0 \\ & x \in \mathcal{X} , \end{aligned}$$

where \mathcal{X} is a bound constraint set, an augmented Lagrangian loop consists in computing an approximate critical point of the auxiliary program

$$\underset{x \in \mathcal{X}}{\text{minimise}} L_\varrho(x, \mu) := f(x) + \left(\mu + \frac{\varrho}{2} g(x) \right)^\top g(x) \quad (2.79)$$

with μ a dual variable associated to the power flow constraints and $\varrho > 0$ a penalty parameter, which are both updated after a finite sequence of primal iterations in (2.79). Using the standard first-order dual update formula, only local convergence of the dual sequence can be proven [16]. On the contrary, in the LANCELOT outer loop, the dual variable μ and the penalty parameter ϱ are updated according to the level of satisfaction of the power flow (equality) constraints, resulting in global convergence of the dual sequence [34]. In order to test TRAP, we use it to compute approximate critical points of the subproblems (2.78), which are of the form (2.51). The rationale behind choosing LANCELOT instead of a standard augmented Lagrangian method as the outer loop is that LANCELOT interrupts the inner iterations at an early stage, based on a KKT tolerance that is updated at every dual iteration. Hence, it does not allow one to really measure the absolute performance of TRAP, although it is likely more efficient than a standard augmented Lagrangian for computing a solution of the OPF program. Thus, for all cases presented next, we provide the results of the combination of TRAP with a basic augmented Lagrangian and LANCELOT. The augmented Lagrangian loop is utilised to show the performance of TRAP as a bound-constrained solver, whereas LANCELOT is expected to provide better overall performance. All results are compared to the solution yielded by the nonlinear interior-point solver IPOPT [152] with the sparse linear solver MA27. Finally, it is important to stress that the results presented in this Section are obtained from a preliminary MATLAB implementation, which is designed to handle small-scale problems. The design of a fully distributed software would involve substantial development and testing, and is thus beyond the scope of this study.

2.2.3.1 Optimal AC power flow in polar coordinates

We consider the AC-OPF problem in polar coordinates

$$\begin{aligned}
 & \text{minimise } \sum_{g \in \mathcal{G}} c_0^g + c_1^g p_g^G + c_2 (p_g^G)^2 & (2.80) \\
 & \text{s. t.} \\
 & \sum_{g \in \mathcal{G}_b} p_g^G = \sum_{d \in \mathcal{D}_b} P_d^D + \sum_{b' \in \mathcal{B}_b} p_{bb'}^L + G_b^B v_b^2 \\
 & \sum_{g \in \mathcal{G}_b} q_g^G = \sum_{d \in \mathcal{D}_b} Q_d^D + \sum_{b' \in \mathcal{B}_b} q_{bb'}^L - B_b^B v_b^2 \\
 & p_{bb'}^L = G_{bb} v_b^2 + (G_{bb'} \cos(\theta_b - \theta_{b'}) + B_{bb'} \sin(\theta_b - \theta_{b'})) v_b v_{b'} \\
 & q_{bb'}^L = -B_{bb} v_b^2 + (G_{bb'} \sin(\theta_b - \theta_{b'}) - B_{bb'} \cos(\theta_b - \theta_{b'})) v_b v_{b'} \\
 & (p_{bb'}^L)^2 + (q_{bb'}^L)^2 + s_{bb'} = (S_{bb'}^M)^2 \\
 & v_b^L \leq v_b \leq v_b^U \\
 & p^L \leq p_g^G \leq p^U \\
 & q^L \leq q_g^G \leq q^U \\
 & s_{bb'} \geq 0,
 \end{aligned}$$

which corresponds to the minimisation of the overall generation cost, subject to power balance constraints at every bus b and power flow constraints on every line bb' of the network, where \mathcal{G} denotes the set of generators and \mathcal{G}_b is the set of generating units connected to bus b . The variables p_g^G and q_g^G are the active and reactive power output at generator g . The set of loads connected to bus b is denoted by \mathcal{D}_b . The parameters P_d^D and Q_d^D are the demand active and reactive power at load unit d . The letter \mathcal{B}_b represents the set of buses connected to bus b . Variables $p_{bb'}^L$ and $q_{bb'}^L$ are the active and reactive power flow through line bb' . Variables v_b and θ_b denote the voltage magnitude and voltage angle at bus b . Constants v_b^L, v_b^U are lower and upper bounds on the voltage magnitude at bus b . Constants p^L, p^U, q^L and q^U are lower and upper bounds on the active and reactive power generation. It is worth noting that a slack variable $s_{bb'}$ has been added at every line bb' in order to turn the usual inequality constraint on the power flow through line bb' into an equality constraint. The derivation of the optimal power flow problem in polar form can be found in [162].

As a simple numerical test example for TRAP, we consider a particular instance of NLP (2.80) on the 9-bus transmission network shown in Fig. 2.8. As in (2.79), the augmented Lagrangian subproblem is obtained by relaxing the equality constraints associated with buses and lines in (2.80). The bound constraints, which can be easily dealt with via projection, remain unchanged. One should notice that NLP (2.80) has partially separable constraints and objective, so that LANCELOT

could efficiently deal with it, yet in a purely centralised manner. In some sense, running TRAP in a LANCELOT outer loop can be seen as a first step towards a distributed implementation of LANCELOT for solving the AC-OPF problem. It is worth noting that the dual updates only require exchange of information between neighbouring nodes and lines. However, each LANCELOT dual update requires a central communication, as the norm of the power flow constraints need to be compared with a running tolerance [34]. For the 9-bus example in Fig. 2.8, the Cauchy search of TRAP

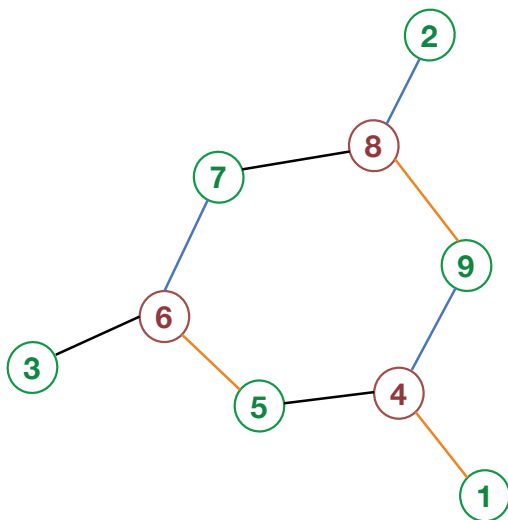


Figure 2.8: The 9-bus transmission network from <http://www.maths.ed.ac.uk/optenergy/LocalOpt/>.

on the augmented Lagrangian subproblem (2.79) can be carried out in five parallel steps. This can be observed by introducing local variables for every bus $b \in \{1, \dots, 9\}$,

$$x_b := (v_b, \theta_b)^\top ,$$

and for every line

$$bb' \in \left\{ \{1, 4\}, \{4, 5\}, \{4, 9\}, \{8, 9\}, \{2, 8\}, \{7, 8\}, \{6, 7\}, \{3, 6\}, \{5, 6\} \right\} ,$$

with the line variable $y_{bb'}$ being defined as

$$y_{bb'} := (p_{bb'}, q_{bb'}, s_{bb'})^\top .$$

The line variables $y_{bb'}$ can be first updated in three parallel steps, which corresponds to

$$\{y_{\{2,8\}}, y_{\{6,7\}}, y_{\{4,9\}}\}, \{y_{\{7,8\}}, y_{\{3,6\}}, y_{\{4,5\}}\}, \{y_{\{8,9\}}, y_{\{5,6\}}, y_{\{1,4\}}\} .$$

Then, the subset

$$\{x_1, x_2, x_3, x_5, x_7, x_9\}$$

can be updated, followed by the subset

$$\{x_4, x_6, x_8\} .$$

As a result, backtracking iterations can be run in parallel at the nodes associated with each line and bus. If a standard trust region Newton method would be applied, the projected search would have to be computed on the same central node without a bound on the number iterations. Thus, the activity detection phase of TRAP allows one to reduce the number of global communications involved in the whole procedure. The results obtained via a basic augmented Lagrangian loop and a LANCELOT outer loop are presented in Tables 2.2 and 2.3 below. The data is taken from the archive <http://www.maths.ed.ac.uk/optenergy/LocalOpt/>. In all Tables of this Section, the first column corresponds to the index of the dual iteration, the second column to the number of iterations in the main loop of TRAP at the current outer step, the third column to the total number of sCG iterations at the current outer step, the fourth column to the level of KKT satisfaction obtained at each outer iteration, and the fifth column is the two-norm of the power flow equality constraints at a given dual iteration. To obtain the results presented in Tables 2.2 and 2.3, the regularisation parameter σ in the refinement stage 6 is set to $1 \cdot 10^{-10}$. For Table 2.2, the maximum number of iterations in the inner loop (TRAP) is fixed to 300 and the stopping tolerance on the level of satisfaction of the KKT conditions to $1 \cdot 10^{-5}$. For Table 2.3 (LANCELOT), the maximum number of inner iterations is set to 100 for the same stopping tolerance on the KKT conditions. In Algorithm 6, a block-diagonal preconditioner is applied. It is worth noting that the distributed implementation of Algorithm 6 is not affected by such a change. To obtain the results of Table 2.2, the initial penalty parameter ϱ is set to 10 and is multiplied by 30 at each outer iteration. In the LANCELOT loop, it is multiplied by 100. In the end, an objective value of 2733.55 up to feasibility $1.64 \cdot 10^{-8}$ of the power flow constraints is obtained, whereas the interior-point

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	79	388	$2.01 \cdot 10^{-7}$	0.530
2	2	40	$2.71 \cdot 10^{-10}$	0.530
3	300	2215	$2.39 \cdot 10^{-2}$	0.292
4	101	2190	$6.50 \cdot 10^{-4}$	$6.56 \cdot 10^{-3}$
5	123	2873	$2.10 \cdot 10^{-3}$	$5.02 \cdot 10^{-6}$
6	56	1194	$4.14 \cdot 10^{-2}$	$1.11 \cdot 10^{-10}$

Table 2.2: Results for the 9-bus AC-OPF (Fig. 2.8) using a standard augmented Lagrangian outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since the refinement stage was not preconditioned.

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	37	257	$7.29 \cdot 10^{-2}$	0.530
2	5	25	$1.01 \cdot 10^{-2}$	0.530
3	6	71	$3.23 \cdot 10^{-5}$	0.530
4	100	1330	$8.30 \cdot 10^{-3}$	$4.33 \cdot 10^{-2}$
5	100	1239	$1.80 \cdot 10^{-3}$	$2.53 \cdot 10^{-3}$
6	100	2269	$4.33 \cdot 10^{-2}$	$2.69 \cdot 10^{-5}$
7	64	1541	$3.2 \cdot 10^{-3}$	$1.64 \cdot 10^{-8}$

Table 2.3: Results for the 9-bus AC-OPF (Fig. 2.8) using a LANCELOT outer loop and TRAP as primal solver. Note that the cumulative number of CG iterations is relatively high, since no preconditioner was applied in the refinement step.

solver IPOPT, provided with the same primal-dual initial guess, yields an objective value of 2733.5 up to feasibility $2.23 \cdot 10^{-11}$. From Table 2.2, one can observe that a very tight KKT satisfaction can be obtained with TRAP. From the figures of Tables 2.2 and 2.3, one can extrapolate that LANCELOT would perform better in terms of computational time (6732 sCG iterations in total) than a basic augmented Lagrangian outer loop (8900 sCG iterations in total), yet with a worse satisfaction of the power flow constraints ($1.64 \cdot 10^{-8}$ against $1.11 \cdot 10^{-10}$). Finally, one should mention that over a set of hundred random initial guesses, TRAP was able to find a solution satisfying the power flow constraints up to $1 \cdot 10^{-7}$ in all cases, whereas IPOPT failed in approximately half of the test cases, yielding a point of local infeasibility.

2.2.3.2 Optimal AC power flow on distribution networks

Algorithm 5 is then applied to solve two AC-OPF problems in rectangular coordinates on distribution networks. Both 47-bus and 56-bus networks are taken from [64]. Our results are compared against the nonlinear interior-point solver IPOPT [152], which is not amenable to a fully distributed implementation, and the SOCP relaxation proposed by [64], which may be distributed (as convex) but fails in some cases, as shown next. It is worth noting that any distribution network is a tree, so a minimum colouring scheme consists of two colours, resulting in four parallel steps for the activity detection in TRAP.

2.2.3.2.1 On the 56-bus AC-OPF: An objective value of 233.9 is obtained with feasibility $8.00 \cdot 10^{-7}$, whereas the nonlinear solver IPOPT yields an objective value of 233.9 with feasibility $5.19 \cdot 10^{-7}$ for the same initial primal-dual guess.

In order to increase the efficiency of TRAP, following a standard recipe, we build a block-diagonal preconditioner from the hessian of the augmented Lagrangian by extracting block-diagonal elements corresponding to buses and lines. Thus, constructing and using the preconditioner can be done in parallel and does not affect the distributed nature of TRAP. In Fig. 2.9, the satisfaction of the KKT conditions for the bound constrained problem (2.79) is plotted for a preconditioned refinement phase and non-preconditioned one. One can conclude from Fig. 2.9 that preconditioning the

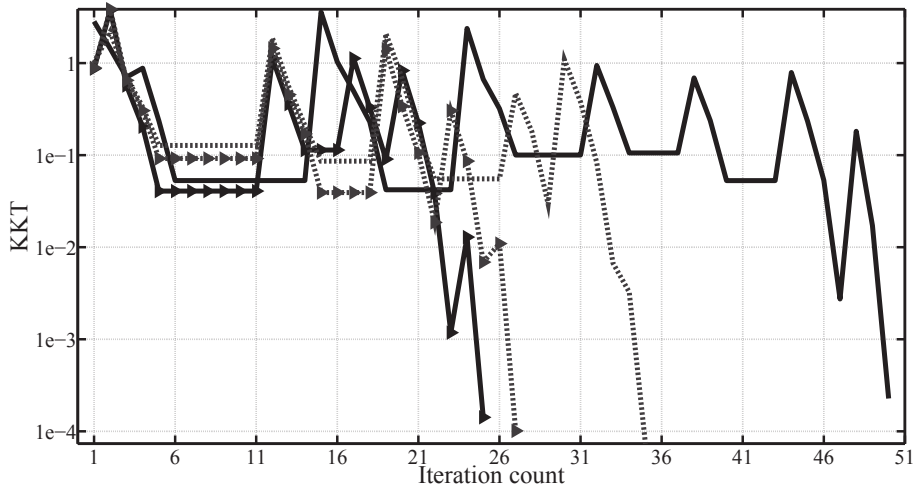


Figure 2.9: KKT satisfaction vs iteration count in the fourth LANCELOT subproblem formed on the AC-OPF with 56 buses. When using a centralised projected search as activity detector (dotted grey) and TRAP (full black). Curves obtained with a preconditioned sCG are highlighted with triangle markers.

refinement phase does not only affect the number of iterations of the sCG Algorithm 6 (Fig. 2.12), but also the performance of the main loop of TRAP. From a distributed perspective, it is very appealing, for it leads to a strong decrease in the overall number of global communications. Finally, from Fig. 2.9, it appears that TRAP and a centralised trust region method (with centralised projected search) are equivalent in terms of convergence speed. From Fig. 2.10, TRAP proves very efficient at

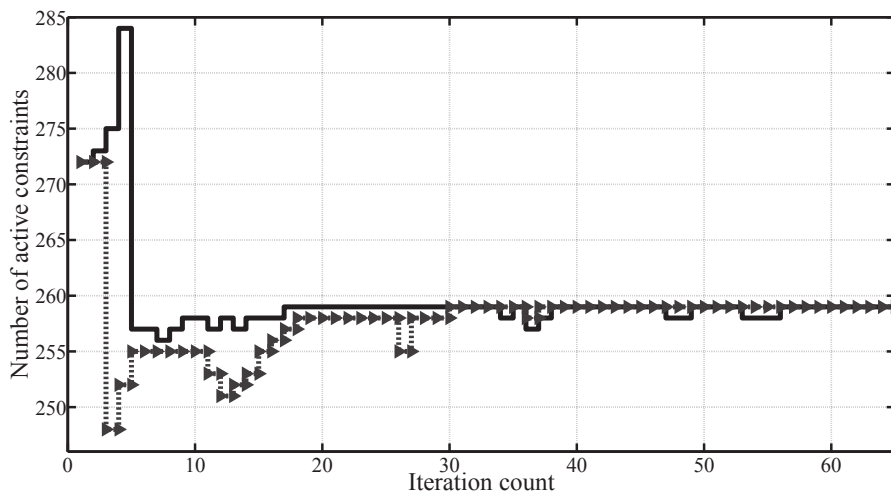


Figure 2.10: Active-set history in the first LANCELOT iteration for the 56-bus AC-OPF. Activity detection in TRAP: TRAP (full black), centralised projected search (dashed grey with triangles).

identifying the optimal active set in a few iterations (more than 10 constraints enter the active-set in the first four iterations and about 20 constraints are dropped in the following two iterations), which is a proof of concept for the analysis of paragraph 2.2.2. Alternating gradient projections appear to be as efficient as a projected search for identifying an optimal active-set, although the iterates travel on different faces, as shown in Fig. 2.10. In Fig. 2.11, the power flow constraints are evaluated after a run of TRAP on program (2.79). The dual variables and penalty coefficient are updated at each outer iteration. Overall, the coupling of TRAP with the augmented Lagrangian appears to be successful and provides similar performance to the coupling with a centralised trust region algorithm.

Tables 2.4 and 2.5 are obtained with an initial penalty coefficient $\rho = 10$ and a multiplicative coefficient of 20.

2.2.3.2.2 On the 47-bus AC-OPF: A generating unit was plugged at node 12 (bottom of the tree) and the load at the substation was decreased to 3 pu. On this modified problem, the SOCP relaxation provides a solution that does not satisfy the nonlinear equality constraints. An objective value of 502.3 is obtained with feasibility $2.57 \cdot 10^{-7}$ for both the AL loop (Tab. 2.6) and

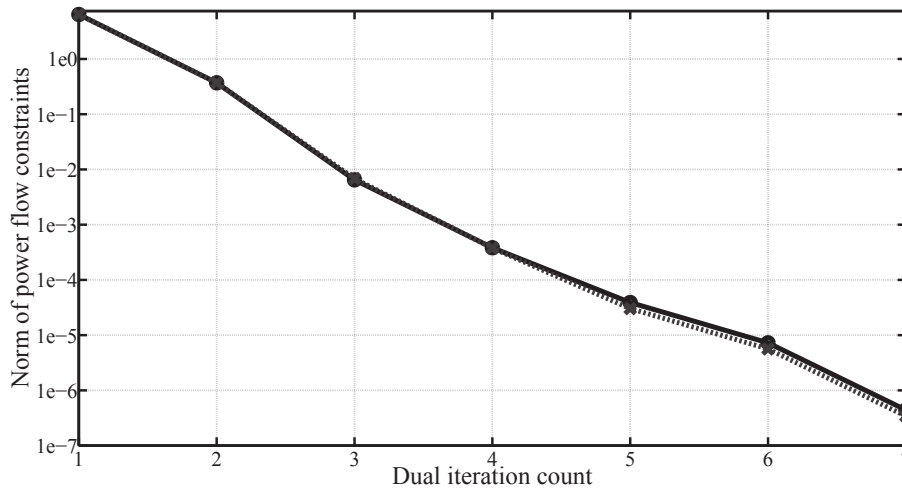


Figure 2.11: Norm of power flow constraints on the 56-bus network against dual iterations of a LANCELOT outer loop with TRAP as primal solver. Inner solver: TRAP (full black), centralised trust region method (dashed grey with cross markers).

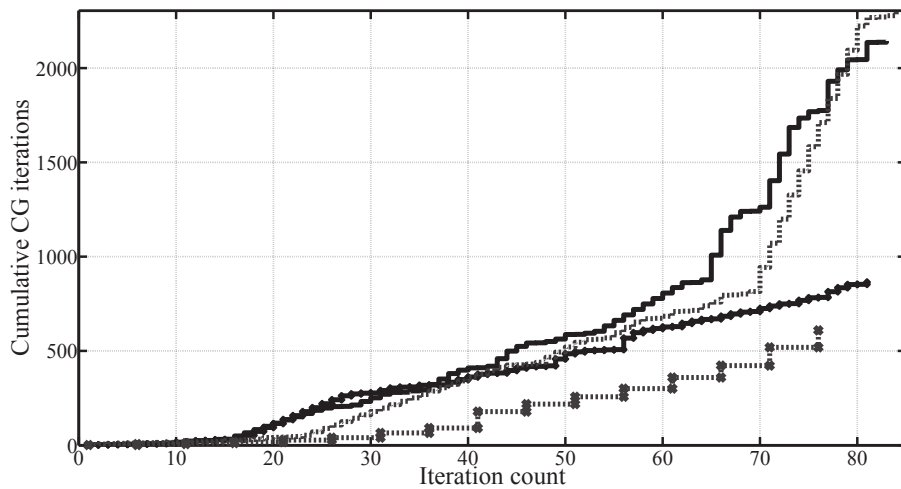


Figure 2.12: Cumulative sCG iterations vs iteration count in the first LANCELOT subproblem formed on the AC-OPF with 56 buses. Results obtained with TRAP as inner solver (full black), with a centralised trust region method (dashed grey). Results obtained with a preconditioned refinement stage are highlighted with cross markers.

the LANCELOT loop (Tab. 2.7). The SOCP relaxation returns an objective value of 265.75, but physically impossible, as the power flow constraints are not satisfied. The nonlinear solver IPOPT

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	122	1382	$8.45 \cdot 10^{-9}$	6.68
2	189	4486	$6.71 \cdot 10^{-9}$	$1.49 \cdot 10^{-1}$
3	139	11865	$9.87 \cdot 10^{-8}$	$8.79 \cdot 10^{-4}$
4	49	3958	$6.75 \cdot 10^{-6}$	$7.92 \cdot 10^{-6}$
5	9	936	$5.45 \cdot 10^{-7}$	$4.58 \cdot 10^{-9}$

Table 2.4: Results for the 56-bus AC-OPF of [64] using a (local) augmented Lagrangian outer loop with TRAP as primal solver.

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	100	924	$9.74 \cdot 10^{-2}$	6.42
2	133	3587	$2.40 \cdot 10^{-3}$	$3.60 \cdot 10^{-1}$
3	54	4531	$1.03 \cdot 10^{-4}$	$4.00 \cdot 10^{-3}$
4	10	858	$4.20 \cdot 10^{-6}$	$1.02 \cdot 10^{-3}$
5	42	3288	$4.37 \cdot 10^{-6}$	$2.32 \cdot 10^{-4}$
6	13	916	$1.82 \cdot 10^{-5}$	$4.35 \cdot 10^{-5}$
7	40	6878	$3.70 \cdot 10^{-7}$	$8.16 \cdot 10^{-6}$
8	6	420	$4.64 \cdot 10^{-6}$	$4.97 \cdot 10^{-7}$

Table 2.5: Results for the 56-bus AC-OPF of [64] using a LANCELOT outer loop with TRAP as primal solver.

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	275	3267	$1.33 \cdot 10^{-7}$	5.80
2	300	7901	$1.39 \cdot 10^{-1}$	$1.12 \cdot 10^{-1}$
3	180	18725	$2.13 \cdot 10^{-6}$	$9.47 \cdot 10^{-5}$
4	26	3765	$5.55 \cdot 10^{-8}$	$6.63 \cdot 10^{-9}$

Table 2.6: Results for the 47-bus AC-OPF of [64] using an augmented Lagrangian outer loop with TRAP as primal solver.

yields an objective value of 502.3 with feasibility $5.4 \cdot 10^{-8}$.

Outer iter. count	# inner it.	# cum. sCG per inner it.	Inner KKT	PF eq. constr.
1	180	1147	$8.64 \cdot 10^{-2}$	5.35
2	300	7128	2.23	$3.12 \cdot 10^{-1}$
3	215	11304	$4.65 \cdot 10^{-5}$	$2.97 \cdot 10^{-3}$
4	9	423	$6.05 \cdot 10^{-5}$	$3.28 \cdot 10^{-5}$
5	8	503	$1.11 \cdot 10^{-8}$	$7.90 \cdot 10^{-7}$
6	2	177	$4.64 \cdot 10^{-6}$	$4.03 \cdot 10^{-8}$

Table 2.7: Results for the 47-bus AC-OPF of [64] using a LANCELOT outer loop with TRAP as primal solver.

Chapter 3

A Parametric Decomposition Algorithm for Non-convex Programs

This chapter focuses on parametric nonconvex problems with separable objective, coupling constraints and separable constraints

$$\begin{aligned} & \underset{x_1, \dots, x_N}{\text{minimise}} \sum_{i=1}^N f_i(x_i) & (3.1) \\ & \text{s.t. } C(x_1, \dots, x_N, \sigma) = 0 \\ & \quad g_1(x_1, s_1) = 0, \dots, g_N(x_N, s_N) = 0 \\ & \quad x_1 \in \Omega_1, \dots, x_N \in \Omega_N, \end{aligned}$$

where σ, s_1, \dots, s_N are parameters. Our goal is to develop a decomposition algorithm to track local optima of NLP (3.1) as the parameters σ and s_1, \dots, s_N change. By tracking, we mean that the algorithm output should stay close to a critical point of (3.1) for different values of the parameters. In the literature, the most important family of optimality-tracking algorithms is the class of predictor-corrector methods [3]. When there are no inequality constraints, the KKT conditions of NLP (3.1) can be written as a nonlinear equation $F(w, s) = 0$, where w is a primal-dual unknown vector and s is a parameter. Given an approximation \bar{w} of a solution to the parametric equation for a parameter \bar{s} , the predictor-corrector scheme builds a new approximate solution \tilde{w} for a parameter $\tilde{s} \neq \bar{s}$ by solving the linearised equation

$$F(\bar{w}, \bar{s}) + \nabla_w F(\bar{w}, \bar{s})(\tilde{w} - \bar{w}) + \nabla_s F(\bar{w}, \bar{s})(\tilde{s} - \bar{s}) = 0,$$

which yields

$$\tilde{w} = \bar{w} - \underbrace{\nabla_w F(\bar{w}, \bar{s})^{-1} \nabla_s F(\bar{w}, \bar{s}) (\tilde{s} - \bar{s})}_{\text{Predictor}} - \underbrace{\nabla_w F(\bar{w}, \bar{s})^{-1} F(\bar{w}, \bar{s})}_{\text{Corrector}} .$$

The second term in the left hand side of the equality above corresponds to a tangential predictor of the solution to the nonlinear equation for parameter \tilde{s} . It can be obtained by linearisation of the solution around \bar{s} and the implicit function theorem. The third term is a corrector and is similar to a Newton step to solve the nonlinear equation. When inequality constraints are present, the inverse $\nabla_w F(\bar{w}, \bar{s})^{-1}$ can be interpreted as the resolution of one Newton subproblem with linearised constraints. If the parameter s appears linearly in the nonlinear equation $F(w, s) = 0$, which turns into $F(w) + Ts = 0$, one obtains

$$F(\bar{w}) + \nabla_w F(\bar{w}, \bar{s}) (\tilde{w} - \bar{w}) + T\tilde{s} = 0 ,$$

and then

$$\tilde{w} = \bar{w} - \nabla_w F(\bar{w}, \bar{s})^{-1} F(\bar{w}, \tilde{s}) .$$

This last equality corresponds to computing an approximate solution \tilde{w} for a parameter \tilde{s} by applying one Newton iteration initialised at the suboptimal solution \bar{w} . In the case of a convex objective and nonlinear equality constraints, such a predictor-corrector scheme with a positive semidefinite hessian approximation is proposed and analysed in [148]. In the particular case of NMPC problems with least-squares tracking cost, the hessian approximation can be computed by means of a Gauss-Newton approximation, which guarantees positive semi-definiteness [119]. However, for a general cost, such as an economic objective for instance, the Gauss-Newton approximation is not as effective. One could use the exact hessian in a Newton subproblem, but the resulting quadratic problem would be nonconvex, and thus the approach would not be computationally efficient, despite recent progress in this direction [128]. However, the approach of [128], which makes use of a mirrored version of the exact hessian in order to ensure positive definiteness, does not have a solid theoretical foundation, despite its good performance in some practical cases. The purpose of this chapter is to propose and analyse an optimality-tracking algorithm, which makes use of exact second order information and comes with stability guarantees. This is also relevant in the context of decomposition methods and distributed optimisation, as the sparsity pattern of the exact hessian of the Lagrangian reflects the coupling topology. Hence, distributed linear algebra techniques can be readily applied.

A practical implementation of distributed optimisation techniques may be cumbersome depending on the target computational platform. The main difficulty stems from the fact that agents

need to communicate at every iteration of the algorithm. Thus, one needs to develop mechanisms to coordinate and synchronise communications. Some of these aspects may be more stringent depending on the set-up. For instance, if we are to implement a distributed optimisation method on a multi-core chip, such as a GPU, communication between cores is more reliable and comes at a higher rate than when the hardware and memory are spread around different locations, as it is the case clusters and distributed embedded optimisation for instance. This aspect is even more stringent if a distributed algorithm is to be applied in a real-time setting, where the computation time may be constrained very tightly, due to communication delays. Cumulating both the real-time and distributed aspects puts very strong constraints on an optimality-tracking algorithm. Therefore, the effect of a limited communication rate on the tracking performance is to be properly investigated.

3.1 A Parametric Augmented Lagrangian Algorithm

3.1.1 Problem formulation and algorithm description

The following class of parametric NLPs with separable cost, partially separable equality constraints and separable inequality constraints is considered

$$\begin{aligned}
 & \underset{x_1, \dots, x_N}{\text{minimise}} \quad J(x) := \sum_{i=1}^N J_i(x_i) & (3.2) \\
 & \text{s.t.} \quad Q_c(x_1, \dots, x_N) = 0, \\
 & \quad \quad g_i(x_i) + T_i s_k = 0, \\
 & \quad \quad x_i \in \Omega_i, \quad i \in \{1, \dots, N\},
 \end{aligned}$$

where $x := (x_1^\top, \dots, x_N^\top)^\top \in \mathbb{R}^n$, with $n = \sum_{i=1}^N n_i \geq 2$ and $x_i \in \mathbb{R}^{n_i}$. The vectors x_i model different agents, while the function $Q_c : \mathbb{R}^n \rightarrow \mathbb{R}^{m_c}$ represents *constraint couplings*.

Remark 3.1. For clarity, the definition of NLP (3.2) is restricted to constraint couplings. However, cost couplings can be addressed by the approach described in the sequel.

The functions $J_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}$ and $g_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^{q_i}$ are *individual cost and constraint functionals* at agent $i \in \{1, \dots, N\}$. In an NMPC context, the nonlinear equality constraint involving g_i models the dynamics of agent i over a prediction horizon. The vector s_k is a parameter, which lies within a set $\mathbb{S} \subseteq \mathbb{R}^p$ and varies with k .

Remark 3.2. When it comes to NMPC, the parameter s_k stands for a state estimate or a reference trajectory and the index k represents a time instant.

The matrices $T_i \in \mathbb{R}^{q_i \times p}$ are constant. The linear dependence of the local equality constraints in the parameter s_k is not restrictive, as extra variables can be introduced in order to obtain this formulation. For all $s \in \mathbb{S}$, we define the equality constraints functional $G : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with $m = m_c + \sum_{i=1}^N q_i$, given $x \in \mathbb{R}^n$ as follows

$$G(x, s) := (Q_c(x)^\top, (g_1(x) + T_1 s)^\top, \dots, (g_N(x) + T_N s)^\top)^\top.$$

For all $i \in \{1, \dots, N\}$, the constraint sets Ω_i are assumed to be bounded boxes. Note that such an assumption is not restrictive, as slack variables can always be introduced. Critical points of NLP (3.2) are denoted by w_k^* or $w^*(s_k)$ without distinction.

Remark 3.3. It is worth noting that the problem formulation (3.2) does not encompass standard NMPC programs, which typically involve terminal weights and constraints that group together

all sub-systems states. Thus, in order to obtain a separable objective subject to partially separable constraints, one may resort to the NMPC design proposed by [77], which does not involve any terminal conditions, but is based on a sufficiently long prediction horizon to ensure closed-loop stability under the optimal NMPC control law. Another possibility is to extend the distributed invariance design procedure of [37] to the nonlinear case via a standard linearisation and level-set shrinking argument. One could also design box-shaped terminal sets and separable terminal quadratic penalties using the approach outlined in [102].

3.1.1.1 A descent-based scheme for parametric nonconvex programs

For every index k , a critical point of the parametric NLP (3.2) is computed approximately. The key idea is to track parameter-dependent local optima of program (3.2) by computing saddle points of the parametric augmented Lagrangian

$$L_\varrho(x, \mu, s) := J(x) + \left(\mu + \frac{\varrho}{2} G(x, s) \right)^\top G(x, s) \quad , \quad (3.3)$$

subject to $x \in \Omega$, where $\Omega := \Omega_1 \times \Omega_2 \times \dots \times \Omega_N$ and $\mu := (\mu_c^\top, \mu_1^\top, \dots, \mu_N^\top)^\top \in \mathbb{R}^{m+q}$, with $q := \sum_{i=1}^N q_i$, is a dual variable associated with the equality constraints

$$Q_c(x) = 0, \quad g_1(x_1) + T_1 s_k = 0, \dots, \quad g_N(x_N) + T_N s_k = 0$$

respectively. The penalty $\varrho > 0$ remains constant for every index k . In the remainder of this chapter, sub-optimality of a variable is highlighted with a $\bar{\cdot}$, and criticality with a \cdot^* . Under appropriate constraint qualifications, an approximate KKT point $(\bar{x}(s_{k+1})^\top, \bar{\mu}(s_{k+1})^\top)^\top$ of (3.2) is constructed by applying a descent method to the parametric augmented Lagrangian function (3.3) at s_{k+1} after initialising the primal iterations at $\bar{x}(s_k)$, and updating the dual variable in a first-order fashion, as described in Algorithm 7 below.

Algorithm 7 Optimality-tracking descent-based algorithm

Input: Suboptimal primal-dual solution $(\bar{x}(s_k)^\top, \bar{\mu}(s_k)^\top)^\top$, parameter s_{k+1} , augmented Lagrangian function $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$

Descent phase: Apply M iterations of a descent method (Algorithm 8, 9, 10 or 11) initialised at $\bar{x}(s_k)$ to minimise the augmented Lagrangian function $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ and obtain a suboptimal primal iterate x^M

$\bar{x}(s_{k+1}) \leftarrow x^M$

Dual update: $\bar{\mu}(s_{k+1}) \leftarrow \bar{\mu}(s_k) + \varrho G(\bar{x}(s_{k+1}), s_{k+1})$

By descent method, we mean that every iteration guarantees that the objective decreases, which is the augmented Lagrangian in the case of Algorithm 7. There exists several good candidates for

a descent method applicable to minimise the augmented Lagrangian. Next, we describe two centralised strategies (Algorithms 8 and 9), namely a projected gradient algorithm and a trust region method, as well as two distributed strategies (Algorithms 10 and 11), which are an alternating projected gradient algorithm and the TRAP method described in Chapter 2.

Algorithm 7 can be regarded as a way to resolve some issues raised by the optimality-tracking scheme of [157], which is based on the following augmented Lagrangian

$$\underset{x \geq 0}{\text{minimise}} \quad L_\varrho(x, \bar{\mu}_k, s_{k+1}) := J(x) + \left\langle \bar{\mu}_k + \frac{\varrho}{2} G(x, s_{k+1}), G(x, s_{k+1}) \right\rangle ,$$

associated with the problem formulation

$$\begin{aligned} & \underset{x \geq 0}{\text{minimise}} \quad J(x) \\ & \text{s.t.} \quad G(x, s_{k+1}) = 0 . \end{aligned}$$

In [157], a critical point of the following quadratic programming problem is to be computed for every index k ,

$$\underset{x \geq 0}{\text{minimise}} \quad \langle \nabla_x L_\varrho(\bar{x}_k, \bar{\mu}_k, s_{k+1}), x - \bar{x}_k \rangle + \frac{1}{2} \langle x - \bar{x}_k, \nabla_{xx}^2 L_\varrho(\bar{x}_k, \bar{\mu}_k, s_{k+1})(x - \bar{x}_k) \rangle , \quad (3.4)$$

via a Projected Successive OverRelaxation (PSOR) [111], which provably converges linearly to a critical point of (3.4) when the hessian matrix

$$\nabla_{xx}^2 L_\varrho(\bar{x}_k, \bar{\mu}_k, s_{k+1}) \quad (3.5)$$

is positive definite (Corollary 2.2 in [111]). When matrix (3.5) is not positive definite, it can still be proven that all limit points of the sequence generated by PSOR are critical points of (3.4), but existence of a limit point is not guaranteed (Theorem 2.1 in [111]). However, from the second-order optimality conditions and Lemma 1.4, one can only deduce that the matrix

$$\nabla_{xx} L_\varrho(x^*(\bar{\mu}_k, s_{k+1}), \bar{\mu}_k, s_{k+1}) + \varrho (Z^*)^\top Z^*$$

is positive definite, where the rows of the matrix Z^* are the coordinate vectors corresponding to the active nonnegativity constraints at the critical point $x^*(\bar{\mu}_k, s_{k+1})$. Thus, the matrix (3.5) is not guaranteed to be positive definite as long as the optimal active-set has not been identified, and strong guarantees on the convergence of the PSOR iterations are lost. Moreover, interior-point or active-set methods tailored to convex quadratic programs cannot be applied in the framework of [157]. Moreover, the convergence of PSOR is likely to be slow without an appropriate tuning,

which may be hard to obtain.

We now state the four descent schemes applied to the parametric augmented Lagrangian in Algorithm 7. We start with the two centralised methods and then move to the two distributed algorithms.

3.1.1.2 Projected gradient on the parametric augmented Lagrangian

The parametric augmented Lagrangian is minimised by means of projected gradient steps. It is important to note that Algorithm 8 is initialised at the previous suboptimal primal point $\bar{x}(s_k)$ and stops after $M \geq 1$ iterations. The step-size is adjusted at every iteration by backtracking. The stopping criterion, which is applied to stop the backtracking loop is designed to ensure sufficient decrease of the parametric augmented Lagrangian at every iteration.

Algorithm 8 Projected gradient on the parametric augmented Lagrangian

Constants: Suboptimal primal variable $\bar{x}(s_k)$ and objective function $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$.
Parameters: Initial curvature estimate $c^{(0)} > 0$, regularisation parameter $r > 0$ and multiplicative coefficient $\beta > 1$.
Warm-start: $x \leftarrow \bar{x}(s_k)$
for $l = 1, \dots, M$ **do**
 Backtracking:
 $c \leftarrow c^{(0)}$
 while $L_\rho(\tilde{x}, \bar{\mu}_k, s_{k+1}) > L_\rho(x, \bar{\mu}_k, s_{k+1}) + \langle \nabla L_\rho(x, \bar{\mu}_k, s_{k+1}), \tilde{x} - x \rangle + \frac{c-r}{2} \|\tilde{x} - x\|_2^2$ **do**
 $\tilde{x} \leftarrow P_\Omega\left(x - \frac{1}{c} \nabla L_\rho(x, \bar{\mu}_k, s_{k+1})\right)$
 $c \leftarrow \beta c$
 end while
 $x \leftarrow \tilde{x}$
end for
Output: $x^M = x$

The backtracking loop starts with a curvature estimate $c^{(0)}$. If the gradient $\nabla L_\rho(\cdot, \bar{\mu}_k, s_{k+1})$ is Lipschitz continuous over Ω , using the descent Lemma, it is easy to see that the backtracking loop stops after a finite number of iterations, as shown in Chapter 2 in a similar context.

3.1.1.3 Trust region methods on the parametric augmented Lagrangian

It is well-known that first-order methods such as the projected gradient (Algorithm 8) can be very ineffective when applied to ill-conditioned problems [119]. Moreover, their local convergence rate is at best linear and convergence can be arbitrarily slow. A natural way to accelerate convergence is to use the gradient projection to identify an active set and solve a Newton subproblem inexactly

on the current subspace [33, 116]. This is justified, since the gradient projection algorithm identifies an optimal active-set after a finite number of iterations [28]. We use this activity detection mechanism in a trust region setting [36]. A standard trust region method applied to the parametric augmented Lagrangian $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ is described in Algorithm 9 below.

Algorithm 9 Trust region algorithm on the parametric augmented Lagrangian

- 1: **Constants:** Initial trust region radius Δ , update coefficients σ_1, σ_2 and σ_3 such that $0 < \sigma_1 < \sigma_2 < 1 < \sigma_3$, test ratios η_1 and η_2 such that $0 < \eta_1 < \eta_2 < 1$, coefficients $\gamma_1 \in]0, 1[$ and $\gamma_2 > 0$ and regularisation coefficient r .
- 2: **Input:** Suboptimal primal variable $\bar{x}(s_k)$ and objective function $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$.
- 3: **Warm-start:** $x \leftarrow \bar{x}(s_k)$
- 4: **for** $l = 1, \dots, M$ **do**
- 5: **Active set identification:**
- 6: Compute the Cauchy point $z \leftarrow P_\Omega(x - \alpha \nabla L_\rho(x, \bar{\mu}_k, s_{k+1}))$ according to requirements (3.7), (3.9) and (3.10).
- 7: **Refinement:**
- 8: Find $y \in \Omega$ by approximately solving via sCG iterations initialised at z

$$\begin{aligned} & \underset{y \in \Omega}{\text{minimise}} \quad m_\rho(y, \bar{\mu}_k, s_{k+1}) + \frac{r}{2} \|y - z\|_2^2 \\ & \text{s.t.} \quad \|y - x\|_\infty \leq \gamma_2 \Delta \\ & \quad \mathcal{A}_\Omega(z) \subseteq \mathcal{A}_\Omega(y) \end{aligned}$$

to ensure $m_\rho(x, \bar{\mu}_k, s_{k+1}) - m_\rho(y, \bar{\mu}_k, s_{k+1}) \geq \gamma_1 (m_\rho(x, \bar{\mu}_k, s_{k+1}) - m_\rho(z, \bar{\mu}_k, s_{k+1}))$.

- 9: **Trust-region update:**
 - 10: $\rho \leftarrow \frac{L_\rho(x, \bar{\mu}_k, s_{k+1}) - L_\rho(y, \bar{\mu}_k, s_{k+1})}{m_\rho(x, \bar{\mu}_k, s_{k+1}) - m_\rho(y, \bar{\mu}_k, s_{k+1})}$
 - 11: **if** $\rho < \eta_1$ **then** ▷ Not successful
 - 12: *(Do not update x)*
 - 13: Pick Δ within $[\sigma_1 \Delta, \sigma_2 \Delta]$
 - 14: **else if** $\rho \in [\eta_1, \eta_2]$ **then** ▷ Successful
 - 15: $x \leftarrow y$
 - 16: Pick Δ within $[\sigma_1 \Delta, \sigma_3 \Delta]$
 - 17: Update objective gradient $\nabla L_\rho(x, \bar{\mu}_k, s_{k+1})$ and model hessian $B_\rho(x, \bar{\mu}_k, s_{k+1})$
 - 18: **else** ▷ Very successful
 - 19: $x \leftarrow y$
 - 20: Pick Δ within $[\Delta, \sigma_3 \Delta]$
 - 21: Update objective gradient $\nabla L_\rho(x, \bar{\mu}_k, s_{k+1})$ and model hessian $B_\rho(x, \bar{\mu}_k, s_{k+1})$
 - 22: **end if**
 - 23: **end for**
 - 24: **Output:** $x^M = x$
-

Given the dual variable $\bar{\mu}_k$ and the parameter s_{k+1} , at every iteration l of Algorithm 9, a model

$m_\varrho(\cdot, \bar{\mu}_k, s_{k+1})$ of the objective $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1})$ is constructed around the current iterate x as follows

$$m_\varrho(x', \bar{\mu}_k, s_{k+1}) := L_\varrho(x, \bar{\mu}_k, s_{k+1}) + \langle \nabla L_\varrho(x, \bar{\mu}_k, s_{k+1}), x' - x \rangle + \frac{1}{2} \langle x' - x, B_\varrho(x, \bar{\mu}_k, s_{k+1})(x' - x) \rangle, \quad (3.6)$$

where $x' \in \mathbb{R}^n$ and $B_\varrho(x, \bar{\mu}_k, s_{k+1}) \in \mathbb{R}^{n \times n}$ is a symmetric matrix. In Algorithm 9, the first phase consists in computing a gradient projection step yielding the so-called Cauchy point z . For the usual convergence guarantees to hold, one needs the step-size α to be sufficiently small to ensure sufficient decrease and containment in a scaled trust region

$$\begin{cases} m_\varrho(z, \bar{\mu}_k, s_{k+1}) \leq m_\varrho(x, \bar{\mu}_k, s_{k+1}) + \nu_0 \langle \nabla L_\varrho(x, \bar{\mu}_k, s_{k+1}), z - x \rangle \\ \|z - x\|_\infty \leq \nu_2 \Delta, \end{cases} \quad (3.7)$$

where $\nu_0 \in]0, 1[$ and $\nu_2 > 0$. One also has to make sure that the step-size α does not become too small. Therefore, it should as well satisfy

$$\alpha \in [\nu_4, \nu_5] \text{ or } \alpha \in [\nu_3 \tilde{\alpha}, \nu_5], \quad (3.8)$$

where $\nu_3, \nu_4, \nu_5 > 0$ and $\tilde{\alpha} > 0$ is such that

$$m_\varrho(z(\tilde{\alpha}), \bar{\mu}_k, s_{k+1}) > m_\varrho(x, \bar{\mu}_k, s_{k+1}) + \nu_0 \langle \nabla L_\varrho(x, \bar{\mu}_k, s_{k+1}), z(\tilde{\alpha}) - x \rangle \quad (3.9)$$

or

$$\|z(\tilde{\alpha}) - x\|_\infty > \nu_1 \Delta, \quad (3.10)$$

where $0 < \nu_1 < \nu_2$.

Compared to standard trust region methods [36], we slightly modify the refinement step by adding a proximal regularisation term to the model, which is

$$\frac{r}{2} \|y - z\|_2^2.$$

This quadratic regularisation actually plays a significant role in the analysis of Section 3.2. The refinement phase is typically computed by means of safeguarded Conjugate Gradient (sCG) iterations [145], which are initialised at the Cauchy point z , and along which the model function

$m_\varrho(\cdot, \bar{\mu}_k, s_{k+1})$ is strictly decreasing (Theorem 2.1 in [145]), so that the inequality

$$m_\varrho(x, \bar{\mu}_k, s_{k+1}) - m_\varrho(y, \bar{\mu}_k, s_{k+1}) \geq \gamma_1 (m_\varrho(x, \bar{\mu}_k, s_{k+1}) - m_\varrho(z, \bar{\mu}_k, s_{k+1}))$$

is always satisfied whenever the sCG iterations are aborted.

The projected gradient and trust region algorithms described above are not suitable for distributed optimisation, as they both rely on centralised backtracking procedures. However, they are still relevant for centralised parametric optimisation and can be regarded as background to the methods shown next. In the next two paragraphs, we describe two decomposition strategies for solving the parametric augmented Lagrangian problem.

3.1.1.4 Alternating projected gradients in parametric augmented Lagrangian

This is essentially Algorithm 2 of Chapter 2 applied to the parametric augmented Lagrangian. In order to make the distributed nature of the algorithm more straightforward, we add some assumptions on the coupling function Q_c .

Assumption 3.1 (Sparse coupling [17]). *The subvariables x_1, \dots, x_N can be re-ordered and grouped together in such a way that a Gauss-Seidel sweep on the function $\|Q_c\|_2^2$ can be performed in P steps among which all subvariables are updated in parallel, where $P \ll N$. The re-ordered and grouped subvariables are denoted by χ_1, \dots, χ_P , so that the re-arranged vector χ is defined by $\chi := (\chi_1^\top, \dots, \chi_P^\top)^\top$. More precisely, for $i \in \{1, \dots, P\}$, we have*

$$\chi_i = \left(x_{i_1}^\top, \dots, x_{i_{p_i}}^\top \right)^\top,$$

where $p_i \geq 1$ is the number of decoupled subvariables in group i . In this paragraph, it is assumed that NLP (3.2) has been re-arranged accordingly and that

$$\Omega = \tilde{\Omega}_1 \times \dots \times \tilde{\Omega}_P,$$

where $\tilde{\Omega}_i = \Omega_{i_1} \times \dots \times \Omega_{i_{p_i}}$.

Remark 3.4. *Assumption 3.1 is standard in distributed computations [17]. It encompasses a large number of practical problems of interest. For consensus problems, in which coupling constraints $x_1 - x_i = 0$ appear for $i \in \{2, \dots, N\}$, one has $P = 2$ updates, corresponding to the update of $\chi_1 = x_1$ followed by the parallel updates of $\chi_2 = (x_2^\top, \dots, x_N^\top)$. When the coupling graph is a tree, such as in the case of a distribution network, one also obtains $P = 2$. Our approach is likely to be more efficient when P is small relative to N .*

Remark 3.5. Next, we write $L_\varrho(x, \bar{\mu}_k, s_{k+1})$ or $L_\varrho(\chi, \bar{\mu}_k, s_{k+1})$ without distinction.

We define the blockwise parametric augmented Lagrangian function at group $i \in \{1, \dots, P\}$, where $P \ll N$ by Assumption 3.1,

$$L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i)} := L_\varrho(\chi_1, \dots, \chi_{i-1}, \cdot, \chi_{i+1}, \dots, \chi_P, \bar{\mu}_k, s_{k+1}) \quad (3.11)$$

and a quadratic model at χ_i , encompassing all subvariables of group i , given a curvature coefficient $c_i > 0$,

$$q(\cdot; \chi_i, c_i) := L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i)}(\chi_i) + \left\langle \nabla L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i)}(\chi_i), \cdot - \chi_i \right\rangle + \frac{c_i}{2} \|\cdot - \chi_i\|_2^2.$$

Given an iteration index $l \geq 1$, we define

$$L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i,l)} := L_\varrho\left(\chi_1^{(l+1)}, \dots, \chi_{i-1}^{(l+1)}, \cdot, \chi_{i+1}^{(l)}, \dots, \chi_P^{(l)}, \bar{\mu}_k, s_{k+1}\right).$$

For every group of agents indexed by $i \in \{1, \dots, P\}$, a regularisation coefficient $r_i > 0$ is chosen. In practice, such a coefficient should be taken as small as possible. Algorithm 10 below splits the parametric augmented Lagrangian problem.

Algorithm 10 Alternating projected gradient on the parametric augmented Lagrangian

- 1: **Constants:** Regularisation coefficients $\{r_i\}_{i=1}^P$, initial curvature coefficients $\{c_i^{(0)}\}_{i=1}^P$, backtracking coefficient $\beta > 1$
 - 2: **Input:** Suboptimal primal subvariables $\bar{\chi}_1(s_k), \dots, \bar{\chi}_P(s_k)$, objective $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$
 - 3: **Warm-start:** $\chi_1 \leftarrow \bar{\chi}_1(s_k), \dots, \chi_P \leftarrow \bar{\chi}_P(s_k)$
 - 4: **for** $l = 1, \dots, M$ **do**
 - 5: **Loop over groups:**
 - 6: **for** $i = 1, \dots, P$ **do**
 - 7: **Backtracking at group i :** ▷ In parallel among decoupled subvariables $x_{i_1}, \dots, x_{i_{p_i}}$
 - 8: $\tilde{\chi}_i \leftarrow \chi_i, c_i \leftarrow c_i^{(0)}$
 - 9: **while** $L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i,l)}(\tilde{\chi}_i) + \frac{r_i}{2} \|\tilde{\chi}_i - \chi_i\|_2^2 > q(\tilde{\chi}_i; \chi_i, c_i)$ **do**
 - 10: $\tilde{\chi}_i \leftarrow P_{\tilde{\Omega}_i} \left(\chi_i - \frac{1}{c_i} \nabla L_{\varrho, \bar{\mu}_k, s_{k+1}}^{(i,l)}(\chi_i) \right)$
 - 11: $c_i \leftarrow \beta \cdot c_i$
 - 12: **end while**
 - 13: $\chi_i \leftarrow \tilde{\chi}_i$
 - 14: **end for**
 - 15: **end for**
 - 16: **Output:** $\chi_1^M \leftarrow \chi_1, \dots, \chi_P^M \leftarrow \chi_P$
-

Each step of the alternating minimisation among the P groups of decoupled subvariables consists of backtracking projected gradient steps in parallel for each of the P groups. Similarly to Chapter 2, one can show that the backtracking loop stops after a finite number of recursions under a blockwise Lipschitz continuity assumption on the gradient of the parametric augmented Lagrangian.

Remark 3.6. *Incremental approaches are broadly applied in NMPC, for fully solving an NLP takes a significant amount of computational resources and may result in unacceptable time delays. Yet, existing incremental NMPC strategies [46, 159] are based on Newton predictor-corrector steps, which require factorisation of a KKT system. This is a computationally demanding task for large-scale systems that cannot be readily carried out in a distributed context. Therefore, Algorithm 10 can be interpreted as a distributed incremental improvement technique for NMPC.*

Remark 3.7. *Note that the active-set at $z^*(s_{k+1})$ may be different from the active-set at $z^*(s_k)$. Hence, Algorithm 10 should be able to detect active-set changes quickly. This is the role of the alternating gradient projections. It is well-known that a standard gradient projection method allows for fast activity detection [28]. Moreover, it has been shown in Chapter 2 that alternating gradient projections enjoy the same desirable property.*

3.1.1.5 TRAP on the parametric augmented Lagrangian (pTRAP)

The TRAP algorithm that was presented in Section 2.2.1 of Chapter 2. As described in Algorithm 11 below, it can be applied to the parametric augmented Lagrangian.

Algorithm 11 pTRAP

- 1: **Constants:** Initial trust region radius Δ , update constants σ_1, σ_2 and σ_3 such that $0 < \sigma_1 < \sigma_2 < 1 < \sigma_3$, test ratios η_1 and η_2 such that $0 < \eta_1 < \eta_2 < 1$, coefficients $\gamma_1 \in]0, 1[$ and $\gamma_2 > 0$
- 2: **Input:** Suboptimal primal subvariables $\bar{\chi}_1(s_k), \dots, \bar{\chi}_P(s_k)$, objective $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$
- 3: **Warm-start:** $\chi_1 \leftarrow \bar{\chi}_1(s_k), \dots, \chi_P \leftarrow \bar{\chi}_P(s_k)$
- 4: **for** $l = 1, \dots, M$ **do**
- 5: **Distributed activity detection (alternating gradient projections):**
- 6: **for** $i = 1 \dots, P$ **do**
- 7: $z_i \leftarrow P_{\Omega_i}(\chi_i - \alpha_i \nabla_i m_\rho(z_{[1, i-1]}, \chi_i, \chi_{[i+1, P]}, \bar{\mu}_k, s_{k+1}))$, \triangleright In parallel in group i
- 8: where α_i is computed according to requirements (2.55), (2.56) and (2.57).
- 9: **end for**
- 10: **Distributed refinement (Algorithm 6):**
- 11: Find $y_1 \in \Omega_1, \dots, y_P \in \Omega_P$ by applying distributed sCG iterations initialised at the Cauchy
- 12: points z_1, \dots, z_P to the problem

$$\underset{y_1 \in \Omega_1, \dots, y_P \in \Omega_P}{\text{minimise}} \quad m_\rho(y, \bar{\mu}_k, s_{k+1}) + \frac{r}{2} \sum_{i=1}^P \|y_i - z_i\|_2^2$$

$$\text{s.t. } \forall i \in \{1, \dots, P\}, \|y_i - \chi_i\|_\infty \leq \gamma_2 \Delta$$

$$\forall i \in \{1, \dots, P\}, \mathcal{A}_\Omega(z_i) \subseteq \mathcal{A}_\Omega(y_i) \quad ,$$

- 13: guaranteeing $m_\rho(\chi, \bar{\mu}_k, s_{k+1}) - m_\rho(y, \bar{\mu}_k, s_{k+1}) \geq \gamma_1 (m_\rho(\chi, \bar{\mu}_k, s_{k+1}) - m_\rho(z, \bar{\mu}_k, s_{k+1}))$
- 14: **Trust region update:**
- 15: $\rho \leftarrow \frac{L_\rho(\chi, \bar{\mu}_k, s_{k+1}) - L_\rho(y, \bar{\mu}_k, s_{k+1})}{m_\rho(\chi, \bar{\mu}_k, s_{k+1}) - m_\rho(y, \bar{\mu}_k, s_{k+1})}$
- 16: **if** $\rho < \eta_1$ **then** \triangleright Not successful
- 17: (Do not update χ)
- 18: Pick Δ within $[\sigma_1 \Delta, \sigma_2 \Delta]$
- 19: **else if** $\rho \in [\eta_1, \eta_2]$ **then** \triangleright Successful
- 20: $\chi_1 \leftarrow y_1, \dots, \chi_P \leftarrow y_P$
- 21: Pick Δ within $[\sigma_1 \Delta, \sigma_3 \Delta]$
- 22: Update objective gradient $\nabla L_\rho(\chi, \bar{\mu}_k, s_{k+1})$ and model hessian $B_\rho(\chi, \bar{\mu}_k, s_{k+1})$.
- 23: **else** \triangleright Very successful
- 24: $\chi_1 \leftarrow y_1, \dots, \chi_P \leftarrow y_P$
- 25: Pick Δ within $[\Delta, \sigma_3 \Delta]$
- 26: Update objective gradient $\nabla L_\rho(\chi, \bar{\mu}_k, s_{k+1})$ and model hessian $B_\rho(\chi, \bar{\mu}_k, s_{k+1})$
- 27: **end if**
- 28: **end for**
- 29: **Output:** $\chi_1^M \leftarrow \chi_1, \dots, \chi_P^M \leftarrow \chi_P$

After M trust region iterations, which may be successful or not, the process is aborted. The rationale for using pTRAP instead of Algorithm 10 is that better performance in terms of stability of the tracking scheme can be expected, as justified by the analysis of Section 3.2.

The main features of each algorithm presented above are summarised in Tab. 3.1 below.

	First-order method	Active-set method	Distributed strategy	Fast local convergence	Convergence analysis
Algorithm 8	✓				Paragraph 3.2.1.1, Theorem 3.2
Algorithm 9		✓		✓	Paragraph 3.2.1.3, Theorem 3.6
Algorithm 10	✓		✓		Paragraph 3.2.1.2, Theorem 3.3
Algorithm 11		✓	✓	✓	Paragraph 3.2.1.4, Theorem 3.8

Table 3.1: Comparison of Algorithm 8, 9, 10 and 11.

3.2 Local Analysis and Contraction Properties

In this section, we investigate the stability properties of Algorithm 7. By stability, we mean that the distance of the suboptimal primal-dual point $\bar{w}(s_k)$ yielded by Algorithm 7 to a KKT point $w^*(s_k)$ remains at least bounded as the parameter s_k varies with k . This is not automatically guaranteed, since Algorithm 7 consists in a fixed number of iterations M of a descent method, which outputs a suboptimal primal solution and a single dual update is performed. Therefore, Algorithm 7 is both primal and dual suboptimal. However, the iterative process is initialised at a primal-dual optimal warm-start, which may not be too far from a KKT point under some conditions that are to be characterised next. We expect that some conditions on the parameter difference $s_{k+1} - s_k$, the number of primal iterations M and the penalty parameter can be derived to ensure stability of the optimality-tracking Algorithm 7. Algorithms 8, 9, 10 and 11 have in common that they enforce a decrease of the parametric augmented Lagrangian at every iteration. In order to investigate the local behaviour of Algorithm 7, we first derive local convergence rates for Algorithms 8, 9, 10 and 11.

In the analysis that follows, we prove that the sequence of iterates generated by a descent method such as Algorithm 8, 9, 10 or 11, converges to a critical point of the parametric augmented Lagrangian. In the nonconvex case, only subsequence convergence can generally be proven for Algorithm 8 and 9 under weak assumptions, and the convergence properties of alternating minimisation techniques such as the ones of Algorithm 10 and 11 are obscure. To circumvent the problem, we resort to the results and tools introduced by [10, 11, 12], among which the cornerstone is the *Kurdyka-Lojasiewicz (KL) inequality* [11]. The role of the KL inequality in optimisation is known since the work of [2], who used it to obtain strong convergence results for the iterates of descent methods on analytic cost functions. It has been extended to more general descent schemes by [12]. In this section, we show that this property along with the results of [10] can be employed to derive novel converge rates for trust region methods (Algorithm 9 and 11), which are at the heart of our analysis.

The second ingredient is a regularity property of the optimality conditions of NLP (3.2) with respect to parameter variations, namely *Robinson's strong regularity* [137]. It is a key property in parametric optimisation and has already appeared in related works [148, 157].

3.2.1 Convergence and local analysis of the primal descent methods

Although they also produce a non-increasing sequence of objectives, Algorithms 9 and 11, which are essentially active-set strategies, are very different from Algorithms 8 and 10. As shown in Section 2.2 of Chapter 2, they enjoy a fast local convergence rate (almost super-linear) that cannot be achieved by Algorithms 8 and 10. Nevertheless, this fast convergence property only holds when

the active-set has settled down, so that the algorithm becomes an inexact Newton method on a face that contains a critical point. Obviously, in the context of parametric optimisation, one cannot reasonably assume that the primal warm-start $\bar{x}(s_k)$ lies on a face containing a critical point of the parametric augmented Lagrangian at s_{k+1} , as active-set changes are likely to occur when the parameter varies from s_k to s_{k+1} . One can only ensure that the primal warm-start \bar{x}_k is close to a critical point of the parametric augmented Lagrangian. Therefore, we feel a strong need to derive new local convergence rates for Algorithms 9 and 11, which are independent of the active-set dynamics.

First, based on [12], we present the convergence properties of Algorithms 8 and 10. Then, using some ideas of [10], we provide novel local convergence rates for Algorithms 9 and 11. Some structural assumptions on the problem data are required for our results to hold. More precisely, NLP (3.2) is assumed to be semi-algebraic.

Assumption 3.2. *The functions Q_c , J_i and g_i are multivariate polynomials. For each $i \in \{1, \dots, N\}$, $\deg(J_i) \geq 2$.*

Remark 3.8. *From a control perspective, this implies that the theoretical developments that follow are valid when NLP (3.2) is obtained via discretisation of optimal control problems with polynomial dynamics and quadratic costs for instance.*

As Q_c , J_i and g_i are multivariate polynomials, the function $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1})$ is a multivariate polynomial, whose degree is assumed to be larger than 2. We define

$$d_L := \deg(L_\varrho(\cdot, \mu, s)) \geq 2 . \quad (3.12)$$

It has been shown that semi-algebraic functions satisfy the KL inequality at their critical points [20]. The following Theorem is a formulation of the KL property for a multivariate polynomial function over a polyhedron. This matches the parametric augmented Lagrangian subproblem. In this particular case, the Lojasiewicz exponent can be explicitly computed. It is proven to be a simple function of the degree of the polynomial and its dimension. Theorem 3.1 that follows is an extension of the result of [41] to the case of a multivariate polynomial over a polyhedral set.

Theorem 3.1. *Let $L : \mathbb{R}^n \rightarrow \mathbb{R}$ be a polynomial function of degree $\deg(L) \geq 2$ with $n \geq 1$. Let Ω be a non-trivial polyhedral set in \mathbb{R}^n . Assume that all restrictions of L to faces of Ω that are not vertices, have degree larger than two. Given x^* a critical point of $L + \iota_\Omega$, there exists constant $\delta > 0$ and $c > 0$ such that for all $x \in \mathcal{B}(x^*, \delta) \cap \Omega$ and all $v \in \mathcal{N}_\Omega(x)$,*

$$\|\nabla L(x) + v\|_2 \geq c |L(x) - L(x^*)|^{\theta(\deg(L), n)} , \quad (3.13)$$

where

$$\theta(d, n) := 1 - \frac{1}{d(3d-3)^{n-1}} . \quad (3.14)$$

Proof. Let x^* be a critical point of $L + \iota_\Omega$. From [10], as $L + \iota_\Omega$ is a semi-algebraic function, there exists a radius $\delta' > 0$, a constant $c' > 0$ and a coefficient $\theta' \in (0, 1)$ such that for all $x \in \mathcal{B}(x^*, \delta') \cap \Omega$ and all $v \in \mathcal{N}_\Omega(x)$,

$$\|\nabla L(x) + v\|_2 \geq c' |L(x) - L(x^*)|^{\theta'} . \quad (3.15)$$

Define θ'_f as the infimum of all θ' for which (3.15) is satisfied. Our goal is to show that

$$\theta'_f \leq \theta(\deg(L), n) ,$$

as it directly implies that (3.13) is satisfied. One can assume that $\theta'_f > 0$, since for $\theta'_f = 0$ the proof would be immediate. For the sake of contradiction, assume that

$$\theta'_f > \theta(\deg(L), n) .$$

Hence, one can pick $\tilde{\theta} \in (\theta(\deg(L), n), \theta'_f)$ and $c'' > 0$, and construct a sequence $\{(x_n, v_n)\}$ satisfying for all $n \geq 1$,

$$\begin{cases} x_n \in \mathcal{B}\left(x^*, \frac{1}{n}\right) \cap \Omega, v_n \in \mathcal{N}_\Omega(x_n) \\ \|\nabla L(x_n) + v_n\|_2 < c'' |L(x_n) - L(x^*)|^{\tilde{\theta}} \end{cases} . \quad (3.16)$$

Without loss of generality, one can find a face \mathcal{F} of Ω , which is not a vertex and contains x^* , and a subsequence $\{x_{n_k}\}$ such that

$$x_{n_k} \in \text{ri } \mathcal{F} ,$$

for k large enough and satisfying (3.16). Moreover, for all $x \in \text{ri } \mathcal{F}$, there exists $p \in \mathbb{R}^{d_{\mathcal{F}}}$ such that

$$x = x^* + Zp ,$$

where $Z \in \mathbb{R}^{n \times d_{\mathcal{F}}}$ is a full column-rank matrix, with $d_{\mathcal{F}}$ the dimension of the affine hull of \mathcal{F} . As the face \mathcal{F} is not a vertex, $d_{\mathcal{F}} \geq 1$. Subsequently, one can define a polynomial function $L^* : \mathbb{R}^{d_{\mathcal{F}}} \rightarrow$

\mathbb{R} as follows

$$L^*(p) := L(x^* + Zp) \quad .$$

From the results of [41] (no matter whether 0 is a critical point of L^* or not, see Remark 3.2 in [11]), as $\deg(L^*) \geq 2$ by assumption, there exists a radius $\delta^* > 0$ and a constant $c^* > 0$ such that for all $p \in \mathcal{B}(0, \delta^*)$

$$\|\nabla L^*(p)\|_2 \geq c^* |L^*(p) - L^*(0)|^{\theta(\deg(L^*), d_{\mathcal{F}})} \quad .$$

However, $\deg(L^*) \leq \deg(L)$ and $d_{\mathcal{F}} \leq n$, which implies that

$$\theta(\deg(L^*), d_{\mathcal{F}}) \leq \theta(\deg(L), n) \quad , \quad (3.17)$$

from the definition of θ in (3.14). As L^* is a continuous function, the radius δ^* can always be chosen such that

$$|L^*(p) - L^*(0)| < 1 \quad .$$

This implies that for all $p \in \mathcal{B}(0, \delta^*)$,

$$\|\nabla L^*(p)\|_2 \geq c^* |L^*(p) - L^*(0)|^{\theta(\deg(L), n)} \quad .$$

Hence, there exists $K \geq 1$ such that for all $k \geq K$,

$$\begin{aligned} \|\nabla L(x_{n_k}) + v_{n_k}\|_2 &\geq \frac{1}{\|Z\|_2} \|Z^\top (\nabla L(x_{n_k}) + v_{n_k})\|_2 \\ &\geq \frac{1}{\|Z\|_2} \|Z^\top (\nabla L(x^* + Zp_{n_k}) + v_{n_k})\|_2 \\ &\geq \frac{1}{\|Z\|_2} \|\nabla L^*(p_{n_k})\|_2 \\ &\geq \frac{c^*}{\|Z\|_2} |L(x_{n_k}) - L(x^*)|^{\theta(\deg(L), n)} \quad . \end{aligned}$$

The third inequality follows from $Z^\top v_{n_k} = 0$, as v_{n_k} is in the normal cone to \mathcal{F} . However, since c'' can be chosen equal to $c^*/\|Z\|_2$ as Ω has finitely many faces, the above implies that

$$|L(x_{n_k}) - L(x^*)|^{\theta(\deg(L), n)} < |L(x_{n_k}) - L(x^*)|^{\tilde{\theta}} \quad .$$

This leads to a contradiction for k large enough so that $|L(x_{n_k}) - L(x^*)| < 1$, as $\tilde{\theta} > \theta(\deg(L), n)$

by assumption. \square

Corollary 3.1. *Under Assumption 3.2, given $\mu \in \mathbb{R}^m$, $s \in \mathbb{S}$ and $\varrho > 0$, $L_\varrho(\cdot, \mu, s) + \iota_\Omega$ satisfies inequality (3.13) around all its critical points with radius $\delta > 0$ and constant $c > 0$, where $L_\varrho(\cdot, \mu, s)$ is the parametric augmented Lagrangian defined in (3.3).*

Proof. This is an immediate consequence of Theorem 3.1, as the function $L_\varrho(\cdot, \mu, s)$ is a multivariate polynomial by Assumption 3.2. \square

For the analysis of the first-order descent schemes 8 and 10, we also require that the gradient of the parametric augmented Lagrangian is Lipschitz continuous on all bounded subsets of \mathbb{R}^n .

Assumption 3.3 (Lipschitz continuity of gradient on bounded subsets). *Given \mathcal{E} a bounded subset in \mathbb{R}^n , a Lagrange multiplier $\mu \in \mathbb{R}^m$, a parameter $s \in \mathbb{S}$ and a penalty $\varrho > 0$, the gradient*

$$x \mapsto \nabla_x L_\varrho(x, \mu, s)$$

is Lipschitz continuous on \mathcal{E} . Its Lipschitz constant is denoted by $\ell_\mathcal{E}(\mu, s, \varrho)$.

Our study of Algorithms 8 and 10 is along the lines of [12]. Regarding Algorithms 9 and 11, the proof is based on a novel mechanism compared to the results of [10], [11], [12] or [21], as the activity detection and refinement process deserves a special treatment. Contrary to Algorithms 8 and 10, we do not resort to the KL inequality 3.13 to establish convergence to a critical point, but instead the KL property is a key tool to obtain a novel local convergence rate compared to existing results on trust region methods in the literature.

3.2.1.1 Convergence of Algorithm 8

In this paragraph, we analyse the asymptotic behaviour of the sequence $\{x^l\}_{l \geq 0}$ generated by Algorithm 8 when $M = \infty$. In order to apply Theorem 2.9 in [12], two ingredients are needed, which are a *sufficient decrease* property and a *relative error* condition.

Lemma 3.1 (Sufficient decrease in Algorithm 8). *Assume that the sequence $\{x^l\}$ is bounded. For all $l \geq 1$,*

$$L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) + \iota_\Omega(x^{l+1}) + \frac{r}{2} \|x^{l+1} - x^l\|_2^2 \leq L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) + \iota_\Omega(x^l) \quad . \quad (3.18)$$

Proof. As a direct consequence of Assumption 3.3 and the descent Lemma, the backtracking loop in Algorithm 8 (Lines 5 to 10) terminates after a finite number of iterations with a curvature esti-

mate c^l and an iterate

$$x^{l+1} = P_{\Omega} \left(x^l - \frac{1}{c^l} \nabla L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}) \right)$$

satisfying

$$\begin{aligned} L_{\varrho} (x^{l+1}, \bar{\mu}_k, s_{k+1}) + \frac{r}{2} \|x^{l+1} - x^l\|_2^2 &\leq L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}) + \langle \nabla L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}), x^{l+1} - x^l \rangle \\ &\quad + \frac{c^l}{2} \|x^{l+1} - x^l\|_2^2 . \end{aligned}$$

By definition of x^{l+1} as the projection of

$$x^l - \frac{1}{c^l} \nabla L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1})$$

onto the closed convex set Ω , we have

$$x^{l+1} = \operatorname{argmin}_{x \in \Omega} \langle \nabla L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}), x - x^l \rangle + \frac{c^l}{2} \|x - x^l\|_2^2 ,$$

hence, as $x^l \in \Omega$,

$$\langle \nabla L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}), x - x^l \rangle + \frac{c^l}{2} \|x - x^l\|_2^2 \leq 0 ,$$

and thus,

$$L_{\varrho} (x^{l+1}, \bar{\mu}_k, s_{k+1}) + \frac{r}{2} \|x^{l+1} - x^l\|_2^2 \leq L_{\varrho} (x^l, \bar{\mu}_k, s_{k+1}) ,$$

which yields the sufficient decrease on $L_{\varrho}(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_{\Omega}$, as $\iota_{\Omega}(x^l) = \iota_{\Omega}(x^{l+1})$. \square

Lemma 3.2 (Relative error condition). *Assume that the sequence $\{x^l\}$ is bounded. There exists a positive scalar $\gamma(\bar{\mu}_k, \varrho, s_{k+1})$ such that*

$$\exists v^{l+1} \in \mathcal{N}_{\Omega}(x^{l+1}), \|\nabla_x L_{\varrho}(x^{l+1}, \bar{\mu}_k, s_{k+1}) + v^{l+1}\|_2 \leq \gamma(\bar{\mu}_k, \varrho, s_{k+1}) \|x^{l+1} - x^l\|_2 \quad (3.19)$$

for all $l \geq 0$.

Proof. From the definition of x^{l+1} , there exists a vector v^{l+1} in $\mathcal{N}_{\Omega}(x^{l+1})$ such that

$$0 = v^{l+1} + \nabla_x L_{\varrho}(x^l, \bar{\mu}_k, s_{k+1}) + c^l (x^{l+1} - x^l) ,$$

which implies that

$$v^{l+1} + \nabla_x L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) = c^l (x^l - x^{l+1}) + \nabla_x L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) - \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) .$$

As the sequence $\{x^l\}$ is bounded, there exists $R > 0$ such that $x^l \in \mathcal{B}(0, R)$ for all $l \geq 0$. Moreover, the backtracking procedure (lines 5 to 10) of Algorithm 8 terminates after a finite number of iterations

$$j^l := \left\lceil \frac{\log(r + \ell_{\mathcal{B}(0,R)}(\bar{\mu}_k, s_{k+1}, \rho)/c^{(0)})}{\log \beta} \right\rceil ,$$

where $c^{(0)}$ is an initial guess in the backtracking loop, as defined in Algorithm 8. Hence,

$$c^l \leq \beta^{j^l+1} c^{(0)} .$$

In conclusion, by Assumption 3.3,

$$\|v^{l+1} + \nabla_x L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1})\|_2 \leq \left(\ell_{\mathcal{B}(0,R)}(\bar{\mu}_k, s_{k+1}, \varrho) + \beta^{j^l+1} c^{(0)} \right) \|x^{l+1} - x^l\|_2 .$$

□

Theorem 3.2. *Taking $M = \infty$ in Algorithm 8, if the primal sequence $\{x^l\}$ is bounded, then it converges to a critical point $x^\infty(\bar{\mu}_k, s_{k+1})$ of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$. Moreover, there exists a constant $C > 0$ such that if $\bar{x}_k \in \mathcal{B}(0, \delta)$, where δ is defined in Theorem 3.1,*

$$\|x^M - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 \leq CM^{-\psi(d_L, n)} \|\bar{x}_k - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 , \quad (3.20)$$

where

$$\psi(d, n) := \frac{1}{d(3d-3)^{n-1} - 2} , \quad (3.21)$$

with $d, n \geq 2$.

Proof. By Theorem 3.1, the function $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ satisfies the KL property. Moreover, sufficient decrease is guaranteed by Lemma 3.1 along with a relative error condition in Lemma 3.2. As the sequence $\{x^l\}$ is assumed to be bounded, global convergence to a critical point of

$$L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$$

is a direct consequence of Theorem 2.9 in [12]. The results of [10] and [12] provide an asymptotic

convergence rate estimate, which is a function of the Lojasiewicz exponent

$$\theta(d_L, n)$$

defined in (3.14), which only depends on the dimension of NLP (3.2) and the degree of the polynomial functions involved in it. This is an important point in our analysis, as $\bar{\mu}_k$ and s_k are varying. As $n \geq 2$ and $d_L \geq 2$,

$$\theta(d_L, n) \in \left(\frac{1}{2}, 1\right) .$$

Inequality (3.22) is then a direct consequence of Theorem 2 in [10] as the initial primal iterate is \bar{x}_k . \square

Remark 3.9. *The R-convergence rate estimate (3.22) shows that the convergence of the primal sequence $\{x^l\}$ is locally sublinear. It is not surprising, as Algorithm 8 is a first-order method. However, the convergence rate (3.22) has a rather theoretical flavour, as it does not explicitly depend on the problem conditioning, and good performance may still be obtained in particular cases.*

3.2.1.2 Convergence of Algorithm 10

Like in Chapter 2, we assume blockwise Lipschitz continuity of the gradient of the parametric augmented Lagrangian, in addition to Assumption 3.3.

Assumption 3.4 (Blockwise Lipschitz continuity of augmented Lagrangian gradient). *For all $i \in \{1, \dots, P\}$, given $\chi_1 \in \tilde{\Omega}_1, \dots, \chi_{i-1} \in \tilde{\Omega}_{i-1}$ and $\chi_{i+1} \in \tilde{\Omega}_{i+1}, \dots, \chi_P \in \tilde{\Omega}_P$, the coordinate gradient*

$$\chi_i \mapsto \nabla_{\chi_i} L_{\varrho}(\chi_1, \dots, \chi_{i-1}, \chi_i, \chi_{i+1}, \dots, \chi_P, \bar{\mu}_k, s_{k+1})$$

is Lipschitz continuous with modulus $\ell_i(\chi_1, \dots, \chi_{i-1}, \chi_{i+1}, \dots, \chi_P, \bar{\mu}_k, s_{k+1}, \varrho)$.

Assumption 3.5 (Upper bounds on blockwise Lipschitz constants). *For all $i \in \{1, \dots, P\}$, there exists scalars $\bar{\ell}_i(\bar{\mu}_k, s_{k+1}, \varrho)$ such that for all $l \geq 1$,*

$$\ell_i(\chi_1^l, \dots, \chi_{i-1}^l, \chi_{i+1}^l, \dots, \chi_P^l, \bar{\mu}_k, s_{k+1}, \varrho) \leq \bar{\ell}_i(\bar{\mu}_k, s_{k+1}, \varrho) .$$

Under assumptions that are identical to 3.4 and 3.5, the sufficient decrease and relative error conditions have been proven in paragraph 2.1.2 of Chapter 2. Hence, we just state a convergence theorem for Algorithm 10.

Theorem 3.3. *Taking $M = \infty$ in Algorithm 10, if the primal sequence $\{\chi^l\}$ is bounded, then it converges to a critical point $\chi^\infty(\bar{\mu}_k, s_{k+1})$ of $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$. Moreover, there exists a constant $C > 0$ such that if $\bar{\chi}(s_k) \in \mathcal{B}(0, \delta)$, where δ is defined in Theorem 3.1,*

$$\|\chi^M - \chi^\infty(\bar{\mu}_k, s_{k+1})\|_2 \leq CM^{-\psi(d_L, n)} \|\bar{\chi}(s_k) - \chi^\infty(\bar{\mu}_k, s_{k+1})\|_2, \quad (3.22)$$

where the function ψ has been defined in (3.21).

For Algorithm 8 and its distributed version, Algorithm 10, the theory developed in [10] and [12] has been readily applied. However, for the trust region Algorithms 9 and 11, the sufficient decrease and relative error conditions need to be expressed in a different way from Lemmas 3.1 and 3.2 for the KL property to be used properly. We tackle this issue in the next two paragraphs.

3.2.1.3 Local analysis of Algorithm 9

The arguments of [26] can be directly applied to ensure convergence of the sequence $\{x^l\}$ generated by Algorithm 9 to a critical point of $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ when $M = \infty$. For this, similar assumptions to the ones of Section 2.2 in Chapter 2 should be satisfied in addition to Assumption 3.2.

Assumption 3.6. *For all $k \geq 0$, the parametric augmented Lagrangian $L_\rho(\cdot, \bar{\mu}_k, s_{k+1})$ is bounded below on the set*

$$\{x \in \Omega : L_\rho(x, \bar{\mu}_k, s_{k+1}) \leq L_\rho(\bar{x}(s_k), \bar{\mu}_k, s_{k+1})\}.$$

Assumption 3.7 (Bounded model hessian). *For all $k \geq 0$, there exists a scalar $\bar{B}_\rho(\bar{\mu}_k, s_{k+1})$ such that*

$$\forall x \in \Omega, \|B_\rho(x, \bar{\mu}_k, s_{k+1})\|_2 \leq \bar{B}_\rho(\bar{\mu}_k, s_{k+1}).$$

For clarity, we state the convergence theorem of Algorithm 9. Its proof readily follows from [26] under the strong second-order optimality condition [119].

Theorem 3.4. *Assume that Assumptions 3.6 and 3.7 hold. Taking $M = \infty$ in Algorithm 9, if the sequence $\{x^l\}$ has a limit point $x^\infty(\bar{\mu}_k, s_{k+1})$, at which the strong second-order optimality condition is fulfilled (Assumption 2.15), then it converges to $x^\infty(\bar{\mu}_k, s_{k+1})$, a critical point of $L_\rho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$.*

The standard results on trust region Newton methods show that the sequence $\{x^l\}$ converges to $x^\infty(\bar{\mu}_k, s_{k+1})$ at a superlinear rate, once the active-set at x^l is equal to the active-set at the critical

point $x^\infty(\bar{\mu}_k, s_{k+1})$ [26, 110]. Obviously, for Algorithm 9, such a convergence rate is not very interesting, as the active-set at $\bar{x}(s_k)$ may not be the same as the one at $x^\infty(\bar{\mu}_k, s_{k+1})$ and may also change in the early iterations. We know [26] that it settles down to the active-set at $x^\infty(\bar{\mu}_k, s_{k+1})$ after a finite number of iterations, but such an analytic expression of this number is hard to establish.

Our result holds when the trust region model $m_\varrho(\cdot, \bar{\mu}_k, s_{k+1})$ is a Newton model, that is

$$\forall x \in \Omega, B_\varrho(x, \bar{\mu}_k, s_{k+1}) = \nabla^2 L_\varrho(x, \bar{\mu}_k, s_{k+1}) \quad .$$

This is needed to ensure that the trust region radius is ultimately bounded away from zero ([26, 110] and Chapter 2). We first establish that the step-size α that is computed during the activity detection phase (lines 5 to 6) is bounded from below.

Lemma 3.3 (Lower bound on Cauchy step-size). *Under Assumptions 3.6 and 3.7, there exists a scalar $\underline{\alpha} > 0$ such that for all iteration indices $l \geq 0$ of Algorithm 9, $\alpha^l \geq \underline{\alpha}$.*

Proof. From (3.8),

$$\alpha^l \geq \nu_4 \text{ or } \alpha^l \geq \nu_3 \tilde{\alpha}^l \quad ,$$

where $\tilde{\alpha}^l$ fulfills (3.9) or (3.10). We only need to study the second case, that is $\alpha^l \geq \nu_3 \tilde{\alpha}^l$. From the properties of the projection onto a closed convex set,

$$-\langle \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1}), z^l(\tilde{\alpha}^l) - x^l \rangle \geq \frac{\|z^l(\tilde{\alpha}^l) - x^l\|_2^2}{\tilde{\alpha}^l} \quad ,$$

where

$$z^l(\tilde{\alpha}^l) = P_\Omega(x^l - \tilde{\alpha}^l \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1})) \quad .$$

Combining the inequality above with condition (3.9) yields

$$\frac{1}{2} \langle z^l(\tilde{\alpha}^l) - x^l, \nabla^2 L_\varrho(x^l, \bar{\mu}_k, s_{k+1})(z^l(\tilde{\alpha}^l) - x^l) \rangle \geq \frac{1 - \nu_0}{\tilde{\alpha}^l} \|z^l(\tilde{\alpha}^l) - x^l\|_2^2 \quad ,$$

and then, via the Cauchy-Schwarz inequality,

$$\tilde{\alpha}^l \geq \frac{2(1 - \nu_0)}{\bar{B}_\varrho(\bar{\mu}_k, s_{k+1})} \quad .$$

Thus, for all $l \geq 1$,

$$\alpha^l \geq \min \left\{ \nu_4, \nu_3 \frac{2(1-\nu_0)}{\overline{B}_\rho(\bar{\mu}_k, s_{k+1})} \right\} .$$

The last case to consider is when $\tilde{\alpha}^l$ satisfies (3.10). From existing results on trust region Newton methods [26, 110], we know that the trust region radius Δ^l is asymptotically bounded away from 0 (Theorem 7.4 in [26], Theorem 5.3 in [110]). Hence, there exists $\bar{\Delta} > 0$ such that for all $l \geq 1$,

$$\Delta^l \geq \bar{\Delta} .$$

Subsequently, for all $l \geq 0$,

$$\|z^l(\tilde{\alpha}^l) - x^l\|_\infty \geq \nu_1 \bar{\Delta} .$$

As the sequence $\{\nabla_x L_\rho(x^l, \bar{\mu}_k, s_{k+1})\}$ is bounded, since $\{x^l\}$ converges (Theorem 3.4) and $L_\rho(\cdot, \bar{\mu}_k, s_{k+1})$ is twice differentiable, the above inequality also implies that $\tilde{\alpha}^l$ is bounded from below. \square

As $\{x^l\}$ is convergent and all iterations are ultimately successful (when taking $M = \infty$ in Algorithm 9), we can recast the sequence $\{x^l\}$ as the subsequence of successful iterations only (the iterate x^l does not change if the iteration is unsuccessful). The key ingredient of our analysis is the sequence $\{u^l\}$ defined by

$$u^l := \max \{ \|y^l - x^l\|_2, \|z^l - x^l\|_2 \} . \quad (3.23)$$

Lemma 3.4 (Sufficient decrease (Algorithm 9)). *There exists a scalar $\kappa_1 > 0$ such that for all $l \geq 0$,*

$$L_\rho(x^l, \bar{\mu}_k, s_{k+1}) - L_\rho(x^{l+1}, \bar{\mu}_k, s_{k+1}) \geq \kappa_1 (u^l)^2 . \quad (3.24)$$

Proof. From the definition of $\{x^l\}$ as the sequence of successful iterations in Algorithm 9,

$$x^{l+1} = y^l .$$

Hence,

$$\begin{aligned}
 L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) &= m_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(y^l, \bar{\mu}_k, s_{k+1}) \\
 &\geq \eta_1 (m_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(y^l, \bar{\mu}_k, s_{k+1})) \\
 &\geq \eta_1 \left(m_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) \right. \\
 &\quad \left. + m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(y^l, \bar{\mu}_k, s_{k+1}) \right) .
 \end{aligned}$$

However,

$$\begin{aligned}
 m_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) &\geq -\nu_0 \langle \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1}), z^l - x^l \rangle \\
 &\geq \frac{\nu_0}{\alpha^l} \|z^l - x^l\|_2^2 .
 \end{aligned}$$

As $m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(y^l, \bar{\mu}_k, s_{k+1}) \geq 0$, it follows from (3.8) that

$$L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) \geq \frac{\eta_1 \nu_0}{\nu_5} \|z^l - x^l\|_2^2 . \quad (3.25)$$

However, the refinement phase of Algorithm 9 (lines 7 to 8) yields

$$m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(y^l, \bar{\mu}_k, s_{k+1}) \geq \frac{r}{2} \|y^l - z^l\|_2^2 .$$

Hence,

$$\begin{aligned}
 L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) &\geq \eta_1 \min \left\{ \frac{\nu_0}{\nu_5}, \frac{r}{2} \right\} \left(\|z^l - x^l\|_2^2 + \|y^l - z^l\|_2^2 \right) \\
 &\geq \eta_1 \min \left\{ \frac{\nu_0}{\nu_5}, \frac{r}{2} \right\} \max \left\{ \|z^l - x^l\|_2^2, \|y^l - z^l\|_2^2 \right\} .
 \end{aligned}$$

However, by the triangle inequality,

$$\|y^l - x^l\|_2 \leq 2 \max \left\{ \|z^l - x^l\|_2, \|y^l - z^l\|_2 \right\} .$$

Subsequently,

$$L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) \geq \frac{\eta_1}{4} \min \left\{ \frac{\nu_0}{\nu_5}, \frac{r}{2} \right\} \|y^l - x^l\|_2^2 . \quad (3.26)$$

Combining inequalities (3.25) and (3.26), we obtain

$$L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) \geq \eta_1 \frac{\nu_0}{\nu_5} \max \left\{ \|z^l - x^l\|_2^2, \|y^l - x^l\|_2^2 \right\},$$

which corresponds to (3.24) by posing

$$\kappa_1 := \eta_1 \frac{\nu_0}{\nu_5}.$$

□

Similarly, a relative error condition can be expressed in terms of the sequence $\{u^l\}$.

Lemma 3.5 (Relative error condition (Algorithm 9)). *Under Assumptions 3.6 and 3.7, there exists a scalar $\kappa_2 > 0$ such that for all $l \geq 0$,*

$$\exists v^{l+1} \in \mathcal{N}_\Omega(x^{l+1}), \|v^{l+1} + \nabla_x L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1})\| \leq \kappa_2 u^l. \quad (3.27)$$

Proof. By definition of the Cauchy point z^l , there exists a vector $v \in \mathcal{N}_\Omega(z^l)$ such that

$$0 = v + \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) + \frac{z^l - x^l}{\alpha^l},$$

which yields

$$\|v + \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1})\|_2 \leq \frac{1}{\underline{\alpha}} \|z^l - x^l\|_2,$$

by Lemma 3.3. Hence, by the reverse triangle inequality and Assumption 3.7,

$$\begin{aligned} \|v + \nabla_x L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1})\|_2 &\leq \frac{1}{\underline{\alpha}} \|z^l - x^l\|_2 + \bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) \|y^l - x^l\|_2 \\ &\leq 2 \max \left\{ \frac{1}{\underline{\alpha}}, \bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) \right\} u^l. \end{aligned} \quad (3.28)$$

From the definition of the refinement phase in Algorithm 9 (lines 7 to 8),

$$\mathcal{A}_\Omega(z^l) \subseteq \mathcal{A}_\Omega(x^{l+1}).$$

As Ω is a polyhedral set, this implies

$$\mathcal{N}_\Omega(z^l) \subseteq \mathcal{N}_\Omega(x^{l+1}),$$

and $v \in \mathcal{N}_\Omega(x^{l+1})$. The relative error condition (3.27) follows by posing $v^{l+1} := v$ and

$$\kappa_2 := 2 \max \left\{ \frac{1}{\underline{\alpha}}, \bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) \right\} .$$

□

Theorem 3.5. *Under Assumptions 3.2, 3.6 and 3.7, the series*

$$\sum_{l \geq 0} u^l$$

is bounded.

Proof. The proof is based on the arguments of [10], but adapts to the new role played by the sequence $\{x^l\}$ defined in (3.23). Without loss of generality, one can assume that

$$L_\varrho(x^\infty(\bar{\mu}_k, s_{k+1}), \bar{\mu}_k, s_{k+1}) = 0 ,$$

where $x^\infty(\bar{\mu}_k, s_{k+1})$ has been defined in Theorem 3.4. Then, the sufficient decrease inequality 3.24 implies that the objective sequence $\{L_\varrho(x^l, \bar{\mu}_k, s_{k+1})\}$ is positive and decreases to 0. As the sequence $\{x^l\}$ converges to the critical point $x^\infty(\bar{\mu}_k, s_{k+1})$ (Theorem 3.4), there exists an integer $l_1 \geq 1$ such that for all l larger than l_1 ,

$$x^l \in \mathcal{B}(x^\infty(\bar{\mu}_k, s_{k+1}), \delta) ,$$

where δ is defined in Theorem 3.1. Hence, by inequality (3.13),

$$\begin{aligned} L_\varrho(x^l, \bar{\mu}_k, s_{k+1})^{\theta(d_L, n)} &\leq \frac{1}{c} \|v^l + \nabla_x L_\varrho(x^l, \bar{\mu}_k, s_{k+1})\|_2 \\ &\leq \frac{\kappa_2}{c} u^{l-1} , \end{aligned} \tag{3.29}$$

where the last inequality follows from Lemma 3.5. However, by convexity of the function

$$t \mapsto -t^{1-\theta}$$

for $t > 0$, it comes

$$\begin{aligned}
 L_\varrho(x^l)^{1-\theta(d_L, n)} - L_\varrho(x^{l+1})^{1-\theta(d_L, n)} &\geq (1 - \theta(d_L, n)) L_\varrho(x^l, \bar{\mu}_k, s_{k+1})^{-\theta(d_L, n)} \left(L_\varrho(x^l, \bar{\mu}_k, s_{k+1}) \right. \\
 &\quad \left. - L_\varrho(x^{l+1}, \bar{\mu}_k, s_{k+1}) \right) \\
 &\geq (1 - \theta(d_L, n)) L_\varrho(x^l, \bar{\mu}_k, s_{k+1})^{-\theta(d_L, n)} \kappa_1 (u^l)^2 \\
 &\geq \frac{c\kappa_1 (1 - \theta(d_L, n)) (u^l)^2}{\kappa_2 u^{l-1}} ,
 \end{aligned}$$

where the second inequality follows from Lemma 3.4 and the third inequality follows from (3.29). At this point, one can follow the same mechanism as in [10] and show by induction that for a fixed $\omega \in (0, 1)$,

$$\sum_{j=l_1}^l u^j \leq \frac{\omega}{1-\omega} u^{l_1-1} + \frac{\kappa_2}{\omega(1-\omega)c\kappa_1(1-\theta(d_L, n))} (L_\varrho(x^{l_1}, \bar{\mu}_k, s_{k+1}) - L_\varrho(x^{l_1+1}, \bar{\mu}_k, s_{k+1})) , \quad (3.30)$$

for all $l \geq l_1$, which yields the result via Assumption 3.6. \square

Based on the upper bound (3.30) on the tail of the series $\sum_{l \geq 0} u^l$ and the results of [10], a local convergence rate estimate for Algorithm 9 can now be derived.

Theorem 3.6 (Local convergence rate of Algorithm 9). *Let $\{x^l\}$ be the sequence generated by Algorithm 9 (successful and unsuccessful iterations). There exists a radius $\eta > 0$ and a constant $C > 0$ such that if $\bar{x}_k \in \mathcal{B}(x^\infty(\bar{\mu}_k, s_{k+1}), \eta)$,*

$$\|x^M - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 \leq CS(M)^{-\psi(d_L, n)} \|\bar{x}_k - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 , \quad (3.31)$$

where

$$\mathcal{S}(M) := \#\{l \in \{0, \dots, M\} : l \text{ successful iteration in Algorithm 9}\} . \quad (3.32)$$

Proof. Given $l \geq 1$, define the sequence $\{\Gamma^l\}$ as

$$\Gamma^l := \sum_{j=l}^{+\infty} u^j .$$

It is well-defined by Theorem 3.5. Moreover,

$$\begin{aligned} \|x^l - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 &\leq \sum_{j=l}^{+\infty} \|x^j - x^{j+1}\|_2 \\ &\leq \Gamma^l, \end{aligned}$$

by definition of Γ^l and u^l in (3.23). From inequality (3.30) with $\omega \in (0, 1)$, it comes

$$\begin{aligned} \Gamma^l &\leq \frac{1}{1-\omega} u^{l-1} + \frac{\kappa_2}{\omega(1-\omega)c\kappa_1(1-\theta(d_L, n))} L_\varrho(x^l, \bar{\mu}_k, s_{k+1})^{1-\theta(d_L, n)} \\ &\leq \frac{1}{1-\omega} u^{l-1} + \frac{\kappa_2}{\omega(1-\omega)c\kappa_1(1-\theta(d_L, n))} \left(\frac{\kappa_2}{c}\right)^{\frac{1-\theta(d_L, n)}{\theta(d_L, n)}} (u^{l-1})^{\frac{1-\theta(d_L, n)}{\theta(d_L, n)}} \\ &\leq \frac{1}{1-\omega} (\Gamma^{l-1} - \Gamma^l) + \frac{\kappa_2}{\omega(1-\omega)c\kappa_1(1-\theta(d_L, n))} \left(\frac{\kappa_2}{c}\right)^{\frac{1-\theta(d_L, n)}{\theta(d_L, n)}} (\Gamma^{l-1} - \Gamma^l)^{\frac{1-\theta(d_L, n)}{\theta(d_L, n)}}, \end{aligned}$$

where the second inequality follows from (3.29). At this point, the reasoning in the proof of Theorem 2 in [10] can be directly applied to obtain the local convergence rate (3.31) after recasting the sequence $\{x^l\}$ as the sequence of successful and unsuccessful iterations in Algorithm 9. \square

3.2.1.4 Local analysis of Algorithm 11

Similarly to Algorithm 9, Algorithm 11 proceeds by activity detection and refinement. Hence, we obtain a local convergence rate via the same reasoning as for Algorithm 9, except that the derivation of the sufficient decrease and relative error inequalities is slightly different due to the alternating projections (lines 5 to 9). The convergence of the sequence $\{\chi^l\}$ to a critical point of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ when $M = \infty$ follows from the analysis in paragraph 2.2.2 of Chapter 2.

Theorem 3.7. *Under Assumptions 2.15, 3.6 and 3.7, by taking $M = \infty$ in Algorithm 11, the sequence $\{\chi^l\}$ converges to $\chi^\infty(\bar{\mu}_k, s_{k+1})$, a critical point of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$.*

Lemma 3.6 (Lower bounds on Cauchy step-sizes). *For all $i \in \{1, \dots, P\}$, there exist scalars $\underline{\alpha}_i > 0$ such that for all $l \geq 0$, $\alpha_i^l \geq \underline{\alpha}_i$.*

Proof. The proof is almost the same as for Lemma 3.3. In the case where $\alpha_i^l \geq \nu_3 \tilde{\alpha}_i^l$ and $\tilde{\alpha}_i^l$ satisfies (2.56), using the same reasoning as in the proof of Lemma 2.7 in paragraph 2.2.1, one can show that

$$\tilde{\alpha}_i^l \geq \frac{2(1-\nu_0)}{\bar{B}_\varrho(\bar{\mu}_k, s_{k+1})}.$$

Lemma 2.12 in paragraph 2.2.2 shows that the trust region radius is bounded away from zero, hence $\tilde{\alpha}_i^l$ is also bounded from below in the case where

$$\|z_i^l(\tilde{\alpha}_i^l) - \chi_i^l\|_\infty \geq \nu_1 \Delta^l .$$

□

We recast the sequence $\{x^l\}$ generated by Algorithm 11 as the subsequence of successful iterations. The sequence $\{u^l\}$ is constructed in the same way as in the previous paragraph

$$u^l := \max \left\{ \|z^l - \chi^l\|_2, \|y^l - \chi^l\|_2 \right\}$$

for $l \geq 0$.

Lemma 3.7 (Sufficient decrease(Algorithm 11)). *There exists a scalar $\kappa_1 > 0$ such that for all $l \geq 0$,*

$$L_\varrho(\chi^l, \bar{\mu}_k, s_{k+1}) - L_\varrho(\chi^{l+1}, \bar{\mu}_k, s_{k+1}) \geq \kappa_1 (u^l)^2 . \quad (3.33)$$

Proof. The proof proceeds as for Lemma 3.4, the only difference being in the derivation of the lower bound on

$$m_\varrho(\chi^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) .$$

For every $i \in \{1, \dots, P\}$, we have

$$\begin{aligned} m_\varrho(z_{[1, i-1]}^l, \chi_i^l, \chi_{[i+1, P]}^l \bar{\mu}_k, s_{k+1}) - m_\varrho(z_{[1, i-1]}^l, z_i^l, \chi_{[i+1, P]}^l \bar{\mu}_k, s_{k+1}) &\geq \frac{\nu_0}{\alpha_i^l} \|z_i^l - \chi_i^l\|_2^2 \\ &\geq \frac{\nu_0}{\nu_5} \|z_i^l - \chi_i^l\|_2^2 . \end{aligned}$$

Hence,

$$m_\varrho(\chi^l, \bar{\mu}_k, s_{k+1}) - m_\varrho(z^l, \bar{\mu}_k, s_{k+1}) \geq \frac{\nu_0}{\nu_5} \|z^l - \chi^l\|_2^2 .$$

The reasoning is then the same as in the proof of Lemma 3.4. □

Lemma 3.8 (Relative error condition (Algorithm 11)). *Under Assumptions 3.6 and 3.7, there exists a scalar $\kappa_2 > 0$ such that for all $l \geq 0$,*

$$\exists v^{l+1} \in \mathcal{N}_{\tilde{\Omega}}(\chi^{l+1}), \|v^{l+1} + \nabla_{\chi} L_\varrho(\chi^{l+1}, \bar{\mu}_k, s_{k+1})\| \leq \kappa_2 u^l . \quad (3.34)$$

Proof. Let $i \in \{1, \dots, P\}$ and $l \geq 0$. From the definition of the Cauchy point z_i^l at subblock i , there exists $v_i \in \mathcal{N}_{\bar{\Omega}_i}(z_i^l)$ such that

$$\begin{aligned} 0 &= v_i + \nabla_{\chi_i} m_\varrho(z_{[1,i-1]}^l, \chi_i^l, \chi_{[i+1,P]}^l, \bar{\mu}_k, s_{k+1}) + \frac{z_i^l - \chi_i^l}{\alpha_i^l} \\ &= v_i + \nabla_{\chi_i} L_\varrho(\chi^l, \bar{\mu}_k, s_{k+1}) + E_i B_\varrho(\chi^l, \bar{\mu}_k, s_{k+1}) E_{[1,i-1]}^\top (z_{[1,i-1]}^l - \chi_{[1,i-1]}^l) + \frac{z_i^l - \chi_i^l}{\alpha_i^l}. \end{aligned}$$

Hence,

$$\|v_i + \nabla_{\chi_i} L_\varrho(\chi^l, \bar{\mu}_k, s_{k+1})\|_2 \leq \bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) \|z^l - \chi^l\|_2 + \frac{1}{\underline{\alpha}_i} \|z_i^l - \chi_i^l\|_2.$$

By taking $v^{l+1} := (v_1^\top, \dots, v_P^\top)^\top$, we obtain

$$\|v^{l+1} + \nabla_\chi L_\varrho(\chi^l, \bar{\mu}_k, s_{k+1})\|_2 \leq P \left(\bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) + \frac{1}{\min\{\underline{\alpha}_i : i \in \{1, \dots, P\}\}} \right) \|z^l - \chi^l\|_2.$$

In conclusion, by the reverse triangle inequality and Assumption 3.7,

$$\begin{aligned} \|v^{l+1} + \nabla_\chi L_\varrho(\chi^{l+1}, \bar{\mu}_k, s_{k+1})\|_2 &\leq P \left(\bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) + \frac{1}{\min\{\underline{\alpha}_i : i \in \{1, \dots, P\}\}} \right) \|z^l - \chi^l\|_2 \\ &\quad + \bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) \|y^l - \chi^l\|_2 \end{aligned} \quad (3.35)$$

and the relative error inequality (3.34) follows by posing

$$\kappa_2 := 2P \left(\bar{B}_\varrho(\bar{\mu}_k, s_{k+1}) + \frac{1}{\min\{\underline{\alpha}_i : i \in \{1, \dots, P\}\}} \right).$$

□

The local convergence rate for Algorithm 11 is proven using the mechanism of the previous paragraph. The proof is the same as for Theorem 3.6.

Theorem 3.8 (Local convergence rate of Algorithm 11). *Let $\{\chi^l\}$ be the sequence generated by Algorithm 11. There exists a radius $\eta > 0$ and a constant $C > 0$ such that if $\bar{\chi}_k \in \mathcal{B}(\chi^\infty(\bar{\mu}_k, s_{k+1}), \eta)$,*

$$\|\chi^M - \chi^\infty(\bar{\mu}_k, s_{k+1})\|_2 \leq CS(M)^{-\psi(d_L, n)} \|\bar{\chi}_k - \chi^\infty(\bar{\mu}_k, s_{k+1})\|_2, \quad (3.36)$$

where

$$\mathcal{S}(M) := \#\{l \in \{0, \dots, M\} : l \text{ successful iteration in Algorithm 11}\}. \quad (3.37)$$

With the convergence rates of Theorems 3.2, 3.3, 3.6 and 3.8, we are equipped for analysing the stability of Algorithm 7.

3.2.2 Local primal-dual contraction

Given a parameter $s \in \mathbb{S}$, KKT points $w^*(s) = (x^*(s)^\top, \mu^*(s)^\top)^\top$ of the parametric nonlinear program (3.2) satisfy $x^*(s) \in \Omega$ and

$$\begin{cases} 0 \in \nabla_x J(x^*(s)) + \nabla_x G(x^*(s), s)^\top \mu^*(s) + \mathcal{N}_\Omega(x^*(s)) \\ G(x^*(s), s) = 0 \end{cases} . \quad (3.38)$$

Relation (3.38) can be re-written as the generalised equation

$$0 \in F(w, s) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w) , \quad (3.39)$$

where

$$F(w, s) := \begin{pmatrix} \nabla_x J(x) + \nabla_x G(x, s)^\top \mu \\ G(x, s) \end{pmatrix} , w = \begin{pmatrix} x \\ \mu \end{pmatrix} .$$

In order to analyse the behaviour of the KKT points of (3.2) as the parameter s_k evolves, the generalised equation (3.39) needs to satisfy some regularity assumptions. This is captured by the *strong regularity* concept [137], which has already been introduced in Chapter 1.3 and applied in Chapter 2, and is stated again here for clarity.

Definition 3.1 (Strong regularity,[137]). *Let Ω be a compact convex set in \mathbb{R}^n and $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ a differentiable mapping. A generalised equation $0 \in f(x) + \mathcal{N}_\Omega(x)$ is said to be strongly regular at a solution $x^* \in \Omega$ if there exists radii $\eta > 0$ and $\kappa > 0$ such that for all $r \in \mathcal{B}(0, \eta)$, there exists a unique $x_r \in \mathcal{B}(x^*, \kappa)$ such that*

$$r \in f(x^*) + \nabla f(x^*)(x_r - x^*) + \mathcal{N}_\Omega(x_r) , \quad (3.40)$$

and the inverse mapping $r \mapsto x_r$ from $\mathcal{B}(0, \eta)$ to $\mathcal{B}(x^, \kappa)$ is Lipschitz continuous.*

Remark 3.10. *In the case of a polyhedral constraint set Ω , it is worth noting that strong regularity incorporates active-set changes in its definition, as the normal cone is taken at x_r in Eq. (3.40). The set of active constraints at x_r may be different from the one at x^* . Nevertheless, Lipschitz continuity of the solution is still preserved locally.*

Remark 3.11. *As the constraint set Ω in (3.2) is polyhedral, it can be shown that strong regularity*

of a KKT point of (3.2) is equivalent to linear independence constraints qualification and strong second-order optimality [49], which are standard assumptions in nonlinear programming [119].

As the parameter s_k changes, strong regularity is assumed for every index k .

Assumption 3.8. For all parameters $s_k \in \mathbb{S}$ and associated solutions w_k^* , the generalised equation (3.39) is strongly regular at w_k^* .

From Assumption 3.8, it can be proven that the nonsmooth manifold formed by the solutions to the parametric program (3.2) is locally Lipschitz continuous. The first step to achieve this fundamental property is the following Theorem proven in [137].

Theorem 3.9. There exists radii $\delta_A > 0$ and $r_A > 0$ such that for all $k \in \mathbb{N}$ and all $s \in \mathcal{B}(s_k, r_A)$, there exists a unique $w^*(s) \in \mathcal{B}(w_k^*, \delta_A)$ such that

$$0 \in F(w^*(s), s) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w^*(s))$$

and for all $s, s' \in \mathcal{B}(s_k, r_A)$,

$$\|w^*(s) - w^*(s')\|_2 \leq \lambda_A \|F(w^*(s'), s) - F(w^*(s'), s')\|_2, \quad (3.41)$$

where λ_A is a Lipschitz constant associated with the strong regularity mapping of (3.39).

Remark 3.12. Theorem 3.9 is actually a refinement of Theorem 2.1 in [137], as the radii δ_A and r_A are assumed not to depend on the parameter $s_k \in \mathbb{S}$.

Relation (3.41) does not exactly correspond to Lipschitz continuity. This point is addressed by the following Lemma.

Lemma 3.9. There exists $\lambda_F > 0$ such that for all $w \in \Omega \times \mathbb{R}^m$,

$$\forall s, s' \in \mathbb{S}, \|F(w, s) - F(w, s')\|_2 \leq \lambda_F \|s - s'\|_2. \quad (3.42)$$

Proof. Let $w \in \Omega \times \mathbb{R}^m$ and $s, s' \in \mathbb{S}$.

$$\begin{aligned} F(w, s) - F(w, s') &= \begin{pmatrix} (\nabla_x G(x, s) - \nabla_x G(x, s'))^\top \mu \\ G(x, s) - G(x, s') \end{pmatrix} \\ &= \begin{pmatrix} 0 \\ T_1(s - s') \\ \dots \\ T_N(s - s') \end{pmatrix}. \end{aligned}$$

Hence, (3.42) holds with

$$\lambda_F = N \cdot \max \{ \|T_1\|_2, \dots, \|T_N\|_2 \} .$$

□

Algorithm 7 is designed to track the nonsmooth solution manifold by traveling from neighbourhood to neighbourhood, where Lipschitz continuity of the primal-dual solution holds. Such tracking procedures have been analysed thoroughly in the unconstrained case by [46] for a Newton-type method, in the constrained case by [157] for an augmented Lagrangian approach and in [148] for an adjoint-based SCP method. These previous tracking strategies are purely centralised second-order strategies and do not readily extend to solving NLPs in a distributed manner. Our Algorithm 7 proposes a novel way of computing predictor steps along the solution manifold via a decomposition approach, which is tailored to convex constraint sets with closed-form projection operators. Such a class encompasses boxes, the non-negative orthant, semi-definite cones and balls for instance. The augmented Lagrangian framework is particularly attractive in this context, as it allows one to preserve ‘nice’ constraints via partial penalisation.

Algorithm 7 is a truncated scheme both in the primal and dual space, as only M primal iterations of a descent method are applied, which are followed by a single dual update. By means of warm-starting, it tracks the nonsmooth solution manifold of the parametric program (3.2). At a given index k , the primal-dual point \bar{w}_k is suboptimal. Thus, a natural question is whether the sub-optimality gap remains stable, as the parameter s_k varies with k , that is if the sub-optimal iterate remains close to the KKT manifold, or converges to it. Intuitively, one can guess that if s_k evolves slowly and the number of primal iterations M is large enough, stability of the sub-optimality error is expected. This section provides a formal statement about the sub-optimality gap and demonstrates that its evolution is governed by the penalty parameter ϱ , the number of primal iterations M and the magnitude of the parameter difference $s_{k+1} - s_k$, which need to be carefully chosen according to the results provided later in the Chapter.

As the overall objective is to analyse the stability of the sub-optimality error $\|\bar{w}_k - w_k^*\|_2$, a unique critical point w_k^* should be defined at every index k . This is one of the roles of strong regularity. Given a critical point w_k^* for problem (3.2) at s_k , its strong regularity (Assumption 3.8) implies that there exists a unique critical point for problem (3.2) at s_{k+1} , assuming that $\|s_{k+1} - s_k\|_2$ is small enough.

Assumption 3.9. *For all $k \geq 0$, $\|s_{k+1} - s_k\|_2 \leq r_A$.*

Remark 3.13. *In an NMPC setting, this assumption is satisfied if the sampling frequency is fast enough compared to the system’s dynamics.*

Lemma 3.10. For all $k \geq 0$ and $s_k \in \mathbb{S}$, given w_k^* such that

$$0 \in F(w_k^*, s_k) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w_k^*) \quad ,$$

there exists a unique $w_{k+1}^* \in \mathcal{B}(w_k^*, \delta_A)$ such that

$$0 \in F(w_{k+1}^*, s_{k+1}) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w_{k+1}^*) \quad .$$

Proof. This is an immediate consequence of Assumption 3.9 and strong regularity of w_k^* for all $k \geq 0$. \square

3.2.2.1 An auxiliary generalised equation

In Algorithm 7, the primal loop, which is initialised at \bar{x}_k (or the reordered variable $\bar{\chi}_k$), converges to $x^\infty(\bar{\mu}_k, s_{k+1})$ (or $\chi^\infty(\bar{\mu}_k, s_{k+1})$), a critical point of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$, by Theorems 3.2, 3.3, 3.4 and 3.7. The following generalised equation characterises critical points of the augmented Lagrangian function $L_\varrho(\cdot, \bar{\mu}, s) + \iota_\Omega$ in the primal-dual space:

$$0 \in H_\varrho(w, d_\varrho(\bar{\mu}), s) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w) \quad , \quad (3.43)$$

where given $\mu_k^* \in \mathbb{R}^m$, one defines $d_\varrho(\bar{\mu}) := (\bar{\mu} - \mu_k^*) / \varrho$ and

$$H_\varrho(w, d_\varrho(\bar{\mu}), s) := \begin{pmatrix} \nabla_x J(x) + \nabla_x G(x, s)^\top \mu \\ G(x, s) + d_\varrho(\bar{\mu}) + \frac{\mu_k^* - \mu}{\varrho} \end{pmatrix} .$$

Lemma 3.11. Let $\bar{\mu} \in \mathbb{R}^m$, $\varrho > 0$ and $s \in \mathbb{S}$. The primal point $x^*(\bar{\mu}, s)$ is a critical point of $L_\varrho(\cdot, \bar{\mu}, s) + \iota_\Omega$ if and only if the primal-dual point

$$w^*(d_\varrho(\bar{\mu}), s) = \begin{pmatrix} x^*(\bar{\mu}, s) \\ \bar{\mu} + \varrho G(x^*(\bar{\mu}, s), s) \end{pmatrix}$$

satisfies (3.43).

Proof. The necessary condition is clear. To prove the sufficient condition, assume that

$$w^*(d_\varrho(\bar{\mu}), s) = (x^*(d_\varrho(\bar{\mu}), s)^\top, \mu^*(d_\varrho(\bar{\mu}), s)^\top)^\top$$

satisfies (3.43). The second half of (3.43) implies that

$$\mu^*(d_\varrho(\bar{\mu}), s) = \bar{\mu} + \varrho G(x^*(d_\varrho(\bar{\mu}), s), s) .$$

Putting this expression in the first part of (3.43), one obtains that $x^*(d_\varrho(\bar{\mu}), s)$ is a critical point of $L_\varrho(\cdot, \bar{\mu}, s) + \iota_\Omega$. \square

In the sequel, a primal-dual point satisfying (3.43) is denoted by $w^*(d_\varrho(\bar{\mu}), s)$ or $w^*(\bar{\mu}, s)$ without distinction. As $x^\infty(\bar{\mu}_k, s_{k+1})$ is a critical point of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$, one can define

$$w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1}) := \begin{pmatrix} x^\infty(\bar{\mu}_k, s_{k+1}) \\ \bar{\mu}_k + \varrho G(x^\infty(\bar{\mu}_k, s_{k+1}), s_{k+1}) \end{pmatrix} , \quad (3.44)$$

which satisfies (3.43). Note that the generalised equation (3.43) is parametric in s and $d_\varrho(\cdot)$, which represents a normalised distance between a dual variable and an optimal dual variable at index k . Assuming that the penalty parameter ϱ is well-chosen, the generalised equation (3.43) can be proven to be strongly regular at a given solution.

Lemma 3.12 (Strong regularity of (3.43)). *There exists $\tilde{\varrho} > 0$ such that for all $\varrho > \tilde{\varrho}$ and $k \geq 0$, (3.43) is strongly regular at $w_k^* = w^*(0, s_k)$.*

Proof. As Ω is polyhedral, this follows from strong regularity of (3.39) for all $k \geq 0$ and the same arguments as in the proof of Lemma 2.1 in Chapter 2. \square

Assumption 3.10. *The penalty parameter satisfies $\varrho > \tilde{\varrho}$.*

From the strong regularity of (3.43) at w_k^* , using Theorem 2.1 in [137], one obtains the following local Lipschitz property of a solution w to (3.43).

Lemma 3.13. *There exists radii $\delta_B > 0$, $r_B > 0$ and $q_B > 0$ such that for all $k \in \mathbb{N}$,*

$$\forall d \in \mathcal{B}(0, q_B), \forall s \in \mathcal{B}(s_k, r_B), \exists! w^*(d, s) \in \mathcal{B}(w_k^*, \delta_B), \\ 0 \in H_\varrho(w^*(d, s), d, s) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w^*(d, s))$$

and for all $d, d' \in \mathcal{B}(0, q_B)$ and all $s, s' \in \mathcal{B}(s_k, r_B)$,

$$\|w^*(d, s) - w^*(d', s')\|_2 \leq \lambda_B \|H_\varrho(w^*(d', s'), d, s) - H_\varrho(w^*(d', s'), d', s')\|_2 ,$$

where $\lambda_B > 0$ is a Lipschitz constant associated with (3.43).

Note that, given $w \in \Omega \times \mathbb{R}^m$, $d, d' \in \mathbb{R}^m$ and $s, s' \in \mathbb{S}$, one can write

$$H_\varrho(w, d, s) - H_\varrho(w, d', s') = F(w, s) - F(w, s') + \begin{pmatrix} 0 \\ d - d' \end{pmatrix},$$

which, from Lemma 3.9, implies the following Lemma.

Lemma 3.14. *There exists $\lambda_H > 0$ such that for all $w \in \Omega \times \mathbb{R}^m$, for all $d, d' \in \mathbb{R}^m$ and all $s, s' \in \mathbb{R}^m$,*

$$\|H_\varrho(w, d, s) - H_\varrho(w, d', s')\|_2 \leq \lambda_H \left\| \begin{pmatrix} d \\ s \end{pmatrix} - \begin{pmatrix} d' \\ s' \end{pmatrix} \right\|_2. \quad (3.45)$$

Proof. After straightforward calculations, one obtains the Lipschitz property with

$$\lambda_H := \sqrt{\max\{\lambda_F^2, 1\} + \lambda_F}.$$

□

3.2.2.2 Derivation of the contraction inequality

In this paragraph, it is proven that under some conditions, which are clarified next, the optimality tracking error $\|\bar{w}_k - w_k^*\|_2$ of Algorithm 7 remains within a specified bounded interval if the parameter s_k varies sufficiently slowly with k .

First, note that given a sub-optimal primal-dual solution \bar{w}_{k+1} and a critical point w_{k+1}^* ,

$$\|\bar{w}_{k+1} - w_{k+1}^*\|_2 \leq \|\bar{w}_{k+1} - w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})\|_2 + \|w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1}) - w_{k+1}^*\|_2, \quad (3.46)$$

where $w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})$ has been defined in (3.44). The analysis then consists of bounding the two right hand side terms in (3.46). The first one can be upper-bounded by applying strong regularity of (3.43) and the second one using the convergence rate of the primal loop in Algorithm 7.

Lemma 3.15. *If $\|s_{k+1} - s_k\|_2$ satisfies*

$$\|s_{k+1} - s_k\|_2 < \min\left\{r_B, \frac{q_B \varrho}{\lambda_A \lambda_F}\right\}, \quad (3.47)$$

where r_B and q_B have been defined in Lemma 3.13, and $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$, then,

$$\|w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1}) - w_{k+1}^*\|_2 \leq \frac{\lambda_B \lambda_H}{\varrho} (\|\bar{w}_k - w_k^*\|_2 + \lambda_A \lambda_F \|s_{k+1} - s_k\|_2). \quad (3.48)$$

Proof. Note that w_{k+1}^* can be rewritten $w_{k+1}^* = w^*(d_\varrho(\mu_{k+1}^*), s_{k+1})$, which is a solution to (3.43) at s_{k+1} .

$$\begin{aligned} \|d_\varrho(\mu_{k+1}^*)\|_2 &= \frac{\|\mu_{k+1}^* - \mu_k^*\|_2}{\varrho} \\ &\leq \frac{\lambda_F \lambda_A}{\varrho} \|s_{k+1} - s_k\|_2 \\ &< q_B, \end{aligned}$$

by applying Theorem 3.9, Lemma 3.9 and from hypothesis (3.47). Moreover,

$$\begin{aligned} \|d_\varrho(\bar{\mu}_k)\|_2 &= \frac{\|\bar{\mu}_k - \mu_k^*\|_2}{\varrho} \\ &\leq \frac{\|\bar{w}_k - w_k^*\|_2}{\varrho} \\ &< q_B. \end{aligned}$$

Now, as $\|s_{k+1} - s_k\|_2 < r_B$ one can apply Lemmas 3.13 and 3.14 to obtain

$$\begin{aligned} \|w^\infty(\bar{\mu}_k, s_{k+1}) - w_{k+1}^*\|_2 &\leq \lambda_B \lambda_H \|d_\varrho(\bar{\mu}_k) - d_\varrho(\mu_{k+1}^*)\|_2 \\ &\leq \frac{\lambda_B \lambda_H}{\varrho} (\|\bar{\mu}_k - \mu_k^*\|_2 + \|\mu_{k+1}^* - \mu_k^*\|_2) \\ &\leq \frac{\lambda_B \lambda_H}{\varrho} (\|\bar{w}_k - w_k^*\|_2 + \lambda_A \lambda_F \|s_{k+1} - s_k\|_2), \end{aligned}$$

by Theorem 3.9. □

In the following Lemma, using the convergence rate estimates presented in Theorems 3.2 or 3.3, we derive a bound on the first summand $\|\bar{w}_{k+1} - w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})\|_2$ when the primal point \bar{x}_{k+1} is computed via Algorithm 9 or 11 respectively.

Lemma 3.16. *Assume that $\{x^l\}$ has been generated by Algorithm 8 or 10. If $\|s_{k+1} - s_k\|_2 < r_B$, $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$ and*

$$\left(1 + \frac{\lambda_H \lambda_B}{\varrho}\right) \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B r_B < \eta,$$

where η has been defined in Theorem 3.2 or 3.3, then

$$\begin{aligned} \|\bar{w}_{k+1} - w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})\|_2 &\leq C(1 + \varrho\lambda_G) M^{-\psi(d_L, n)} \left(\lambda_B \lambda_H \|s_{k+1} - s_k\|_2 \right. \\ &\quad \left. + \|\bar{w}_k - w_k^*\|_2 \left(1 + \frac{\lambda_B \lambda_H}{\varrho}\right) \right), \end{aligned} \quad (3.49)$$

where $\lambda_G > 0$ is the Lipschitz constant of $G(\cdot, s_{k+1})$ on the ball containing the convergent sequence $\{x^l\}$.

Proof. From Algorithm 7, it follows that

$$\begin{aligned} \|\bar{w}_{k+1} - w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})\|_2 &\leq \left\| \left(\begin{array}{c} \bar{x}_{k+1} - x^\infty(\bar{\mu}_k, s_{k+1}) \\ \varrho(G(\bar{x}_{k+1}, s_{k+1}) - G(x^\infty(\bar{\mu}_k, s_{k+1}), s_{k+1})) \end{array} \right) \right\|_2 \\ &\leq (1 + \varrho\lambda_G) \|\bar{x}_{k+1} - x^\infty(\bar{\mu}_k, s_{k+1})\|_2. \end{aligned}$$

In order to apply Theorem 3.2 or 3.3, one first needs to show that \bar{x}_k lies in the ball $\mathcal{B}(x^\infty(\bar{\mu}_k, s_{k+1}), \eta)$,

$$\begin{aligned} \|\bar{x}_k - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 &\leq \|\bar{x}_k - x^*(0, s_k)\|_2 + \|x^*(0, s_k) - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 \\ &\leq \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B (\|d_\varrho(\bar{\mu}_k)\|_2 + \|s_{k+1} - s_k\|_2) \\ &\leq \left(1 + \frac{\lambda_H \lambda_B}{\varrho}\right) \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B \|s_{k+1} - s_k\|_2 \\ &< \delta, \end{aligned} \quad (3.50)$$

where the second step follows from strong regularity of (3.43) at $w^*(0, s_k)$ and the hypotheses mentioned above. Thus one can use the R-convergence rate estimate of Theorem 3.2 or 3.3 and apply the inequalities in (3.50) to obtain (3.49). \square

A related Lemma can be stated when Algorithm 9 or 11 is applied to minimise the parametric augmented Lagrangian.

Lemma 3.17. *Assume that $\{x^l\}$ has been generated by Algorithm 9 or 11. If $\|s_{k+1} - s_k\|_2 < r_B$, $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$ and*

$$\left(1 + \frac{\lambda_H \lambda_B}{\varrho}\right) \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B r_B < \eta,$$

where η has been defined in Theorem 3.6 or 3.8, then

$$\begin{aligned} \|\bar{w}_{k+1} - w^\infty(d_\varrho(\bar{\mu}_k), s_{k+1})\|_2 &\leq C(1 + \varrho\lambda_G) \mathcal{S}(M)^{-\psi(d_L, n)} \left(\lambda_B \lambda_H \|s_{k+1} - s_k\|_2 \right. \\ &\quad \left. + \|\bar{w}_k - w_k^*\|_2 \left(1 + \frac{\lambda_B \lambda_H}{\varrho}\right) \right), \end{aligned} \quad (3.51)$$

where $\lambda_G > 0$ is the Lipschitz constant of $G(\cdot, s_{k+1})$ on the ball containing the convergent sequence $\{\chi^l\}$ and $\mathcal{S}(M)$ has been defined in (3.32) or (3.37) respectively.

Proof. The reasoning is the same as for the proof of Lemma 3.16, except that Theorems 3.6 or 3.8 are applied. \square

Regarding the first-order methods described in Algorithms 8 and 10, by gathering the results of Lemmas 3.15 and 3.16, one can state the following Theorem, which provides an upper-bound on the sub-optimality error at index $k + 1$, expressed as a linear combination of the sub-optimality error at index k and the magnitude of the parameter difference.

Theorem 3.10 (Contraction with Algorithms 8 and 10). *Assume that \bar{x}_{k+1} has been generated via Algorithm 8 or 10. Given an index k , if the primal-dual error $\|\bar{w}_k - w_k^*\|_2$, the number of primal iterations M , the penalty parameter ϱ and the parameter difference $\|s_{k+1} - s_k\|_2$ satisfy*

- $\|s_{k+1} - s_k\|_2 < \min \left\{ r_A, r_B, \frac{q_B \varrho}{\lambda_A \lambda_F} \right\}$,
- $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$,
- $\varrho > \tilde{\varrho}$,
-

$$\left(1 + \frac{\lambda_H \lambda_B}{\varrho}\right) \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B r_B < \eta, \quad (3.52)$$

then the following contraction inequality is satisfied at all indices $k \geq 0$:

$$\|\bar{w}_{k+1} - w_{k+1}^*\|_2 \leq \beta_w(\varrho, M) \|\bar{w}_k - w_k^*\|_2 + \beta_s(\varrho, M) \|s_{k+1} - s_k\|_2, \quad (3.53)$$

where

$$\beta_w(\varrho, M) := C(1 + \varrho\lambda_G) \left(1 + \frac{\lambda_B \lambda_H}{\varrho}\right) M^{-\psi(d_L, n)} + \frac{\lambda_B \lambda_H}{\varrho}, \quad (3.54)$$

and

$$\beta_s(\varrho, M) := C(1 + \varrho\lambda_G) \lambda_B \lambda_H M^{-\psi(d_L, n)} + \frac{\lambda_B \lambda_H \lambda_A \lambda_F}{\varrho} . \quad (3.55)$$

Proof. This is a direct consequence of Lemmas 3.15 and 3.16. \square

Remark 3.14. Note that the last hypothesis (3.52) may be quite restrictive, since $\|\bar{w}_k - w_k^*\|_2$ needs to be sufficiently small for it to be satisfied. However, in many cases the radius η is large ($+\infty$ for strongly convex functions).

A related contraction inequality can be derived when the trust region Algorithms 9 or 11 are applied in order to compute \bar{x}_{k+1} .

Theorem 3.11 (Contraction with Algorithms 9 and 11). Assume that \bar{x}_{k+1} has been generated via Algorithm 9 or 11. Given an index k , if the primal-dual error $\|\bar{w}_k - w_k^*\|_2$, the number of primal iterations M , the penalty parameter ϱ and the parameter difference $\|s_{k+1} - s_k\|_2$ satisfy

- $\|s_{k+1} - s_k\|_2 < \min \left\{ r_A, r_B, \frac{q_B \varrho}{\lambda_A \lambda_F} \right\}$,
- $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$,
- $\varrho > \tilde{\varrho}$,
-

$$\left(1 + \frac{\lambda_H \lambda_B}{\varrho} \right) \|\bar{w}_k - w_k^*\|_2 + \lambda_H \lambda_B r_B < \eta , \quad (3.56)$$

then the following contraction inequality is satisfied at all indices $k \geq 0$:

$$\|\bar{w}_{k+1} - w_{k+1}^*\|_2 \leq \beta_w(\varrho, M) \|\bar{w}_k - w_k^*\|_2 + \beta_s(\varrho, M) \|s_{k+1} - s_k\|_2 , \quad (3.57)$$

where

$$\beta_w(\varrho, M) := C(1 + \varrho\lambda_G) \left(1 + \frac{\lambda_B \lambda_H}{\varrho} \right) \mathcal{S}(M)^{-\psi(d_L, n)} + \frac{\lambda_B \lambda_H}{\varrho} , \quad (3.58)$$

and

$$\beta_s(\varrho, M) := C(1 + \varrho\lambda_G) \lambda_B \lambda_H \mathcal{S}(M)^{-\psi(d_L, n)} + \frac{\lambda_B \lambda_H \lambda_A \lambda_F}{\varrho} . \quad (3.59)$$

Proof. This is a direct consequence of Lemmas 3.15 and 3.17. \square

Remark 3.15. *It is worth noting that Theorem 3.11 does not make use of the specific nature of Algorithms 9 and 11, which benefit from the fast local convergence of Newton's method once the active-set has settled down. However, if the warm-start \bar{x}_k is sufficiently close to $x^\infty(\bar{\mu}_k, s_{k+1})$ and lies on the same active face, then from Theorem 2.6,*

$$\|\bar{x}_{k+1} - x^\infty(\bar{\mu}_k, s_{k+1})\| \leq \epsilon^M \|\bar{x}_k - x^\infty(\bar{\mu}_k, s_{k+1})\|_2 ,$$

where $\epsilon \in (0, 1)$ can be made arbitrarily small. Subsequently, the contraction coefficients become

$$\beta_w(\varrho, M) := C(1 + \varrho\lambda_G) \left(1 + \frac{\lambda_B\lambda_H}{\varrho}\right) \epsilon^M + \frac{\lambda_B\lambda_H}{\varrho}$$

and

$$\beta_s(\varrho, M) := C(1 + \varrho\lambda_G) \lambda_B\lambda_H\epsilon^M + \frac{\lambda_B\lambda_H\lambda_A\lambda_F}{\varrho} .$$

Compared to the expressions (3.58) and (3.59), the coefficients $\beta_w(\varrho, M)$ and $\beta_s(\varrho, M)$ can be made smaller than one with a much smaller number of primal iterations, for a fixed penalty ϱ . Thus, if the optimal active-set is rapidly identified, which is generally the case with gradient projections, then the tracking performance of Algorithm 7 with Algorithm 9 or 11 is improved compared to the case where Algorithm 8 or Algorithm 10 are used as primal solvers.

In order to ensure stability of the sequence of sub-optimal iterates \bar{w}_k , the parameter difference $\|s_{k+1} - s_k\|_2$ has to be sufficiently small and the coefficient $\beta_w(\varrho, M)$ needs to be strictly less than 1. This is clearly satisfied if the penalty parameter ϱ is sufficiently large to make $\lambda_B\lambda_H/\varrho$ small in (3.54) (or (3.58)). Yet the penalty parameter ϱ also appears in $1 + \varrho\lambda_G$. Hence it needs to be balanced by a large enough number of primal iterations M in order to make the first summand in (3.54) (or (3.58)) small. More precisely, enforcing $\beta_w(\varrho, M) < 1$ is equivalent to

$$C\lambda_G M^{-\psi(d_L, n)} \varrho^2 + (CM^{-\psi(d_L, n)}(1 + \lambda_G\lambda_B\lambda_H) - 1) \varrho + \lambda_B\lambda_H(1 + CM^{-\psi(d_L, n)}) < 0 . \quad (3.60)$$

The minimum of the second order polynomial in ϱ (3.60) is attained at

$$\varrho_{\min} = \frac{1}{2\lambda_G} \left(\frac{1}{CM^{-\psi(d_L, n)}} - 1 - \lambda_G\lambda_B\lambda_H \right) .$$

For making ϱ_{\min} positive, the number of primal iterations M needs to be sufficiently large, so that

$$M^{-\psi(d_L, n)} < \frac{1}{C(1 + \lambda_G \lambda_B \lambda_H)} .$$

One can readily show that the minimum of (3.60) is negative if

$$C^2 M^{-2\psi(d_L, n)} (\lambda_B \lambda_H \lambda_G - 1)^2 - 2CM^{-\psi(d_L, n)} (1 + 3\lambda_B \lambda_G \lambda_H) + 1 > 0 ,$$

which is also satisfied for a sufficiently large M . The same analysis applies to the second coefficient $\beta_s(\varrho, M)$ in order to mitigate the effect of the parameter difference $\|s_{k+1} - s_k\|_2$ on the sub-optimality error at $k + 1$.

Corollary 3.2 (Boundedness of the error sequence). *Assume that ϱ and M have been chosen so that $\beta_w(\varrho, M)$ and $\beta_s(\varrho, M)$ are strictly less than 1, and $\varrho > \tilde{\varrho}$. Let $r_w > 0$ such that*

$$\eta - \left(1 + \frac{\lambda_H \lambda_B}{\varrho}\right) r_w - \lambda_H \lambda_B r_s > 0$$

and $r_w < q_B \varrho$, with $r_s > 0$ such that

$$r_s < \frac{(1 - \beta_w(\varrho, M))r_w}{\beta_s(\varrho, M)} .$$

If $\|\bar{w}_0 - w_0^*\|_2 < r_w$ and for all $k \geq 0$,

$$\|s_{k+1} - s_k\|_2 \leq \min \left\{ r_s, r_A, r_B, \frac{q_B \varrho}{\lambda_A \lambda_F} \right\} , \quad (3.61)$$

then for all $k \geq 0$, the error sequence satisfies

$$\|\bar{w}_k - w_k^*\|_2 < r_w .$$

Proof. The proof proceeds by a straightforward induction. At $k = 0$, $\|\bar{w}_0 - w_0^*\|_2 < r_w$ by assumption. Let $k \geq 0$ and assume that $\|\bar{w}_k - w_k^*\|_2 < r_w$. As $\|s_{k+1} - s_k\|_2 < r_A$, by applying Theorem 3.9, there exists a unique $w_{k+1}^* \in \mathcal{B}(w_k^*, \delta_A)$, which satisfies (3.39). As $\|s_{k+1} - s_k\|_2$ satisfies (3.61), $\|\bar{w}_k - w_k^*\|_2 < q_B \varrho$, $\varrho > \tilde{\varrho}$ and (3.56) is satisfied, from the choice of r_w and r_s , we have

$$\begin{aligned} \|\bar{w}_{k+1} - w_{k+1}^*\|_2 &\leq \beta_w(\varrho, M) \|\bar{w}_k - w_k^*\|_2 + \beta_s(\varrho, M) \|s_{k+1} - s_k\|_2 \\ &\leq \beta_w(\varrho, M) r_w + \beta_s(\varrho, M) \|s_{k+1} - s_k\|_2 \\ &\leq r_w , \end{aligned}$$

as

$$\|s_{k+1} - s_k\|_2 \leq r_s < \frac{(1 - \beta_w(\varrho, M)) r_w}{\beta_s(\varrho, M)} .$$

It is worth noting that from the choice of r_w and r_s , the condition (3.56) that is needed for the contraction (3.57), is also recursively satisfied. \square

3.2.2.3 Improved contraction via continuation

In Algorithm 7, only one dual update is performed to move from parameter s_k to parameter s_{k+1} . This contrasts with standard augmented Lagrangian methods, in which the Lagrange multiplier μ and the penalty parameter ϱ are updated after every sequence of primal iterations. Intuitively, one would expect that applying several dual updates instead of just one, drives the suboptimal solution \bar{w}_{k+1} closer to the optimal one w_{k+1}^* , thus enhancing the tracking performance as the parameter s_k varies. However, as the number of primal iterations M is fixed a priori, it is not obvious at all why this would happen, as primal iterations generally need to become more accurate when the dual variable moves closer to optimality, as shown in Chapter 2. Therefore, we resort to an homotopy mechanism [3] so as to fully take advantage of property (3.57).

The parameter s can be seen as an extra degree of freedom in Algorithm 7, which can also be updated along the iterations of Algorithm 7. More precisely, instead of carrying out a sequence of primal descent steps to find a critical point of $L_\varrho(\cdot, \bar{\mu}_k, s_{k+1}) + \iota_\Omega$ directly at the parameter s_{k+1} , one moves from s_k towards s_{k+1} step by step, with each step corresponding to a dual update and a sequence of primal iterations. The proposed approach can be seen as a form of ‘tracking in the tracking’. More precisely, one defines a finite sequence $\{s_k^j\}$ of D parameters along an homotopy path $\{(1 - \tau) s_k + \tau s_{k+1} : \tau \in [0, 1]\}$ by

$$s_k^j := \left(1 - \frac{j}{D}\right) s_k + \frac{j}{D} s_{k+1}, \quad j \in \{0, \dots, D\} , \quad (3.62)$$

where $D \geq 2$. This modification results in Algorithm 12 below. At every step j , the parameter s is first updated. A sequence of descent steps is then applied given the current parameter s and multiplier μ , which is updated at the end of step j . In a sense, Algorithm 12 consists in repeatedly applying Algorithm 7 on an artificial dynamics defined by the homotopy steps.

The rationale behind Algorithm 12 is that it allows for a stronger contraction effect on the sub-optimality error $\|\bar{w}_{k+1} - w_{k+1}^*\|_2$ than Algorithm 7, as formalised by the following Theorem.

Lemma 3.18 (Optimality along the homotopy path). *Given an index $k \geq 0$, for all $j \in \{1, \dots, D\}$,*

Algorithm 12 Homotopy-based optimality tracking algorithm

Input: Suboptimal primal-dual solution $(\bar{x}(s_k)^\top, \bar{\mu}_k^\top)^\top$, parameters s_k and s_{k+1} .

$s \leftarrow s_k, \mu \leftarrow \bar{\mu}_k, x^{\text{wms}} \leftarrow \bar{x}_k$

Continuation loop:

for $j = 1 \dots D$ **do**

$s \leftarrow s + \frac{s_{k+1} - s_k}{D}$

Primal updates:

Compute M iterations of Algorithm 8, 10, 9 or 11 initialised at x^{wms} and obtain x^M

$x^{\text{wms}} \leftarrow x^M$

Dual update: $\mu \leftarrow \mu + \varrho G(x^M, s)$

end for

$\bar{x}(s_{k+1}) \leftarrow x^{\text{wms}}; \bar{\mu}_{k+1} \leftarrow \mu$

there exists a unique primal-dual variable $w^*(s_k^j) \in \mathcal{B}(w_k^*, r_A)$ satisfying

$$0 \in F(w^*(s_k^j), s_k^j) + \mathcal{N}_{\Omega \times \mathbb{R}^m}(w^*(s_k^j)) . \quad (3.63)$$

Proof. This comes directly from the strong regularity of (3.39), Assumption 3.9 and

$$\|s_k^j - s_k\|_2 \leq \|s_{k+1} - s_k\|_2 ,$$

for all $j \in \{1, \dots, D\}$. □

Remark 3.16. Note that the parametric program (3.2) at parameter $s_k^j, j \in \{1, \dots, D\}$, is feasible, by strong regularity of (3.39) at $w^*(s_k^0)$, since $\|s_k^j - s_k\|_2 < r_A$. However, in general, for an arbitrarily large parameter difference $\|s_{k+1} - s_k\|_2$, this is not true, as the set of parameters for which NLP (3.2) is feasible is generally not convex.

Theorem 3.12 (Improved contraction via continuation). Assume that $\varrho > \tilde{\varrho}$ and that ϱ and M have been chosen so that $\beta_w(\varrho, M), \beta_s(\varrho, M) < 1$. Given an index $k \geq 0$, if $\|\bar{w}_k - w_k^*\|_2 < r_w$, where r_w satisfies the assumptions of Corollary 3.2, and $\|s_{k+1} - s_k\|_2$ satisfies (3.61), then the primal-dual sub-optimal variable \bar{w}_{k+1} yielded by Algorithm 12 satisfies the following inequality

$$\|\bar{w}_{k+1} - w_{k+1}^*\|_2 \leq \beta_w^D(\varrho, M) \|\bar{w}_k - w_k^*\|_2 + \beta_s(\varrho, M) \frac{\sum_{i=0}^{D-1} \beta_w^i(\varrho, M)}{D} \|s_{k+1} - s_k\|_2 . \quad (3.64)$$

Proof. For all $j \in \{1, \dots, D\}$, define

$$\bar{\mu}_k^j := \bar{\mu}_k^{j-1} + \varrho G(\bar{z}_k^j, s_k^j)$$

with $\bar{\mu}_k^0 := \bar{\mu}_k$ and where \bar{z}_k^j is obtained after M iterations of Algorithm 8, 10, 9 or 11 on $L_\varrho(\cdot, \bar{\mu}_k^{j-1}, s_k^j) + \iota_\Omega$. Thus, one can define a sub-optimal primal-dual variable

$$\bar{w}_k^j := \left((\bar{z}_k^j)^\top, (\bar{\mu}_k^j)^\top \right)^\top$$

for the homotopy parameter s_k^j . By applying Corollary 3.2, we obtain that for all $j \in \{0, \dots, D-1\}$,

$$\|\bar{w}_k^j - w^*(s_k^j)\|_2 < r_w < q_B \varrho ,$$

since

$$\|s_k^{j+1} - s_k^j\|_2 = \frac{\|s_{k+1} - s_k\|_2}{D} < \min \left\{ r_s, r_A, r_B, \frac{q_B \varrho}{\lambda_A \lambda_F} \right\} .$$

It can also be readily shown that for all $j \in \{0, \dots, D-1\}$,

$$\left(1 + \frac{\lambda_H \lambda_B}{\varrho} \right) \|\bar{w}_k^j - w^*(s_k^j)\|_2 + \lambda_H \lambda_B \|s_k^{j+1} - s_k^j\|_2 < \eta . \quad (3.65)$$

Subsequently, one can apply the same reasoning as for proving Theorem 3.10, and get that for all $j \in \{0, \dots, D-1\}$,

$$\|\bar{w}_k^{j+1} - w^*(s_k^{j+1})\|_2 \leq \beta_w(\varrho, M) \|\bar{w}_k^j - w^*(s_k^j)\|_2 + \beta_s(\varrho, M) \|s_k^{j+1} - s_k^j\|_2 . \quad (3.66)$$

By iterating inequality (3.66) from $j = 0$ to $D-1$, we obtain

$$\begin{aligned} \|\bar{w}_{k+1} - w_{k+1}^*\|_2 &\leq \beta_w(\varrho, M) \|\bar{w}_k^{D-1} - w^*(s_k^{D-1})\|_2 + \frac{\beta_s(\varrho, M)}{D} \|s_{k+1} - s_k\|_2 \\ &\leq \dots \\ &\leq \beta_w^D(\varrho, M) \|\bar{w}_k^0 - w^*(s_k^0)\|_2 + \beta_s(\varrho, M) \frac{\sum_{j=0}^{D-1} \beta_w^j(\varrho, M)}{D} \|s_{k+1} - s_k\|_2 , \end{aligned}$$

which is exactly inequality (3.64). \square

As $\beta_w(\varrho, M) < 1$, $\beta_s(\varrho, M) < 1$ and $D \geq 2$, it follows that

$$\beta_w^D(\varrho, M) < \beta_w(\varrho, M)$$

and

$$\beta_s(\varrho, M) \frac{\sum_{i=0}^{D-1} \beta_w^i(\varrho, M)}{D} < \beta_s(\varrho, M) \quad ,$$

which implies that the contraction (3.64) is stronger than (3.57). In practice, the coefficients $\beta_w(\varrho, M)$ and $\beta_s(\varrho, M)$ in (3.57) can be reduced by an appropriate tuning of the penalty ϱ and the number of primal steps M . Yet this approach is limited, as previously discussed. Therefore, Algorithm 12 provides a more efficient and systematic way of improving the optimality tracking performance. Superiority of Algorithm 12 over Algorithm 7 is demonstrated on a numerical example in Section 3.3.

3.3 Computational Considerations and Numerical Experiments

By making use of partial penalisation, Algorithm 7 allows for a more general problem formulation than the related tracking algorithm of [157], in which the primal QP sub-problem is assumed to have non-negativity constraints only. Moreover, the approach of [157] is likely to be efficient only when the sub-optimal solution lies on the optimal active face so as to guarantee positive definiteness of the hessian of the augmented Lagrangian. In practice, such a requirement seems to be unrealistic in the case of large reference changes or disturbances. In contrast, our framework can handle any polynomial nonconvex objective subject to convex constraint set Ω_i , for which the projection is easy to evaluate. This happens when Ω_i is a ball, an ellipsoid, a box, the non-negative orthant or even second-order conic constraints and the semi-definite cone. However, the theoretical properties derived in Section 3.2 seem to be limited to polyhedral constraint sets.

Remark 3.17. *For many nonconvex sets, such as spheres or mixed-integer sets, the projection can be obtained in closed-form. However, the analysis of Section 3.2 does not readily extend, as Robinson's strong regularity is defined for closed convex sets [137].*

Remark 3.18. *In a distributed framework, any convex polyhedral set Ω_i could be handled by Algorithm 7, as a non-negative slack variable can be introduced for each agent.*

Algorithm 7 can be further refined by introducing local copies of the variables. Considering the NLP

$$\begin{aligned} & \text{minimise } J(x_1, \dots, x_P) \\ & \text{s.t. } G(x_1, \dots, x_P) = 0 \\ & \quad x_1 \in \Omega_1, \dots, x_P \in \Omega_P, \end{aligned}$$

variables y_i can be incorporated in the equality constraints, resulting in

$$\begin{aligned} & \text{minimise } J(y_1, \dots, y_P) \\ & \text{s.t. } G(y_1, \dots, y_P) = 0 \\ & \quad y_i - x_i = 0 \quad \forall i \in \{1, \dots, P\} \\ & \quad x_1 \in \Omega_1, \dots, x_P \in \Omega_P. \end{aligned}$$

Subsequently, at iteration $l + 1$ of Algorithm 8 or 10, some of the steps are given by

$$\text{minimise}_{x_i \in \Omega_i} \nu_i^\top (y_i^{l+1} - x_i) + \frac{\rho}{2} \|y_i^{l+1} - x_i\|_2^2 + \frac{\alpha_i}{2} \|x_i - x_i^l\|_2^2, \quad ,$$

where ν_i is a dual variable associated with the equality constraint $y_i - x_i = 0$. This step can be rewritten

$$\underset{x_i \in \Omega_i}{\text{minimise}} \left\| x_i - \frac{1}{\alpha_i + \varrho} (\alpha_i x_i^l + \varrho y_i^{l+1} + \nu_i) \right\|_2 ,$$

which corresponds to projecting

$$\frac{1}{\alpha_i + \varrho} (\alpha_i x_i^l + \varrho y_i^{l+1} + \nu_i)$$

onto Ω_i . This type of an approach is useful if the minimisation over the y_i variables is tractable, for instance when J is multi-convex and G is multilinear, and the projection onto Ω_i is cheap to compute.

Algorithms 7 and 12 are tested on two nonlinear systems, a DC motor (centralised) in paragraph 3.3.1 and a formation of three unicycles (distributed) in paragraph 3.3.2. The effect of the penalty parameter ϱ and the sampling period Δt is analysed, assuming that a fixed number of iterations can be performed per second. Thus, given a sampling period Δt , the number of communications between the P groups of agents is limited to a fixed value, which models practical limitations of distributed computations. In particular, it is shown that the theoretical results proven in Section 3.2 are able to predict the practical behaviour of the combined system-optimiser dynamics quite well, and that tuning the optimiser's step-size ϱ and the system's step-size Δt has an effect on the closed-loop trajectories.

From a practical perspective, the purpose of the simulations that follow is to investigate the effect of a limited computational power or limited communication rate on the closed-loop performance of our scheme. This is of particular importance in the case of distributed NMPC problems, as in practice, only a limited number of packets can be exchanged between the P groups of agents within a fixed amount of time, which implies that a suboptimal solution is yielded by Algorithm 10 or 11.

Remark 3.19. *In the following examples, the first optimal primal-dual solution w_0^* is computed using the distributed algorithm 4. A random perturbation is then applied to this KKT point.*

3.3.1 DC motor

The first example is a DC motor with continuous-time bilinear dynamics

$$\dot{x} = Ax + Bx \cdot u + c ,$$

where

$$A = \begin{pmatrix} -R_a/L_a & 0 \\ 0 & -B/J \end{pmatrix}, B = \begin{pmatrix} 0 & -k_m/L_a \\ k_m/J & 0 \end{pmatrix},$$

$$c = \begin{pmatrix} u_a/L_a \\ -\tau_l/J \end{pmatrix},$$

and the parameters are borrowed from the experimental identification presented in [42]:

$$L_a = 0.307 \text{ H}, R_a = 12.548 \text{ } \Omega, k_m = 0.22567 \text{ Nm/A}^2, J = 0.00385 \text{ Nm}\cdot\text{sec}^2,$$

$$B = 0.00783 \text{ Nm}\cdot\text{sec}, \tau_l = 1.47 \text{ Nm}, u_a = 60 \text{ V}.$$

The first component of the state variable x_1 is the armature current, while the second component x_2 is the angular speed. The control input u is the field current of the machine. The control objective is to make the angular speed track a piecewise constant reference $x_2^{\text{ref}} = \pm 2 \text{ rad/sec}$, while satisfying the following state and input constraints:

$$\underline{x} = \begin{pmatrix} -2 \text{ A} \\ -8 \text{ rad/sec} \end{pmatrix}, \bar{x} = \begin{pmatrix} 5 \text{ A} \\ 1.5 \text{ rad/sec} \end{pmatrix},$$

$$\underline{u} = 1.27 \text{ A}, \bar{u} = 1.4 \text{ A}.$$

The continuous-time NMPC problem for reference tracking is discretised at a given sampling period Δt using an explicit Euler method, which results in a bilinear NLP. Although the consistency of the explicit Euler integrator is 1, only the first control input is applied to the real system, implying that the prediction error with respect to the continuous-time dynamics is small for sufficiently small sampling periods Δt . For simulating the closed-loop system under the computed NMPC control law, the MATLAB integrator `ode45` is used with the sampling period Δt . The prediction horizon is fixed at 30 samples. This is a key requirement for the analysis that follows, as explained later.

In general, the computational power of an embedded computing platform is quite limited, meaning that the total number of primal steps that can be computed within one second by Algorithms 7 and 12 is fixed and finite. Later on, we refer to this number as the *computational power*, expressed in proj/sec . The results plotted in Figs. 3.1, 3.2, 3.3 and 3.4 are obtained for a computational power of $2 \cdot 10^3 \text{ proj/sec}$. In Fig. 3.1, it clearly appears that a better tracking performance is obtained for $\Delta t = 0.018 \text{ sec}$, compared to a lower sampling period ($\Delta t = 0.004 \text{ sec}$) or a larger sampling period ($\Delta t = 0.04 \text{ sec}$). The effect of the system's step-size Δt on the performance of Algorithm 7 given a fixed computational power is demonstrated more clearly in Fig. 3.5.

Another key parameter is the penalty coefficient ϱ , which can also be interpreted as a step-size

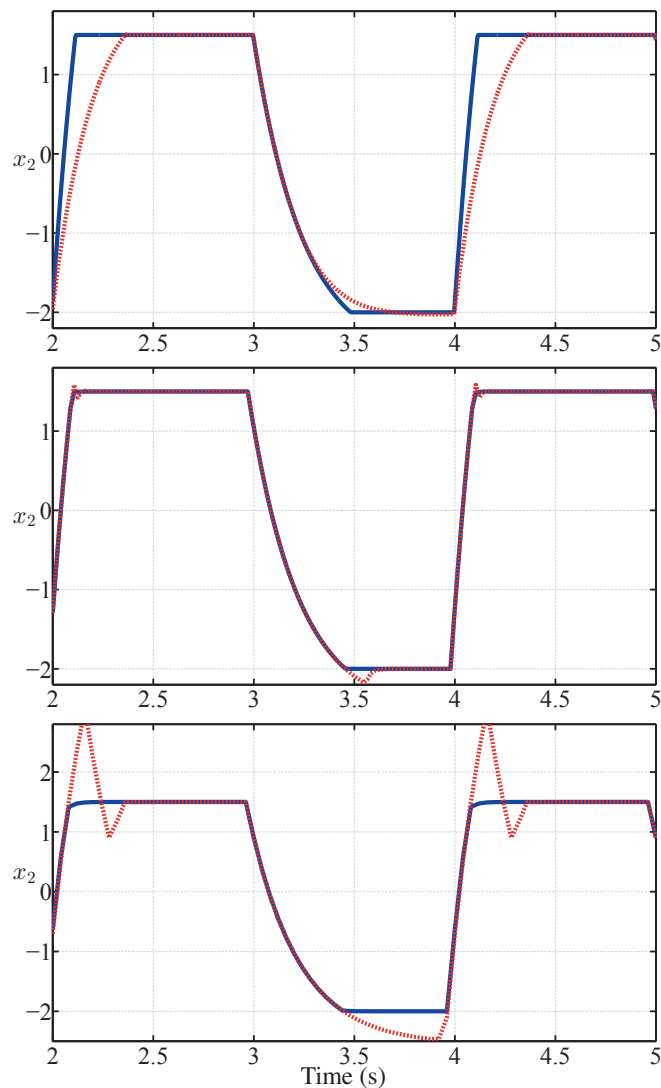


Figure 3.1: Angular speed against time for increasing sampling periods Δt and a fixed computational power $2 \cdot 10^3 \text{proj/sec}$: 0.004 sec (top), 0.018 sec (middle) and 0.04 sec (bottom). The sub-optimal trajectory obtained with Algorithm 7 is plotted in dashed red, while the full NMPC trajectory obtained using IPOPT (for the same Δt) is in blue.

for the optimiser. In order to demonstrate the effect of ϱ on the efficacy of our optimality tracking splitting scheme, the sampling period Δt is fixed at 0.018 sec given a computational power of $2 \cdot 10^3 \text{proj/sec}$, which implies that the total number of primal iterations is $M = 36$, and ϱ is made vary within $\{20, 100, 1 \cdot 10^3\}$. Figure 3.2 shows that a better tracking performance is obtained with $\varrho = 100$ than with $\varrho = 20$ or $\varrho = 1 \cdot 10^3$. This can be deduced from the expression of the coefficient $\beta_w(\varrho, M)$ in Eq. (3.54), as explained in paragraph 3.2.2.2. The optimal choice of the penalty

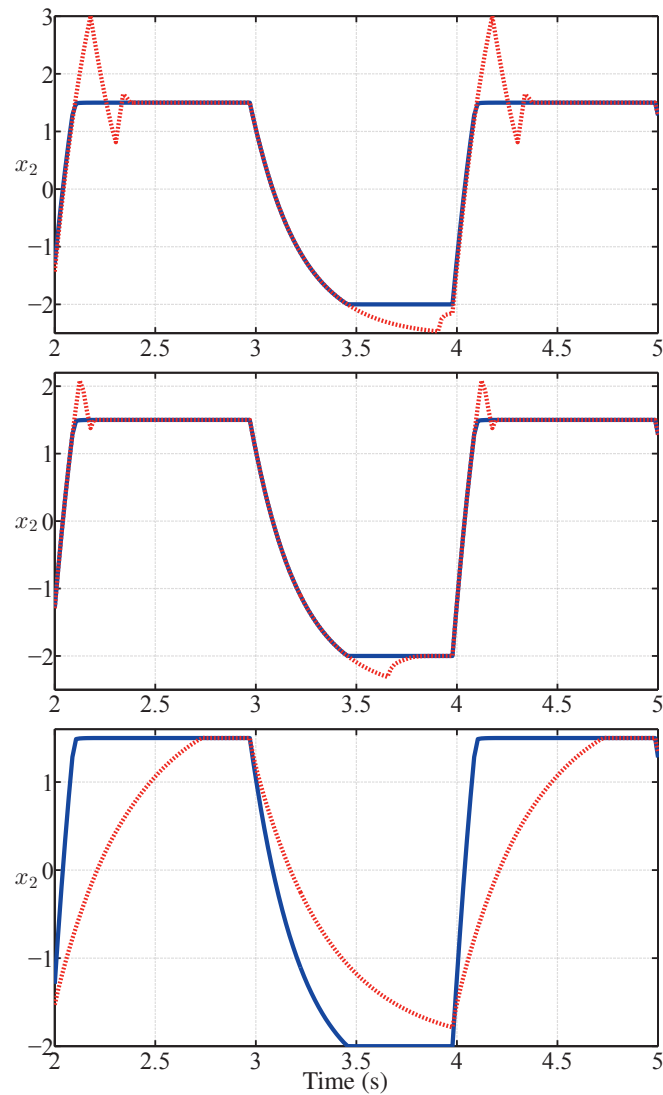


Figure 3.2: Angular speed against time for increasing penalty parameters ϱ and a fixed computation power $2 \cdot 10^3 \text{proj/sec}$: 20 (top), 100 (middle) and 1000 (bottom). The sub-optimal trajectory obtained with Algorithm 7 is plotted in dashed red, while the full NMPC trajectory obtained using IPOPT (for the same Δt) is in blue.

parameter is known to be critical to the convergence speed of ADMM, which is very similar to the optimality tracking splitting schemes of Algorithms 10 or 11, since they can be interpreted as truncated Gauss-Seidel procedures in an augmented Lagrangian. To our knowledge, this effect has only been observed for ADMM-type techniques when dealing with convex programs. When solving nonconvex programs using augmented Lagrangian techniques, it is commonly admitted that ϱ should be increased at every dual iteration in order to ensure convergence to a KKT point. Taking ϱ

too large is known to result in ill-conditioning. For Algorithm 7, the analysis is different, as ϱ does not only affect the algorithm at the level of linear algebra, but does impact the contraction of the primal-dual sequence, and thus the convergence speed over time, or tracking performance. Thus our study provides a novel interpretation of the choice of ϱ via a parametric analysis in a nonconvex framework. The effect of the optimiser step-size ϱ on the closed-loop performance fully appears in Fig. 3.6. Satisfaction of the KKT conditions of the parametric augmented Lagrangian problem

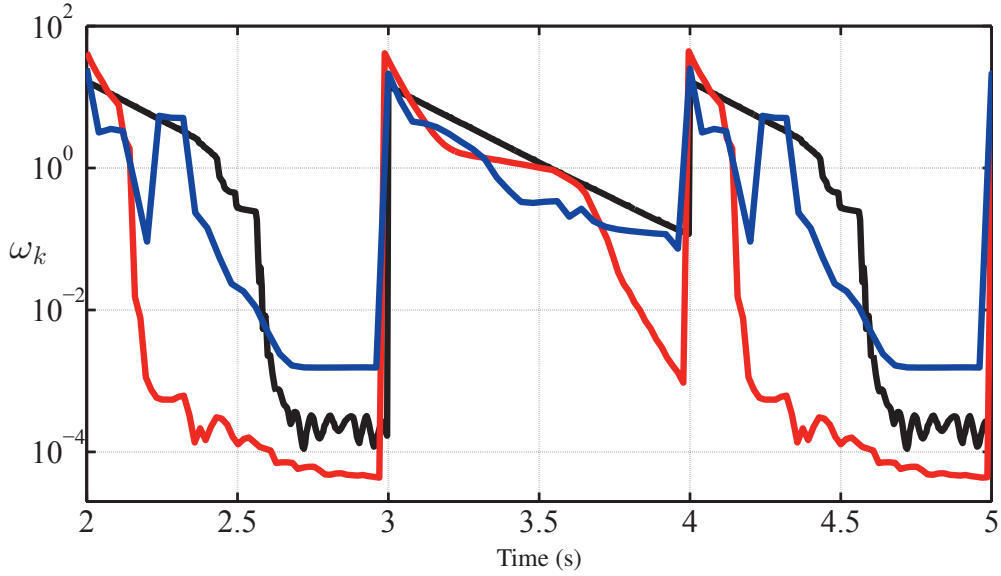


Figure 3.3: Optimality of bound constrained augmented Lagrangian program for different sampling periods Δt and a fixed computation power $2 \cdot 10^3$ proj/sec: 0.004 sec (black), 0.018 sec (red) and 0.04 sec (blue).

$$\underset{x \in B(\underline{x}, \bar{x})}{\text{minimise}} L_\varrho(x, \bar{\mu}_k, s_{k+1})$$

is measured along the closed-loop trajectory by computing

$$\omega_k := \|\pi_{B(\underline{z}, \bar{z})}(\bar{z}(\bar{\mu}_{k-1}, s_k) - \nabla L_\varrho(\bar{z}(\bar{\mu}_{k-1}, s_k), \bar{\mu}_{k-1}, s_k)) - \bar{z}(\bar{\mu}_{k-1}, s_k)\|_2,$$

which is plotted in Fig. 3.3. Over time, convergence towards low criticality values is faster for $\Delta t = 0.18$ sec, than for shorter sampling period ($\Delta t = 0.004$ sec) or larger sampling period ($\Delta t = 0.04$ sec). The same effect can be observed for the feasibility of the nonlinear equality constraints $G(\cdot, s_k)$, as pictured in Fig. 3.4.

From the results presented in Figures 3.1, 3.2, 3.3 and 3.4, one may conclude that sampling

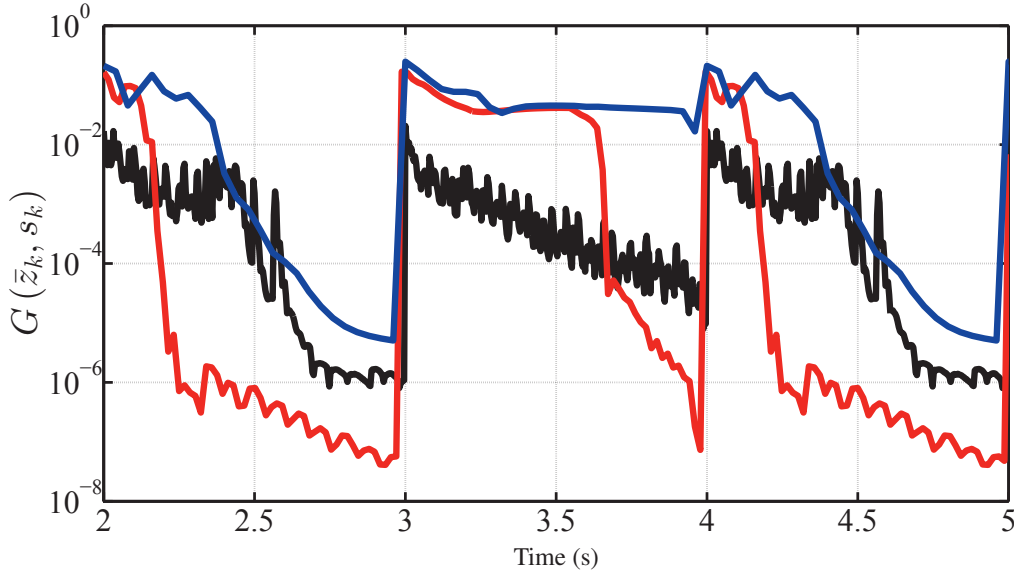


Figure 3.4: Norm of equality constraints $\|G(\bar{z}_k, s_k)\|_2$ for different sampling periods Δt and a fixed computation power $2 \cdot 10^3$ proj/sec: 0.004 sec (black), 0.018 sec (red) and 0.04 sec (blue).

faster does not necessarily result in better performance of Algorithm 7. This behaviour is confirmed by Figure 3.5. For every computational power within $\{1 \cdot 10^3, 2 \cdot 10^3, 3 \cdot 10^3, 4 \cdot 10^3\}$, the sampling period is made to vary from $\Delta t = 2 \cdot 10^{-3}$ sec to $\Delta t = 4 \cdot 10^{-2}$ sec. The tracking performance is assessed by computing the normalised L^2 -norm of the difference between the full-NMPC output trajectory obtained with IPOPT [152] and the output signal obtained with Algorithm 7 (at the same sampling period), on a fixed time interval between 2 sec and 4 sec. More precisely, the optimality tracking error is defined by

$$E := \sqrt{\frac{1}{N_s} \sum_{k=1}^{N_s} (y_k^* - \bar{y}_k)^2},$$

where $\{y_k^*\}$ is the system output signal obtained with IPOPT, $\{\bar{y}_k\}$ is the system output signal obtained with Algorithm 7 (for the same Δt) and N_s is the number of time samples. For a fast sampling rate, the error E appears to be quite large ($1 \cdot 10^0$), as the warm-starting point is close to the optimal solution but only few primal steps can be evaluated, resulting in little improvement of the initial guess in terms of optimality. This effect can even be justified further by Theorem 3.10: as the number of primal iterations M is fixed by the sampling period, the term $M^{-\psi(d_L, n)}$ in the expression of $\beta_w(\varrho, M)$ and $\beta_s(\varrho, M)$ is not sufficiently small to dampen the effect of the term $1 + \varrho\lambda_G$, and thus the contraction (3.57) becomes looser, thus degrading the closed-loop performance. As

the sampling becomes slower, more primal iterations can be carried out and subsequently, the error E is reduced. The same reasoning as before on $\beta_w(\varrho, M)$ and $\beta_s(\varrho, M)$ can be made. However, if the sampling frequency $1/\Delta t$ is too low, the initial guess is very far from the optimal point, to the point that Assumption 3.9 may not be satisfied anymore, hence the error increases again. Thus, at every computational power, an optimal sampling period is obtained. As the computation power increases, the optimal Δt appears to decrease and the associated optimality tracking error E drops.

Remark 3.20. *Note that we compare the behaviour of our parametric optimisation algorithm on NLPs of fixed dimension, no matter what the sampling period is, as the number of prediction samples has been fixed. This means that the prediction time changes as the sampling period varies, which may have an effect on the closed-loop behaviour. However, it is important to remember that the error E is measured with respect to the closed-loop trajectory under the optimal full-NMPC control law computed at the same sampling period.*

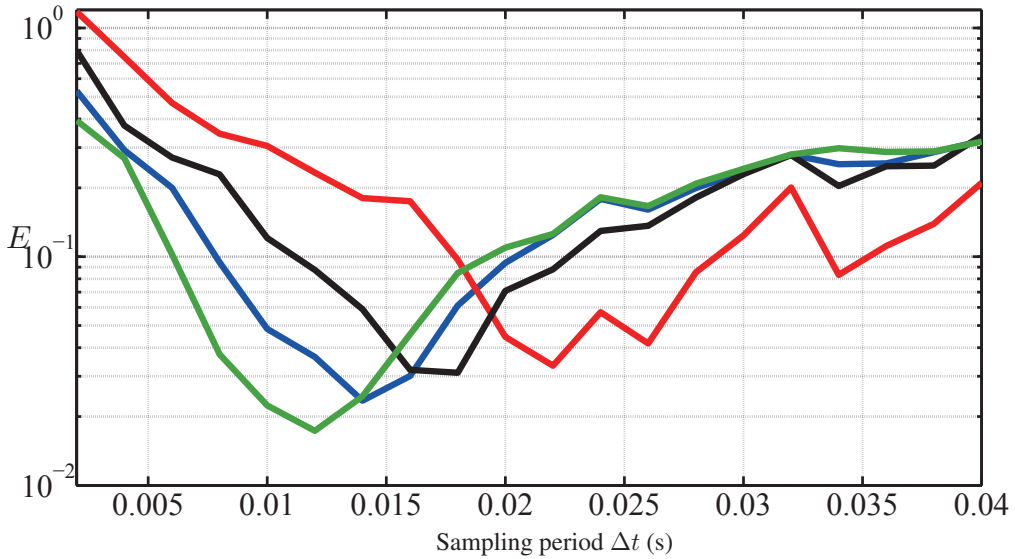


Figure 3.5: Evolution of the optimality tracking error E against sampling period for different computation power: $1 \cdot 10^3$ primal iterations per sec.(red), $2 \cdot 10^3$ (black), $3 \cdot 10^3$ (blue) and $4 \cdot 10^3$ (green).

An interesting aspect of the nonconvex splitting Algorithm 7 is that the step-size ϱ has an effect on the closed-loop behaviour of the nonlinear dynamics, as shown in Fig. 3.6. Given fixed sampling period and computational power, the tracking performance can be improved by tuning the optimiser step-size ϱ . In a sense, ϱ can now be interpreted as a tuning parameter for the NMPC controller. In particular, for a fixed number of primal iterations M , choosing ϱ too large makes

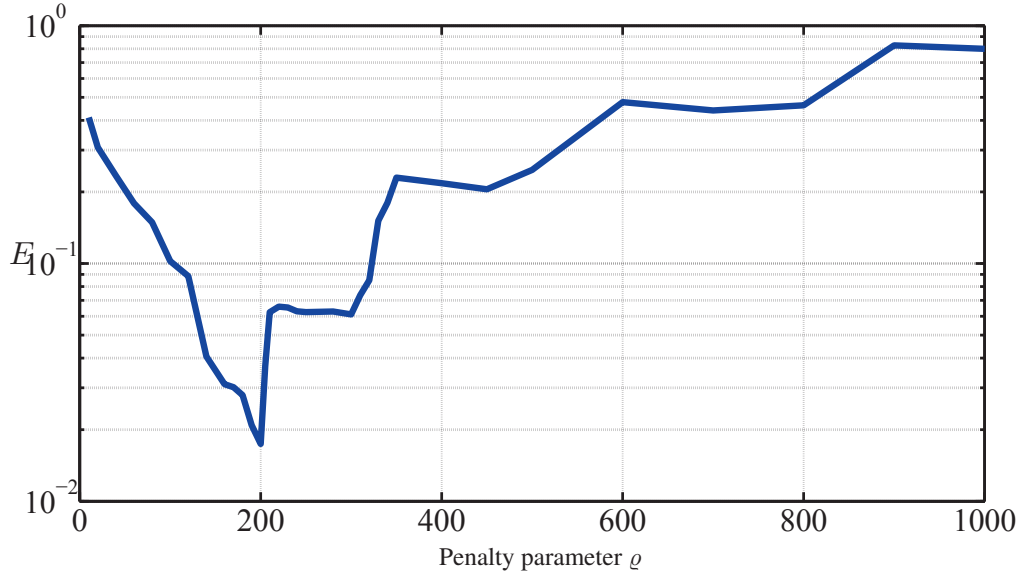


Figure 3.6: Evolution of the optimality tracking error E against penalty parameter ρ for $2 \cdot 10^3$ proj/sec and $\Delta t = 0.018$ sec.

the numerical value of the contraction coefficients $\beta_w(\rho, M)$ and $\beta_s(\rho, M)$ blow up, subsequently degrading the tracking performance. From the arguments developed in paragraph 3.2.2.3, one can reasonably expect Algorithm 12 to track the time-dependent optima more accurately than Algorithm 7. This is confirmed by Fig. 3.7.

3.3.2 Collaborative tracking of unicycles

The second example is a collaborative tracking problem based on NMPC. Three unicycles are controlled so that a leader follows a predefined path, while two followers maintain a fixed formation. This control objective can be translated into the cost function of an NMPC problem, which is then written

$$\int_0^T \left(\|x^{(1)}(t) - x^r(t)\|_{Q_1}^2 + \|u^{(1)}(t)\|_{R_1}^2 + \|u^{(2)}(t)\|_{R_2}^2 + \|u^{(3)}(t)\|_{R_3}^2 \right. \\ \left. + \|x^{(1)}(t) - x^{(2)}(t) - d_{1,2}\|_{Q_{1,2}}^2 \right. \\ \left. + \|x^{(1)}(t) - x^{(3)}(t) - d_{1,3}\|_{Q_{1,3}}^2 \right) dt,$$

where $Q_1, Q_{1,2}, Q_{1,3}, R_1, R_2, R_3$ are positive definite matrices, $d_{1,2}, d_{1,3}$ are vectors that define the formation between unicycles 1, 2 and 3 and x^r is a reference path. All agents 1, 2 and 3 follow the

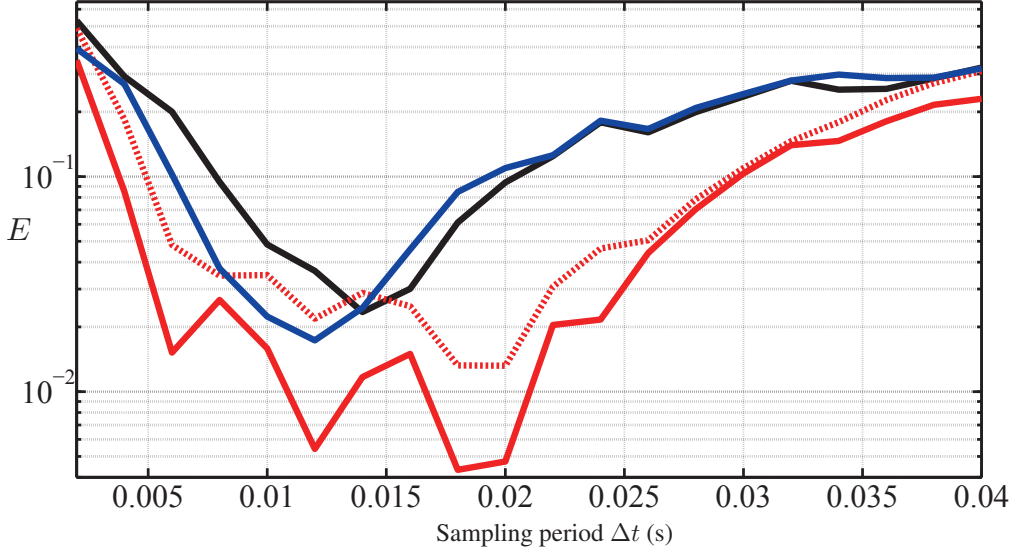


Figure 3.7: Evolution of the optimality tracking error E against sampling period Δt . Algorithm 7 for $3 \cdot 10^3$ proj/sec in black, for $4 \cdot 10^3$ proj/sec in blue. Algorithm 12 with 3 homotopy steps for $3 \cdot 10^3$ proj/sec in dashed red, with 4 homotopy steps for $4 \cdot 10^3$ proj/sec in red.

standard unicycle dynamics

$$\begin{cases} \dot{x}_1 = u_1 \cos x_3 \\ \dot{x}_2 = u_1 \sin x_3 \\ \dot{x}_3 = u_2 \end{cases},$$

subject to input constraints

$$u_1 \in [0, 0.5] \quad , \quad u_2 \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] .$$

The continuous-time NMPC problem is discretised using a Runge-Kutta integrator of order 4 [78], while the closed-loop system is simulated with the MATLAB adaptive step-size integrator `ode45`. In the resulting finite-dimensional NLP, two cost coupling terms appear between agents 1 and 2, as well as agents 1 and 3. This can be addressed by the splitting Algorithm 7. Moreover, the whole procedure then consists in a sequence of alternating steps between agent 1 and the group $\{2, 3\}$, which can compute descent steps in parallel without requiring any communication. For this particular NLP with cost-couplings, the dual updates can be performed in parallel. Results of the collaborative tracking NMPC are presented in Figures 3.8 and 3.9. The number of iterations/communications

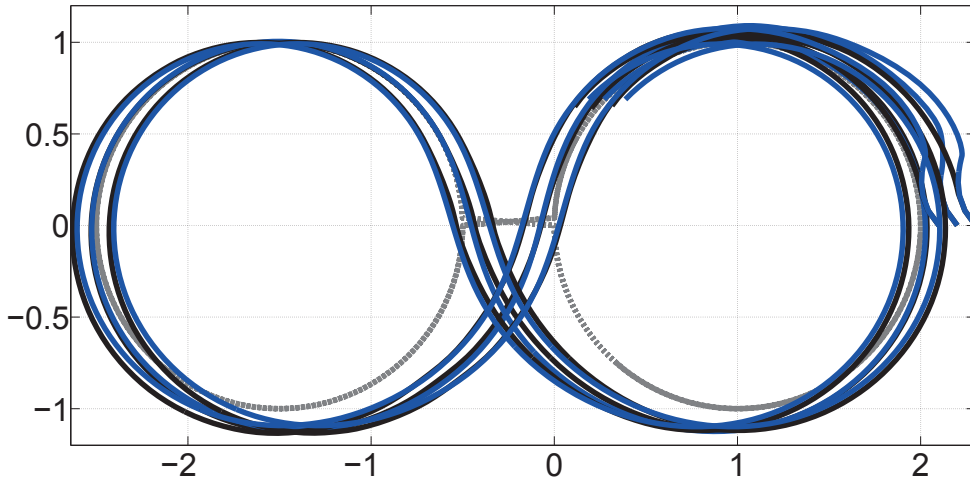


Figure 3.8: Trajectories of the three-unicycles formation for 300 proj/sec , $\Delta t = 0.20 \text{ sec}$ and $\varrho = 3 \cdot 10^3$.

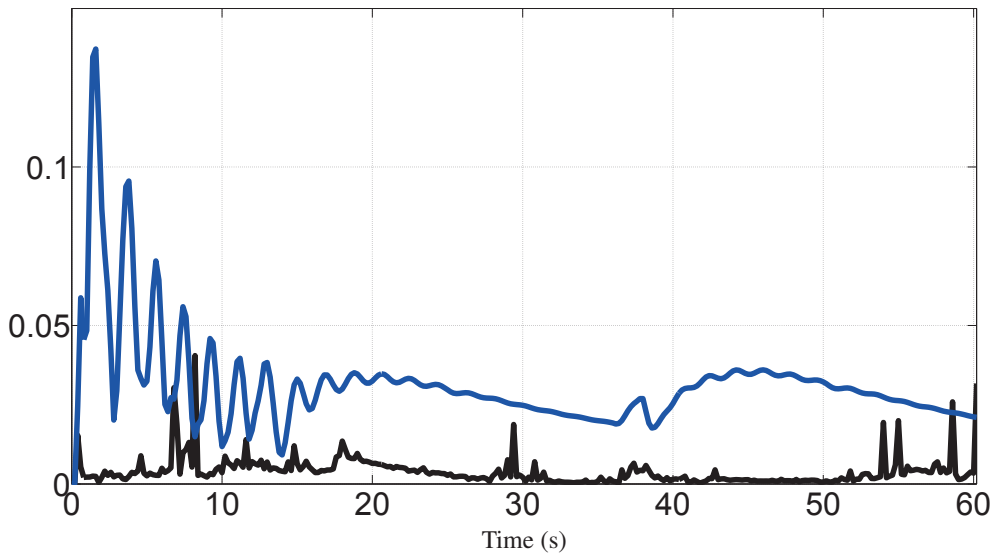


Figure 3.9: Evolution of the formation error between unicycles 1 and 2 for Algorithm 7 (blue), compared with the formation error obtained with the full NMPC (IPOPT, black).

per second has been fixed at 300 and the sampling period set to $\Delta t = 0.20 \text{ sec}$. Within the sampling period, this results in $M = 60$ exchanges of packets between agent 1 and agents 2, 3, which per-

form their computations in parallel. The penalty parameter was $\varrho = 3 \cdot 10^3$. The formation-keeping NMPC has been first simulated with the unicycles in closed-loop with the full-NMPC control law, computed using IPOPT with accuracy $1 \cdot 10^{-7}$, which is purely centralised, hence not very interesting from a practical point of view, in this particular case. The full-NMPC trajectory is plotted in black in Fig. 3.8, while the one obtained using Algorithm 7 is represented in blue. The closed-loop formation error

$$\epsilon_{1,2} := \|x^{(1)} - x^{(2)} - d_{1,2}\|_2$$

is plotted in Fig. 3.9. At every reference change, the error rises, but decreases again as the tracking converges. The performance could be further improved by tuning the penalty ϱ or performing a few homotopy steps as in Algorithm 12.

Chapter 4

Applications in Optimal Control

Two applications of the parametric algorithms studied in Chapter 3 are presented. The first example consists of the parametric optimal control problem as it arises in NMPC. In order to turn the infinite-dimensional problem into a finite-dimensional NLP that can be solved by the TRAP algorithm introduced in Chapter 2 for instance, the so-called direct approaches resort to different discretisation techniques. In a collocation scheme, the state trajectory and its approximation are collocated at quadrature points, which are chosen to minimise the integral of the residuals between the approximate and the true state profile [133, 98], thus resulting in a small discretisation error. The equality constraints are written in terms of the approximate state profile and interpolation polynomials, so that the continuous-time OCP is directly turned into a large and sparse NLP. It can be shown that collocations are implicit Runge-Kutta integrators. They are thus A-stable and recommended for stiff systems. On the contrary, in a shooting scheme, equality constraints are evaluated by means of explicit or implicit integrators, which has a tendency to increase the nonlinearity in the problem and induce ill-conditioning. Interesting discussions on the advantages and drawbacks of the different discretisation methods in NMPC can be found in [160, 47]. In the remainder, we present a multiple shooting strategy based on the method of multipliers. In the static case, it is a tailored implementation of the LANCELOT algorithm [34] for solving the partially separable NLP resulting from the multiple shooting discretisation. Regarding online NMPC, we resort to the parametric augmented Lagrangian algorithm analysed in Chapter 3. In the context of decomposition strategies for optimal control, it is worth pointing out that the evaluation of the multiple shooting constraints can be easily parallelised.

Our second example is the multi-stage AC-OPF problem, in which different AC-OPF problems are coupled in time via dynamical storage systems. The purpose of solving the AC-OPF is to compute power set-points for all generating units in a power network so as to minimise a specific operating cost. Solving the AC-OPF is challenging, as the network model yields nonlinear equality constraints, which make the NLP nonconvex. Thus, computing a global optimum in the

case of large networks is almost intractable, although some advances have recently been made via semidefinite relaxations [108], moment relaxations [97] or second-order cone relaxations [64]. In practice, nonlinear solvers are widely used in order to determine local minima that satisfy the nonlinear power flow constraints [164]. However, as claimed in Chapter 2, as the AC-OPF can be extremely large (10,000 nodes), distributed optimisation techniques become highly relevant, as shown in [105]. Motivated by the work of [69], we consider the optimal power flow problem over distribution networks with storage elements at buses. This leads to a finite-horizon optimal control problem, which we are interested in solving via distributed optimisation techniques in a real-time context. This is motivated by the high sampling rates needed by recent applications in power systems [15], which entail truncating the iterations of a large-scale solver in order to meet the time requirements and reduce latency.

4.1 A Direct Optimal Control Algorithm Based on Augmented Lagrangian

In this section, a novel multiple shooting algorithm based on an augmented Lagrangian technique is presented. It differs quite significantly from the initial approach of [19], in which sequential quadratic programming (SQP) is applied to solve the NLP resulting from the multiple shooting discretisation. In fact, it is worth noting that all subsequent development on multiple shooting has always been based on SQP [109, 101], especially in an online setting [91]. This is probably justified by the fact that SQP methods are particularly efficient on NLPs, in which the level of nonlinearities in the constraints is high. They also benefit from local superlinear convergence when a quasi-Newton approximation is applied [119]. However, despite recent progress [27, 40], the convergence of SQP methods is not robust to inexact solutions of the subproblems, which is required for large-scale or distributed optimisation. In optimal control, the quadratic programming subproblems are highly structured. Hence, efficient resolution techniques, such as the dual Newton strategy implemented in QPDUNES, can be applied to reduce the computation time on large-scale problems [61], but computational results are lacking in terms of global efficiency of the resulting SQP algorithm and its scalability properties have not been assessed with respect to existing large-scale interior-point solvers for instance. Besides, the line-search globalisation, which is a core ingredient in the existing SQP software, is not advisable in a distributed setting, due to its high cost in terms of communications.

On the contrary, augmented Lagrangian methods are well-suited to large-scale or distributed programs. This is mainly due to the fact that they can be implemented matrix-free and that their convergence is not hampered by inexact solutions of the subproblems [34, 57]. These two features

are also valuable for real-time optimisation, where early termination is often required to satisfy hard time constraints. Moreover, in Chapter 3, in the context of online distributed optimisation, contraction of the iterates yielded by a parametric augmented Lagrangian scheme has been established and expressed as a function of the number of iterations and the penalty coefficient. To our knowledge, such an analysis has not been carried out for online distributed SCP methods. Thus, it is still not very clear whether a splitting technique such as ADMM could be terminated at an early stage when solving a parametric distributed convex QP, while ensuring stability of the suboptimality error. In the context of multiple shooting, augmented Lagrangian approaches are also attractive, since evaluating the gradient of the augmented Lagrangian is less costly than computing the jacobian of the shooting constraints, as shown next. However, augmented Lagrangian techniques suffer from a slower local convergence rate (linear or superlinear) than SQP techniques (superlinear or quadratic). The first-order dual update is also a weakness, as the process can be driven towards infeasible points when applied to solve difficult nonlinear problems. In contradiction with the disappointing outlook on augmented Lagrangian algorithms in optimal control given by [60], we show that the parametric algorithm described in Chapter 3 is very promising and competitive with the state-of-the-art in real-time NMPC.

4.1.1 The optimal control problem and its multiple shooting discretisation

The problem we consider is that of finding state and input profiles $(x^*(\cdot), u^*(\cdot))$ satisfying the necessary conditions of optimality of the optimal control problem

$$\begin{aligned} & \underset{u(\cdot)}{\text{minimise}} \int_0^T l(x(t), u(t)) dt & (4.1) \\ & \text{s.t. } x(0) = \hat{x}_0, \\ & \dot{x}(t) = f(x(t), u(t)), \\ & \forall t \in [0, T], x(t) \in \mathcal{X}, u(t) \in \mathcal{U}, \end{aligned}$$

where $T \in]0, +\infty[$, $f : \mathbb{R}^{n_x \times n_u} \rightarrow \mathbb{R}^{n_x}$ and $l : \mathbb{R}^{n_x \times n_u} \rightarrow \mathbb{R}$ are continuously differentiable on $\mathcal{X} \times \mathcal{U}$ and $\hat{x}_0 \in \mathbb{R}^{n_x}$, which stands for a state estimate. The sets \mathcal{X} and \mathcal{U} are assumed to be box constraint sets. For clarity of exposition, we do not consider any Mayer term and terminal constraint. Path constraints $g(x(t), u(t)) \leq 0$ could be incorporated into problem (4.1), but we discard them for simplicity.

In order to transform the OCP (4.1) into a finite-dimensional NLP, direct methods proceed by parameterising the continuous control profile $u(\cdot)$ using a finite number of parameters, whose optimal values can be computed by means of a nonlinear solver. Similarly to [109], we resort to a piece-

wise constant parameterisation of the control profile, which is written at all time instants $t \in [0, T]$,

$$u(t) = \sum_{i=0}^{N-1} q_i l_{[t_i, t_{i+1}]}(t) \quad ,$$

where $N \geq 1$, $\{q_i\}_0^{N-1} \subset \mathbb{R}^{n_u}$ and the mesh $\{t_i\}_{i=0}^N \subset [0, T]$ is such that $t_0 := 0$ and $t_N := T$. In order to parameterise the state profile $x(\cdot)$, shooting nodes $\{s_i\}_{i=0}^N \subset \mathbb{R}^{n_x}$ and shooting constraints are introduced for each interval $[t_i, t_{i+1}]$ as follows

$$s_{i+1} - x(t_{i+1}; s_i, q_i) = 0, \quad i \in \{0, N-1\} \quad , \quad (4.2)$$

where $x(\cdot; s_i, q_i)$ is the solution of the boundary value problem

$$\begin{cases} \forall t \in [t_i, t_{i+1}], \dot{x}(t) = f(x(t), q_i) \\ x(t_i) = s_i \end{cases} \quad (4.3)$$

The role of the shooting constraints is to ensure continuity of the state profile at the ends of every shooting interval. The objective of the OCP (4.1) is also subdivided according to the mesh $\{t_i\}_{i=0}^N$ as follows

$$\begin{aligned} \int_0^T l(x(t), u(t)) dt &= \sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} l(x(t), u(t)) dt \\ &= \sum_{i=0}^{N-1} y(t_{i+1}; s_i, q_i) \quad , \end{aligned}$$

where $y(\cdot; s_i, q_i)$ is the solution of the boundary value problem

$$\begin{cases} \forall t \in [t_i, t_{i+1}], \dot{y}(t; s_i, q_i) = l(x(t; s_i, q_i), q_i) \\ y(t_i; s_i, q_i) = 0 \end{cases} \quad (4.4)$$

which is coupled with (4.3) via the state $x(\cdot; s_i, q_i)$. Finally, the NLP resulting from the multiple-

shooting discretisation is

$$\begin{aligned}
 & \underset{\{s_i\}_{i=0}^N, \{q_i\}_{i=0}^{N-1}}{\text{minimise}} \quad \sum_{i=0}^{N-1} y(t_{i+1}; s_i, q_i) \\
 & \text{s.t. } s_0 - \hat{x}_0 = 0, \\
 & \quad s_{i+1} - x(t_{i+1}; s_i, q_i) = 0, \\
 & \quad s_i \in \mathcal{X}, q_i \in \mathcal{U}, i \in \{0, \dots, N-1\} \quad ,
 \end{aligned} \tag{4.5}$$

where for all $i \in \{0, \dots, N-1\}$, $x(\cdot; s_i, q_i)$ and $y(\cdot; s_i, q_i)$ are solutions of the following augmented boundary value problem

$$\begin{cases} \forall t \in [t_i, t_{i+1}], \dot{v}(t; s_i, q_i) = F(v(t; s_i, q_i), q_i) := \begin{pmatrix} f(x(t; s_i, q_i), q_i) \\ l(x(t; s_i, q_i), q_i) \end{pmatrix} \\ v(t_i; s_i, q_i) = \begin{pmatrix} s_i \\ 0 \end{pmatrix} . \end{cases} \tag{4.6}$$

where the augmented state is denoted by

$$v(t; s_i, q_i) := \begin{pmatrix} x(t; s_i, q_i) \\ y(t; s_i, q_i) \end{pmatrix} \in \mathbb{R}^{n_x+1} .$$

It is worth noting that NLP 4.5 has a partially separable structure, as the shooting node s_i is only coupled with nodes s_{i-1} and s_{i+1} . The primal optimiser of NLP (4.5) is defined by

$$z := (s_0^\top, q_0^\top, \dots, s_{N-1}^\top, q_{N-1}^\top, s_N^\top)^\top \in \mathbb{R}^{N(n_x+n_u)+n_x} , \tag{4.7}$$

and the dual optimiser associated with the equality constraints, as

$$\mu := (\mu_{0,1}, \dots, \mu_{0,n_x}, \dots, \mu_{N,1}, \dots, \mu_{N,n_x})^\top \in \mathbb{R}^{(N+1)n_x} . \tag{4.8}$$

The primal box constraint set corresponding to variable z is denoted by

$$\mathcal{Z} := \mathcal{X} \times \mathcal{U} \times \dots \times \mathcal{U} \times \mathcal{X} . \tag{4.9}$$

4.1.2 The augmented Lagrangian algorithm

Instead of linearising the shooting constraint (4.2) with respect to the shooting node s_i and control q_i as in [109, 91], we relax it by means of an augmented Lagrangian penalty and thus introduce

$$\mathcal{L}_\varrho(z, \mu, \hat{x}_0) := \left(\mu_0 + \frac{\varrho}{2} (s_0 - \hat{x}_0) \right)^\top (s_0 - \hat{x}_0) + \sum_{i=1}^N L_\varrho(v(t_i; s_{i-1}, q_{i-1}), s_i, \mu_i) \quad , \quad (4.10)$$

where the local augmented Lagrangian is defined by

$$L_\varrho(v(t; s, q), s', \nu) := y(t; s, q) + \left(\nu + \frac{\varrho}{2} (s' - x(t; s, q)) \right)^\top (s' - x(t; s, q)) \quad ,$$

for $s, s' \in \mathbb{R}^{n_x}$, $q \in \mathbb{R}^{n_u}$, $\nu \in \mathbb{R}^{n_x}$ and $\varrho > 0$.

Remark 4.1. *One could use different penalty coefficients for each of the shooting intervals. Increasing the penalty has a tendency to cause numerical difficulties. Hence, smaller penalties could be applied where tight satisfaction of the shooting constraints may not be necessary.*

The augmented Lagrangian algorithm has already been presented in Chapter 2 and 3. Therefore, we briefly recall the main phases without going into details.

4.1.2.1 Dual updates

The dual variables associated with the shooting constraints are updated in a first-order fashion at very iteration k of an outer loop

$$\mu_i^{k+1} = \mu_i^k + \varrho^k (s_i^k - x(t_i; s_{i-1}^k, q_{i-1}^k)) \quad , \quad i \in \{1, \dots, N\} \quad (4.11)$$

and

$$\mu_0^{k+1} = \mu_0^k + \varrho^k (s_0^k - \hat{x}_0) \quad , \quad (4.12)$$

where the shooting nodes $\{s_i^k\}_{i=0}^N$ and inputs $\{q_i^k\}_{i=0}^N$ are obtained by computing an approximate critical point of the bound-constrained augmented Lagrangian subproblem

$$\underset{\{s_i\}_{i=0}^N, \{q_i\}_{i=0}^{N-1}}{\text{minimise}} \quad \left(\mu_0^k + \frac{\varrho^k}{2} (s_0 - \hat{x}_0) \right)^\top (s_0 - \hat{x}_0) + \sum_{i=1}^N L_{\varrho^k}(v(t_i; s_{i-1}, q_{i-1}), s_i, \mu_i^k) \quad (4.13)$$

s.t. $s_i \in \mathcal{X}$, $q_i \in \mathcal{U}$, $i \in \{0, \dots, N-1\}$

$s_N \in \mathcal{X}$

and the penalty ϱ^k is increased. As explained in Chapter 2, the accuracy of the first-order optimality conditions in (4.13) at z^k is tightened after every outer iteration k .

The outer loop that we have just described only ensures local convergence of the primal-dual sequence $\left\{ \left((z^k)^\top, (\mu^k)^\top \right)^\top \right\}$ to a KKT point of (4.5). Global convergence guarantees can be obtained by adapting the dual update to the level of satisfaction of the shooting constraints [34]. This adaptive update scheme has been successfully applied in the LANCELOT software [35]. More precisely, given a positive tolerance ϵ^k , if

$$\|s_0^k - \hat{x}_0\|_2^2 + \sum_{i=1}^N \|s_i^k - x(t_i; s_{i-1}^k, q_{i-1}^k)\|_2^2 \leq (\epsilon^k)^2, \quad (4.14)$$

then the dual updates (4.11) and (4.12) are performed, the penalty ϱ^k remains unchanged and the tolerance on feasibility ϵ^k as well as the tolerance on optimality in (4.13) are shrunk. If condition (4.14) is not satisfied, roughly speaking if the gradient of a local dual function is not sufficiently accurate, then the penalty ϱ^k is increased in order to drive the process towards feasibility at the next outer iteration.

In an online NMPC context where a fixed number of iterations is required, only one dual update is computed once a new state estimate \tilde{x}_0 is available. Given the primal-dual warm-start

$$\bar{w}(\hat{x}_0) := \left(\bar{s}_0(\hat{x}_0)^\top, \bar{q}_0(\hat{x}_0)^\top, \dots, \bar{s}_{N-1}(\hat{x}_0)^\top, \bar{q}_{N-1}(\hat{x}_0)^\top, \bar{s}_N(\hat{x}_0), \bar{\mu}_0(\hat{x}_0)^\top, \dots, \bar{\mu}_N(\hat{x}_0)^\top \right)^\top,$$

the suboptimal primal point $\bar{z}(\tilde{x}_0)$ is obtained after M iterations of a descent algorithm applied to the augmented Lagrangian subproblem (4.13) at \tilde{x}_0 , while the dual point $\bar{\mu}(\tilde{x}_0)$ is computed in the following way:

$$\bar{\mu}_i(\tilde{x}_0) = \bar{\mu}_i(\hat{x}_0) + \varrho(s_i(\tilde{x}_0) - x(t_i; s_{i-1}(\tilde{x}_0), q_{i-1}(\tilde{x}_0))), \quad i \in \{1, \dots, N\}, \quad (4.15)$$

and

$$\bar{\mu}_0(\tilde{x}_0) = \bar{\mu}_0(\hat{x}_0) + \varrho(s_0(\tilde{x}_0) - \tilde{x}_0), \quad i \in \{1, \dots, N\}, \quad (4.16)$$

given a fixed penalty ϱ .

4.1.2.2 Primal updates

Remarkably, the partially separable structure of problem (4.13) makes it suitable for alternating minimisations, either over the entire loop, as in Algorithm 10, or only for activity detection, as in Algorithm 11. This is illustrated in Fig. 4.1 below. In the static case, similarly to [35], a trust

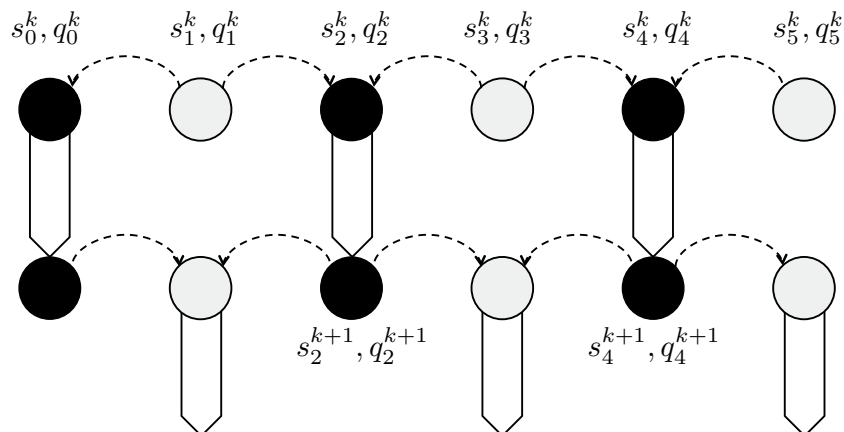


Figure 4.1: Scheme of alternating minimisation in the augmented Lagrangian subproblem resulting from the multiple-shooting discretisation. Two sets of parallel steps are required for updating all shooting nodes and inputs.

region quasi-Newton method is applied to find an approximate critical point of the augmented Lagrangian. In the dynamic case, where \hat{x}_0 is updated at every time instant, the trust region iterations are cut after a fixed count, along the lines of Algorithm 9.

Remark 4.2. *It is worth pointing out that the first-order methods 8 and 10 could be applied in the context of multiple shooting, but their performance is likely to be worsened by the ill-conditioning of NLP (4.5), due to the shooting constraints that are evaluated by integration of the nonlinear dynamics.*

In the context of multiple-shooting, computing gradients is generally done via sensitivity analysis, as explained next. However, obtaining second-order information can be computationally very expensive. Therefore, we resort to a quasi-Newton scheme. In order to take advantage of the partial separability of the problem, we resort to Symmetric Rank One (SR1) updates. More precisely, our goal is to have a quasi-Newton scheme, in which the blockwise structure of the hessian of the augmented Lagrangian is preserved, as depicted in Fig. 4.2 below. In the exact hessian, the overlapping (grey) blocks may be even sparser. The partially separable augmented Lagrangian (4.10) can be rewritten

$$\begin{aligned} \mathcal{L}_\rho(z, \mu, \hat{x}_0) &= \sum_{i=0}^{N-1} \phi_i(s_i, q_i, s_{i+1}) \\ &= \sum_{i=0}^{N-1} \phi_i(E_i z) \quad , \end{aligned}$$

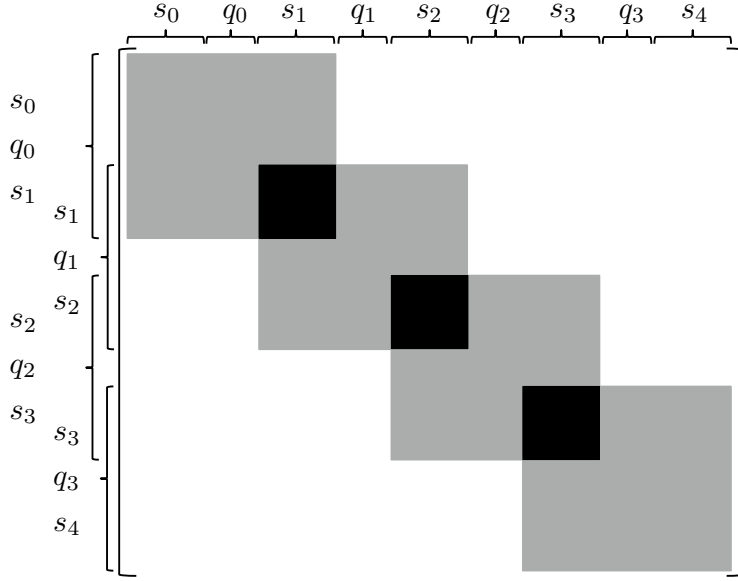


Figure 4.2: Sparsity structure of the hessian of the augmented Lagrangian.

where the group functions ϕ_i are defined by

$$\phi_i(s_i, q_i, s_{i+1}) := \begin{cases} \left(\mu_0 + \frac{\rho}{2}(s_0 - \hat{x}_0)\right)^\top (s_0 - \hat{x}_0) + L_\rho(v(t_1; s_0, q_0), s_1, \mu_1), & \text{if } i = 0, \\ L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}), & \text{if } i \in \{1, \dots, N-1\}, \end{cases}$$

and given $i \in \{0, \dots, N-1\}$, the matrix $E_i \in \mathbb{R}^{(2n_x+n_u) \times (N(n_x+n_u)+n_x)}$ is

$$E_i = \begin{bmatrix} 0_{(2n_x+n_u) \times (n_x+n_u)i} & I_{(2n_x+n_u) \times (2n_x+n_u)} & 0_{(2n_x+n_u) \times (N-i-1) \cdot (n_x+n_u)} \end{bmatrix}.$$

Hence, the hessian of the multiple-shooting augmented Lagrangian is given by

$$\nabla_{z,z}^2 \mathcal{L}_\rho(z, \mu, \hat{x}_0) = \sum_{i=0}^{N-1} E_i^\top \nabla^2 \phi_i(E_i z) E_i.$$

Each of the Hessians $\nabla^2 \phi_i(E_i z)$ corresponds to the group of shooting variables $\{s_i, q_i, s_{i+1}\}$. For the remainder, for each $i \in \{0, \dots, N-1\}$, we introduce a variable

$$r_i := (s_i^\top, q_i^\top, s_{i+1}^\top)^\top. \quad (4.17)$$

Instead of computing an SR1 estimate of the full hessian $\nabla_{z,z}^2 L_\varrho(z, \mu, \hat{x}_0)$, we perform quasi-Newton approximations of the group matrices $\nabla^2 \phi_i(r_i)$ separately for $i \in \{0, \dots, N-1\}$. If the quasi-Newton matrix at the shooting group i , corresponding to variables $\{s_i, q_i, s_{i+1}\}$, at iteration k of Algorithm 9 or 11 is denoted by B_i^k , the SR1 update is

$$B_i^{k+1} = \begin{cases} B_i^k & \text{if } \langle \theta_i^k, s_i^k \rangle \leq \beta \|\theta_i^k\|_2 \|s_i^k\|_2 \\ B_i^k + \frac{\theta_i^k (\theta_i^k)^\top}{\langle \theta_i^k, s_i^k \rangle} & \text{otherwise,} \end{cases} \quad (4.18)$$

where $\beta \in (0, 1)$ and

$$\begin{aligned} s_i^k &:= r_i^{k,+} - r_i^k \\ \theta_i^k &:= y_i^k - B_i^k s_i^k \\ y_i^k &:= \nabla \phi_i(r_i^{k,+}) - \nabla \phi_i(r_i^k), \end{aligned}$$

with $r_i^{k,+}$ corresponding to the candidate point in the trust region loop, which is generated as an inexact solution of the trust region subproblem. The SR1 update (4.18) itself requires

$$N(2n_x + n_u)(3(2n_x + n_u) + 2) + N(3(2n_x + n_u) + 2(2n_x + n_u)^2)$$

floating point operations, instead of

$$(N(n_x + n_u) + n_x)(3(N(n_x + n_u) + n_x) + 2)$$

floating point operations if it was computed on the full matrix. Making use of partial separability in the rank-one updates is advantageous in terms of complexity, but also produces more accurate hessian estimates.

Remark 4.3. *The SR1 update (4.18) is computed at every iteration (successful or unsuccessful) of the trust region loop in Algorithm 9 or 11, as recommended in [119]. To obtain fast local convergence, the model has to be improved along the failed directions, otherwise candidates could again be generated in these directions, thus preventing superlinear convergence. In the context of multiple-shooting, this means that the sensitivity analysis presented next is carried out at every iteration. Therefore, it needs to be computationally efficient, otherwise the performance of the algorithm may be deeply impacted.*

Subsequently, the quasi-Newton approximation of the hessian $\nabla_{z,z}^2 \mathcal{L}_\varrho(z, \mu, \hat{x}_0)$ is

$$B = \sum_{i=0}^{N-1} E_i^\top B_i E_i .$$

However, in our implementation, the matrix B is only assembled for computing a preconditioner, since it is used in conjugate gradient (CG) iterations, which are based on structured matrix-vector products, as addressed in the next paragraph.

4.1.2.3 Solving the trust region subproblem

The most computationally expensive step in the trust region algorithm is the refinement phase, in which the model function is minimised on the null space of the active constraints at the Cauchy point, as follows

$$\begin{aligned} & \underset{p}{\text{minimise}} \quad \langle \nabla_z^\sigma L_\varrho(z, \mu, \hat{x}_0), Zp \rangle + \frac{1}{2} \langle p, Z^\top B_\sigma Z p \rangle & (4.19) \\ & \text{s.t. } z + Zp \in \mathcal{Z} \\ & \quad \|Zp\|_\infty \leq \gamma \Delta , \end{aligned}$$

where the rows of Z are an orthonormal basis of the null space of the active constraints at the Cauchy point z^C , $\Delta > 0$ is the trust region radius, σ and γ are positive scalars, the constraints set \mathcal{Z} is defined in (4.9) and

$$\nabla_z^\sigma L_\varrho(z, \mu, \hat{x}_0) := \nabla_z L_\varrho(z, \mu, \hat{x}_0) - \sigma(z - x), \quad B_\sigma := B + \frac{\sigma}{2} I .$$

As explained in Chapter 2, the trust region subproblem (4.19) is solved approximately by means of PCG iterations initialised at 0, which ensure a decrease of the model function. At every PCG iteration, the most costly operation is the matrix-vector product against the reduced quasi-Newton approximation $Z^\top B_\sigma Z$, which is represented in Fig. 4.3. In order to make use of the block structure of $Z^\top B_\sigma Z$, one has to know the indices of the free variables in every shooting group $\{s_i, q_i, s_{i+1}\}$. The worst-case complexity of the structured matrix-vector product is when all variables are free and is equal to

$$2(2n_x + n_u)^2 N + (2n_x + n_u) N .$$

Before starting the PCG iterations, a preconditioner is built from the reduced quasi-Newton matrix $Z^\top B_\sigma Z$. In the remainder, it is denoted by P and is a positive definite matrix. The re-

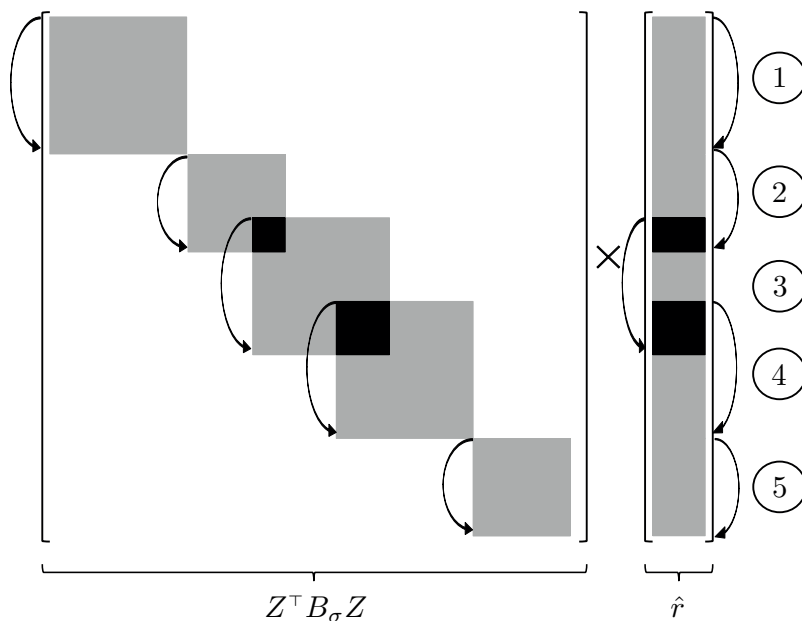


Figure 4.3: Structured product against reduced quasi-Newton hessian approximation. The size of each block depends of the number of free variables at the corresponding shooting group.

duced quasi-Newton matrix $Z^T B_\sigma Z$ has a block-diagonal structure, as shown in Fig. 4.3 for instance. Therefore, one can expect banded preconditioners to be reasonably efficient. The construction of the banded preconditioner P from the matrix $Z^T B_\sigma Z$ is depicted in Fig. 4.4. A certain number of diagonal bands is extracted from $Z^T B_\sigma Z$ and stored in P , as shown in Fig. 4.4. It is worth noting that the number of bands does not need to be larger than $2n_x + n_u$, which would correspond to having the full hessian as preconditioner. Thus, the preconditioner P is a symmetric matrix stored in band format. As the preconditioner P appears in the CG iterations via linear systems of the form

$$Pv = w, \quad (4.20)$$

we use an LDL factorisation of P to solve (4.20), that is

$$P = LDL^T,$$

where L is a lower triangular matrix with 1 on the diagonal and D is a diagonal matrix with positive elements. The LDL factorisation is stored in band format with the matrix D in the first band

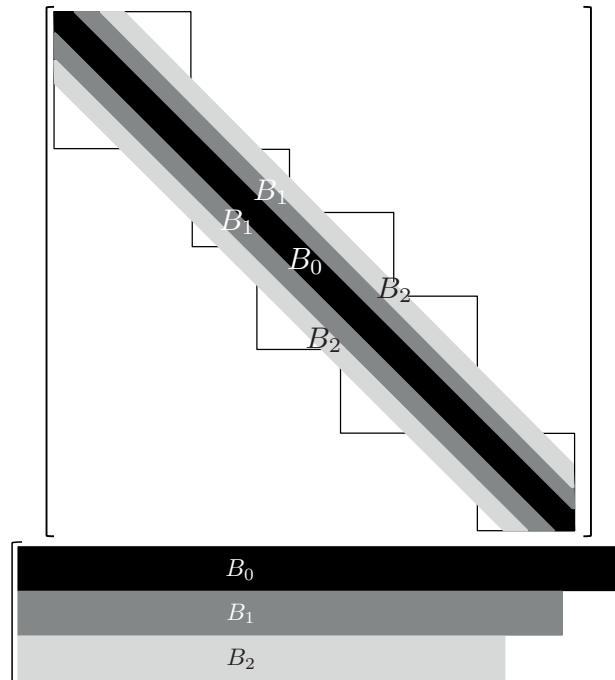


Figure 4.4: Construction of the banded preconditioner.

and the matrix L in the lower bands. In order to make the preconditioner P positive definite, a Gershgorin modification [119] is applied when computing the LDL factorisation. It consists in perturbing the diagonal D so that the Gershgorin disks lie in the positive half of the real line. The solution of (4.20) is then computed via a forward-backward solve in band format.

4.1.2.4 Gradient generation

The main advantage of introducing the augmented Lagrangian relaxation of the shooting constraints instead of merely linearising around the current iterate becomes clear when it comes to the sensitivity generation. In fact the gradient of the augmented Lagrangian (4.10) with respect to the

primal shooting variable z is given by

$$\nabla_z \mathcal{L}_\rho(z, \mu, \hat{x}_0) = \begin{pmatrix} \mu_0 + \frac{\rho}{2} (s_0 - \hat{x}_0) + \nabla_{s_0} L_\rho(v(t_1; s_0, q_0), s_1, \mu_1) \\ \nabla_{q_0} L_\rho(v(t_1; s_0, q_0), s_1, \mu_1) \\ \nabla_{s_1} L_\rho(v(t_1; s_0, q_0), s_1, \mu_1) + \nabla_{s_1} L_\rho(v(t_2; s_1, q_1), s_2, \mu_2) \\ \vdots \\ \nabla_{s_i} L_\rho(v(t_i; s_{i-1}, q_{i-1}), s_i, \mu_i) + \nabla_{s_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \\ \nabla_{q_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \\ \vdots \\ \nabla_{s_N} L_\rho(v(t_N; s_{N-1}, q_{N-1}), s_N, \mu_N) \end{pmatrix}. \quad (4.21)$$

In the expression of the gradient of the multiple-shooting augmented Lagrangian (4.21), for every shooting block i , the gradients with respect to shooting node s_i and control q_i

$$\nabla_{s_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \quad \text{and} \quad \nabla_{q_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \quad (4.22)$$

appear. The most natural way to compute the gradients in (4.22) is to apply the chain rule, which leads to

$$\begin{aligned} \nabla_{s_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) &= \nabla_{s_i} v(t_{i+1}; s_i, q_i)^\top \nabla_v L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \\ \nabla_{q_i} L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) &= \nabla_{q_i} v(t_{i+1}; s_i, q_i)^\top \nabla_v L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}), \end{aligned}$$

where the sensitivities

$$\nabla_{s_i} v(t_{i+1}; s_i, q_i) \in \mathbb{R}^{(n_x+1) \times n_x} \quad \text{and} \quad \nabla_{q_i} v(t_{i+1}; s_i, q_i) \in \mathbb{R}^{(n_x+1) \times n_u}$$

can be obtained by integrating the state sensitivity equation

$$\begin{cases} \forall t \in [t_i, t_{i+1}], \quad \frac{d \nabla_s v(t; s_i, q_i)}{dt} = \nabla_v F(v(t; s_i, q_i), q_i) \nabla_s v(t; s_i, q_i) \\ \nabla_s v(t_i; s_i, q_i) = \begin{bmatrix} I_{n_x} \\ 0 \end{bmatrix} \end{cases}$$

and the input sensitivity equation

$$\begin{cases} \forall t \in [t_i, t_{i+1}], \frac{d\nabla_q v(t; s_i, q_i)}{dt} = \nabla_v F(v(t; s_i, q_i), q_i) \nabla_q v(t; s_i, q_i) + \nabla_q F(v(t; s_i, q_i), q_i) \\ \nabla_q v(t_i; s_i, q_i) = 0 \end{cases} ,$$

which requires integrating $(n_x + 1) \times (n_x + n_u)$ ordinary differential equations. However, by doing so, we lose the advantage provided by the augmented Lagrangian, which is the ability to apply *adjoint sensitivity analysis* and thus reduce the complexity in the gradient computation. Given $i \in \{0, \dots, N - 1\}$, we define a function $\mathcal{F} : \mathbb{R}^{2n_x + n_u} \rightarrow \mathbb{R}$ by

$$\mathcal{F}(r_i) := L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \quad ,$$

where r_i has been defined in (4.17). By adjoining the augmented state dynamics (4.6), we obtain a function

$$\tilde{\mathcal{F}}(r_i) = \mathcal{F}(r_i) + \int_{t_i}^{t_{i+1}} \langle \lambda_v(t), F(v(t; s_i, q_i), q_i) - \dot{v}(t; s_i, q_i) \rangle dt \quad , \quad (4.23)$$

where an adjoint mapping $\lambda_v : \mathbb{R} \rightarrow \mathbb{R}^{n_x + 1}$ associated with the augmented state $v(t; s_i, q_i)$ is introduced. The key idea is that

$$\tilde{\mathcal{F}}(r_i) = \mathcal{F}(r_i) \quad ,$$

since for all $t \in [t_i, t_{i+1}]$,

$$\dot{v}(t; s_i, q_i) = F(v(t; s_i, q_i), q_i) \quad ,$$

so that we can compute the gradient of $\tilde{\mathcal{F}}$ in place of the gradient of \mathcal{F} . By doing so, the adjoint term in (4.23) is used to cancel the terms containing the sensitivity matrices. Given $j \in \{1, \dots, 2n_x + n_u\}$ a coordinate index of the vector r_i , an integration by parts on the integral term in (4.23) yields

$$\begin{aligned} \partial_j \mathcal{F}(r_i) = & \underbrace{\partial_j L_\rho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1})}_{\neq 0 \text{ if } j \in \{n_x + n_u + 1, \dots, 2n_x + n_u\}} \\ & + \underbrace{\int_{t_i}^{t_{i+1}} \langle \lambda_v(t), \partial_j F(v(t; s_i, q_i), q_i) \rangle dt}_{\neq 0 \text{ if } j \in \{n_x + 1, \dots, n_x + n_u\}} + \underbrace{\langle \lambda_v(t_i), \partial_j v(t_i; s_i, q_i) \rangle}_{\neq 0 \text{ if } j \in \{1, \dots, n_x\}} \quad , \end{aligned} \quad (4.24)$$

by choosing the adjoint mapping λ_v as the solution of the adjoint boundary value problem

$$\begin{cases} \lambda_v(t_{i+1}) = \nabla_v L_\varrho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) \\ \forall t \in [t_i, t_{i+1}], \dot{\lambda}_v(t) = -\nabla_v F(v(t; s_i, q_i), q_i)^\top \lambda_v(t) \end{cases} \quad (4.25)$$

Integrating (4.25) backwards in time yields $\lambda_v(t_i)$. In (4.24), each of the three terms corresponds to a different contribution:

- $\langle \lambda_v(t_i), \partial_j v(t_i; s_i, q_i) \rangle$ is the gradient of \mathcal{F} with respect to the initial condition s_i ,
- $\int_{t_i}^{t_{i+1}} \langle \lambda_v(t), \partial_j F(v(t; s_i, q_i), q_i) \rangle dt$ is the gradient of \mathcal{F} with respect to the control input q_i ,
- $\partial_j L_\varrho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1})$ is the gradient of \mathcal{F} with respect to s_{i+1} .

In order to compute the integral term in (4.24), we introduce the input adjoint mapping $\lambda_u : \mathbb{R} \rightarrow \mathbb{R}^{n_u}$ as

$$\lambda_u(t)_j := \int_t^{t_{i+1}} \langle \lambda_v(t), \partial_{q_j} F(v(t; s_i, q_i), q_i) \rangle dt \quad ,$$

where $j \in \{1, \dots, n_u\}$. Hence, the input adjoint dynamics are

$$\dot{\lambda}_u(t)_j = -\langle \partial_{q_j} F(v(t; s_i, q_i), q_i), \lambda_v(t) \rangle$$

subject to the final condition

$$\lambda_u(t_{i+1})_j = 0 \quad ,$$

for $j \in \{1, \dots, n_u\}$. Finally, we define the full adjoint mapping $\lambda : \mathbb{R} \rightarrow \mathbb{R}^{n_x+n_u+1}$ as

$$\lambda(t) := (\lambda_v(t)^\top, \lambda_u(t)^\top)^\top \quad ,$$

which is a solution of the boundary value problem

$$\begin{cases} \lambda(t_{i+1}) = (\nabla_v L_\varrho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1})^\top, 0^\top)^\top \\ \forall t \in [t_i, t_{i+1}], \dot{\lambda}(t) = -\left(\nabla_v F(v(t; s_i, q_i), q_i) \quad \nabla_q F(v(t; s_i, q_i), q_i) \right)^\top \lambda_v(t) \end{cases} \quad ,$$

with

$$\nabla_v L_\varrho(v(t_{i+1}; s_i, q_i), s_{i+1}, \mu_{i+1}) = \begin{pmatrix} -\mu_{i+1} - \varrho(s_{i+1} - x(t_{i+1}; s_i, q_i)) \\ 1 \end{pmatrix}$$

and

$$\nabla_v F(v(t; s_i, q_i), q_i) = \begin{bmatrix} \nabla_x f(x(t; s_i, q_i), q_i) & 0 \\ \nabla_x l(x(t; s_i, q_i), q_i)^\top & 0 \end{bmatrix}.$$

Subsequently, the gradient of \mathcal{F} with respect to r_i is given by

$$\begin{aligned} \nabla_{s_i} \mathcal{F}(r_i) &= \lambda_v(t_i) \\ \nabla_{q_i} \mathcal{F}(r_i) &= \lambda_u(t_i) \\ \nabla_{s_{i+1}} \mathcal{F}(r_i) &= \mu_{i+1} + \varrho(s_{i+1} - x(t_{i+1}; s_i, q_i)), \end{aligned}$$

where λ_v corresponds to the first n_x components of λ_w . Finally, the full gradient of the augmented Lagrangian (4.10) with respect to the primal variable z is

$$\nabla_z \mathcal{L}_\varrho(z, \mu, \hat{x}_0) = \begin{pmatrix} \mu_0 + \frac{\varrho}{2}(s_0 - \hat{x}_0) + \lambda_v(t_0) \\ \lambda_u(t_0) \\ \mu_1 + \varrho(s_1 - x(t_1; s_0, q_0)) + \lambda_v(t_1) \\ \vdots \\ \mu_i + \varrho(s_i - x(t_i; s_{i-1}, q_{i-1})) + \lambda_v(t_i) \\ \lambda_u(t_i) \\ \vdots \\ \mu_N + \varrho(s_N - x(t_N; s_{N-1}, q_{N-1})) \end{pmatrix}. \quad (4.26)$$

Thus, evaluating $\nabla_z \mathcal{L}_\varrho(z, \mu, \hat{x}_0)$ requires

$$N(\mathcal{I}(n_x) + \mathcal{I}(n_u)) + 4(N+1)n_x$$

floating point operations, where $\mathcal{I}(n)$ denotes the cost of integration of n ODEs, whereas computing the gradient of the full augmented Lagrangian via forward sensitivity analysis involves

$$(N+3)n_x + N(\mathcal{I}(n_x^2) + \mathcal{I}(n_x n_u))$$

floating point operations. In conclusion, the adjoint sensitivity analysis is particularly advisable when the number of states and inputs is large.

To conclude this paragraph, we summarise the complexity of each phase of the algorithm in Tab. 4.1. $N(n_x + n_u) + n_x$ is the problem dimension and n_b is the number of off-diagonal bands in the preconditioner. Most of the complexity estimates below are in worst case, which

	Phase	Complexity (flops)
Cauchy point computation	Projected search	$\Theta(3(N(n_x + n_u) + n_x))$
	Active-set extraction	$4(N(n_x + n_u) + n_x)$
Banded preconditioner	Build	$N(2n_x + n_u)n_b$ $+n_b(N(n_x + n_u) + n_x)(8 + n_b)$
	Apply (forward-backward solve)	$2(N(n_x + n_u) + n_x)(2n_b + 1)$
PCG loop	Structured Matrix-vector product	$2(2n_x + n_u)^2 N + (2n_x + n_u) N$
	Safeguarding	$8(N(n_x + n_u) + n_x)$
	Directions & residuals	$10(N(n_x + n_u) + n_x)$
Integration	States	$N\mathcal{I}(n_x)$
	Adjoint	$N(\mathcal{I}(n_x) + \mathcal{I}(n_u))$
SR1 update		$N(2n_x + n_u)(5(2n_x + n_u) + 5)$

Table 4.1: Worst-case complexity estimates of the main phases in the primal loop of the multiple-shooting augmented Lagrangian algorithm.

corresponds to all variables being free. From Tab. 4.1, one can expect the algorithm to behave well on problems with a large number of shooting nodes, as most of the phases have complexity $\mathcal{O}(N(n_x + n_u) + n_x)$. When the number of states n_x or inputs n_u becomes large, the bottlenecks are the structured matrix-vector products and the SR1 updates, which both involve a term $(2n_x + n_u)^2$.

4.1.3 Software description

Based on the multiple-shooting algorithm described in paragraphs 4.1.1 and 4.1.2, a C++ software package has been implemented. Its architecture is presented in the class diagram in Fig. 4.5. The code is designed to solve continuous-time optimal control problems of the form

$$\begin{aligned}
 & \underset{x(\cdot), u(\cdot)}{\text{minimise}} \int_{t_0}^{t_f} l(x(t), u(t), y^r(t), u^r(t)) dt + V(x(T)) & (4.27) \\
 & \text{s.t. } \forall t \in [t_0, t_f], \dot{x}(t) = f(x(t), u(t)) \\
 & \quad \forall t \in [t_0, t_f], g(x(t), u(t)) \leq 0 \\
 & \quad \forall t \in [t_0, t_f], \underline{x} \leq x(t) \leq \bar{x}, \underline{u} \leq u(t) \leq \bar{u} ,
 \end{aligned}$$

where the functions l , f and g are differentiable, y^r and u^r are output and input references and V is a Mayer term.

One of the salient ingredients of the code is C++ *template classes*, which are used in order to avoid virtual methods, which can seriously impact code performance if called at a high rate. We think this is important in our context, since several function and gradient evaluations are carried out per time instant in a real-time setting, by calling the user-defined methods:

- `MyDynamics` that implements the right hand side of the dynamics $f(x(t), u(t))$ as well as its state and input jacobians,
- `MyTrackCost` that implements the objective $l(x(t), u(t), y^r(t), u^r(t))$ as well as its state and input gradients,
- `MyPathConstraint` implementing the path-constraint $g(x(t), u(t))$ as well as its state and input gradients,
- `MyMayer` implementing the Mayer term V and its gradient,
- `MyTrajectory` that implements the references y^r and u^r .

The user-defined classes are then combined with the rest of the code via the template classes:

- `AugODEEquation` implementing the augmented dynamics from the classes `MyDynamics` and `MyTrackCost` as well as its adjoint,
- `ExplicitRKIntegrator` containing explicit Runge-Kutta integrators [44],
- `MuShoot` implementing the evaluation of the shooting constraints and objective,

- `NonlinOCpsolver` containing the trust region and augmented Lagrangian algorithms and aggregating the classes `ActivityDetector` for computing the Cauchy point and `PreconRefine` for performing the safeguarded PCG iterations.

The banded preconditioner is implemented in the class `Preconditioner`. The number of bands can be set by the user.

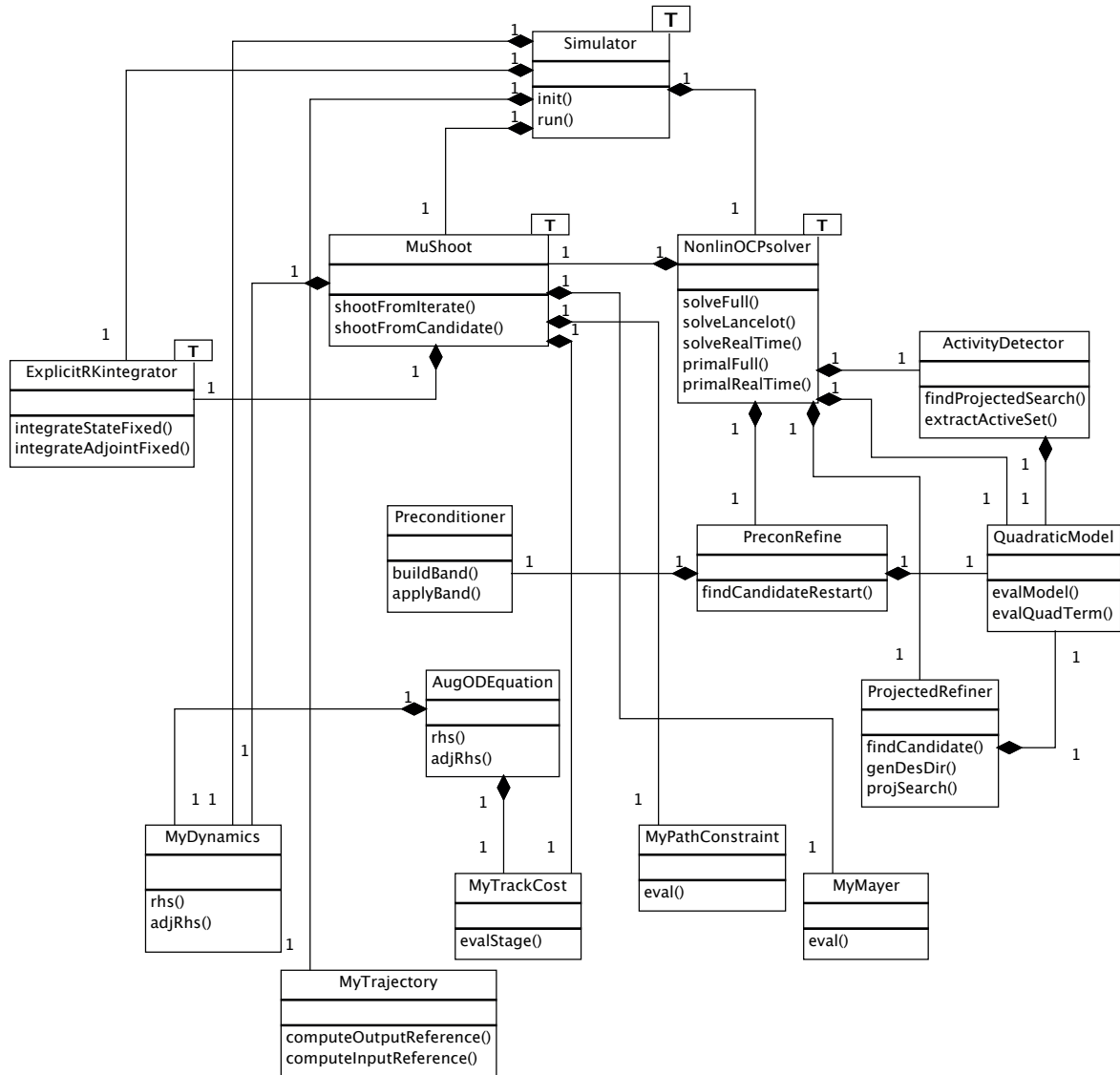


Figure 4.5: Class diagram of the C++ software for solving continuous-time OCPs via multiple-shooting and augmented Lagrangian. The box **T** stands for C++ template class.

4.1.4 Numerical experiments

The efficacy of the algorithm and software presented in paragraphs 4.1.2 and 4.1.3 is assessed by means of numerical examples. In particular, in a real-time setting, we compare the performance of our software with the existing codes for online NMPC, which are based on the so-called *real-time iteration* [46] and are available in the ACADO toolkit [91]. The basic principle of the real-time iteration is to solve a single convex QP per time instant, which is constructed from the jacobian and gradient of a multiple-shooting NLP. The real-time iteration consists of:

- a preparation phase, during which integration and sensitivity analysis are performed, along with a condensing step leading to a small-scale convex QP [91],
- a feedback phase, during which the convex QP is solved via a tailored convex QP solver such as QPOASES [58], FORCES [48] or QPDUNES [62].

The purpose of sparse convex QP solvers such as QPDUNES is to avoid condensing the QP, as it becomes a computational bottleneck when solving NMPC problems with long prediction horizons [61]. It is important to note that all the QP solvers in the ACADO toolkit use direct linear algebra operations. Therefore, one can reasonably expect their scalability to be limited, especially for QPOASES and FORCES. We actually demonstrate this last point on an NMPC problem with long horizon. Contrary to the real-time iteration strategy, in principle, our algorithm performs several integrations and sensitivity generations per time step. However, this is not necessarily a drawback. First, the adjoint sensitivity analysis is much cheaper than the forward sensitivity analysis used in the real-time iteration [153]. Secondly, one can easily reduce the number of primal iterations by enforcing a looser stopping criterion based on the satisfaction of the KKT conditions of the parametric augmented Lagrangian subproblem. Such a strategy proves effective in practice, as demonstrated in the following examples. Finally, the evaluation of the shooting constraints and gradients can be easily parallelised [99], hence reducing the computational burden when dealing with large-scale programs.

4.1.4.1 Inverted pendulum

We consider the application of our real-time NMPC algorithm to control an inverted pendulum, as shown in Fig. 4.6 below and whose dynamics is

$$\begin{cases} \ddot{x} = \frac{ml \sin(\theta) \dot{\theta}^2 + mg \cos(\theta) \sin(\theta) + u}{M + m - m(\cos \theta)^2}, \\ \ddot{\theta} = -\frac{ml \cos(\theta) \sin(\theta) \dot{\theta}^2 + u \cos \theta + (M + m)g \sin \theta}{l(M + m - m(\cos \theta)^2)}, \end{cases} \quad (4.28)$$

where x is the horizontal position of the cart, θ the angular position of the pendulum and u the force applied to the cart, which is the control input. The model parameters are

$$m = 0.1\text{kg}, M = 1\text{kg}, l = 0.5\text{m} ,$$

from [130], where m is the mass of the inverted pendulum, M is the mass of the chart and l the length of the pendulum.

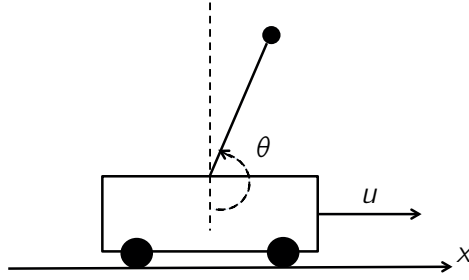


Figure 4.6: Schematic illustrating the inverted pendulum.

The control objective is to move the inverted pendulum from the stable equilibrium $\theta = 0$ to the unstable one $\theta = \pi$ and stabilise it while satisfying the following state and input constraints

$$-2 \leq x \leq 2, \quad -20 \leq u \leq 20 .$$

After transforming Eq. (4.28) into

$$\dot{z}(t) = f(z(t), u(t))$$

$$:= \begin{pmatrix} z_2(t) \\ \frac{ml \sin(z_3(t)) z_4(t)^2 + mg \cos(z_3(t)) \sin(z_3(t)) + u(t)}{M + m - m(\cos z_3(t))^2} \\ z_4(t) \\ -\frac{ml \cos(z_3(t)) \sin(z_3(t)) z_4(t)^2 + u(t) \cos z_3(t) + (M + m) g \sin z_3(t)}{l(M + m - m(\cos z_3(t))^2)} \end{pmatrix} ,$$

with

$$z := \begin{pmatrix} x & \dot{x} & \theta & \dot{\theta} \end{pmatrix}^\top .$$

The control problem can be easily formalised as an NMPC program for tracking the input and state

references

$$u^r = 0 \quad \text{and} \quad z^r = \begin{pmatrix} 0 & 0 & \pi & 0 \end{pmatrix}^\top ,$$

which leads to

$$\begin{aligned} & \underset{z,u}{\text{minimise}} \int_{t_0}^{t_0+T} (z(t) - z^r)^\top Q (z(t) - z^r) + (u(t) - u^r)^\top R (u(t) - u^r) dt & (4.29) \\ & + z(T)^\top P z(T) \\ \text{s.t. } & z(t_0) = \hat{z}_0 , \\ & \dot{z}(t) = f(z(t), u(t)) , \\ & -2 \leq z_1(t) \leq 2 , \\ & -20 \leq u(t) \leq 20 . \end{aligned}$$

where Q , R and P are positive definite matrices defined as

$$Q = \text{diag}(10, 10, 0.1, 0.1), \quad R = (0.01), \quad P = Q .$$

For the NMPC stability guarantees to hold [77], the OCP (4.29) is to be solved fully at every time instant as the parameter \hat{z}_0 varies. Instead, we apply the parametric tracking Algorithm 7 coupled with the trust region loop 9 to find an approximate critical point of the parametric augmented Lagrangian subproblem. Stability of the optimality-tracking error follows from the analysis in Chapter 3.

As a first scenario, the prediction horizon is set to $T = 1$ sec along with $N = 20$ shooting intervals. The multiple-shooting discretisation yields a small-scale NLP with 104 variables and 84 nonlinear equality constraints. The shooting constraints and adjoints are evaluated via 1 step of an explicit Runge-Kutta integrator of order 4. The inverted pendulum dynamics is simulated using 10 steps of a 4th order explicit Runge-Kutta integrator. The maximum number of trust region iterations is set to 5 and the penalty to 60. We use the banded preconditioner with 6 bands. It is worth noting that with 8 bands, the entire nonzero part of the SR1 approximation is taken into account. Results are shown in Fig. 4.7, Fig. 4.8, Fig. 4.9, Fig. 4.10 and Fig. 4.11.

The trajectories plotted in Fig 4.7 show that although a very limited number of iterations are carried out, the suboptimal NMPC controller is able to stabilise the inverted pendulum around the desired unstable set-point. The suboptimal trajectory is also close to the one obtained by running the LANCELOT outer loop until a tight feasibility is obtained. The euclidean norm of the multiple-shooting constraints is plotted in Fig. 4.8, along with the KKT satisfaction on the augmented

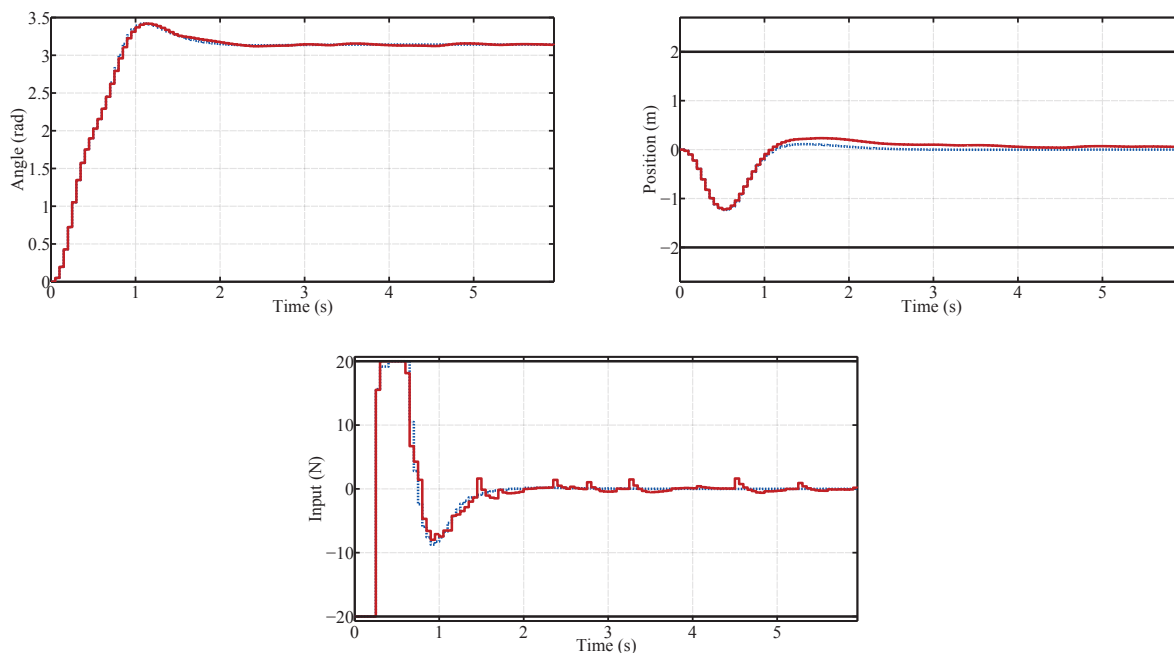


Figure 4.7: Angular, horizontal position and control input of inverted pendulum using MUTRAL with $T = 1$ sec and $N = 20$. The sub-optimal trajectories obtained with Algorithm 9 are plotted in dashed red, while the full NMPC trajectories obtained using a complete augmented Lagrangian dual loop are plotted in blue.

Lagrangian subproblem in Fig. 4.9. One can observe that a relatively low satisfaction of the KKT conditions is sufficient to obtain a suitable control law. Figures 4.10 and 4.11 show that the computation time of our algorithm is directly correlated to the cumulative number of PCG iterations. For some problem instances, this number can be very low (around 1), while for some others it is close to 50. This is mainly due to the loose tolerance on the KKT conditions and the warm-starting effect.

On a 2.5 GHz processor with an 8GB memory, over a horizon of 4 sec, we compare average computation times of our algorithm, called MUTRAL for **m**ultiple-shooting via **t**rust-region and **a**ugmented **L**agrangian, ACADO-FORCES, ACADO-QPDUNES and ACADO-QPOASES. With MUTRAL, we obtained an average computation time of $230\mu\text{s}$ and a worst-case computation time of $552\mu\text{s}$.

From Tab. 4.2, it appears that for a short horizon and small NMPC problem, MUTRAL is not the most suitable code. As a second scenario, we set the prediction horizon to 3 sec with $N = 60$ shooting intervals, which results in a larger NLP with 304 variables and 244 equality constraints. We keep the KKT tolerance at 0.1, the penalty at 60 and the number of bands at 6. However, we truncate the maximum number of trust region iterations and set it to 3. The resulting closed-loop

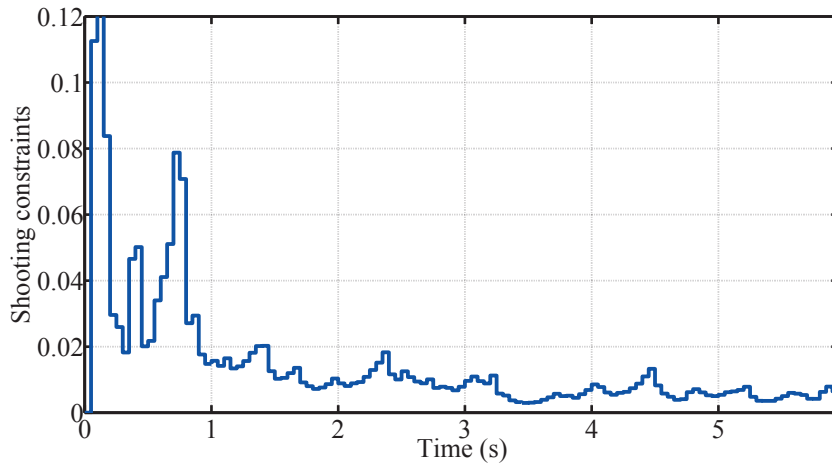


Figure 4.8: Euclidean norm of shooting constraints using MUTRAL with $T = 1$ sec and $N = 20$.

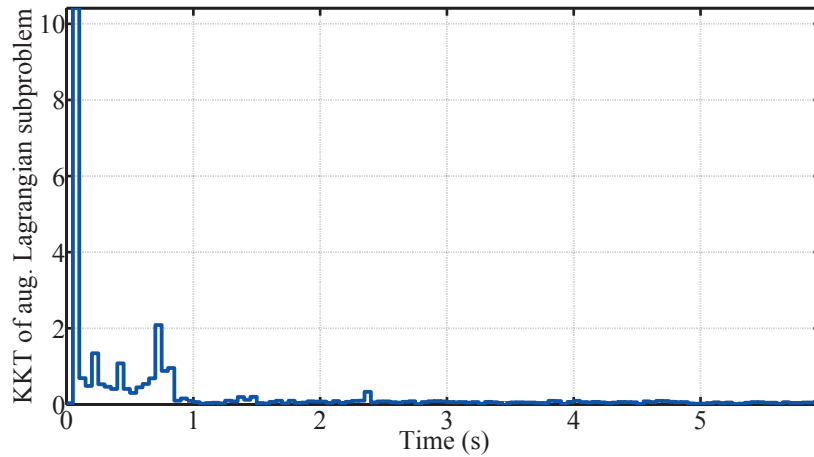
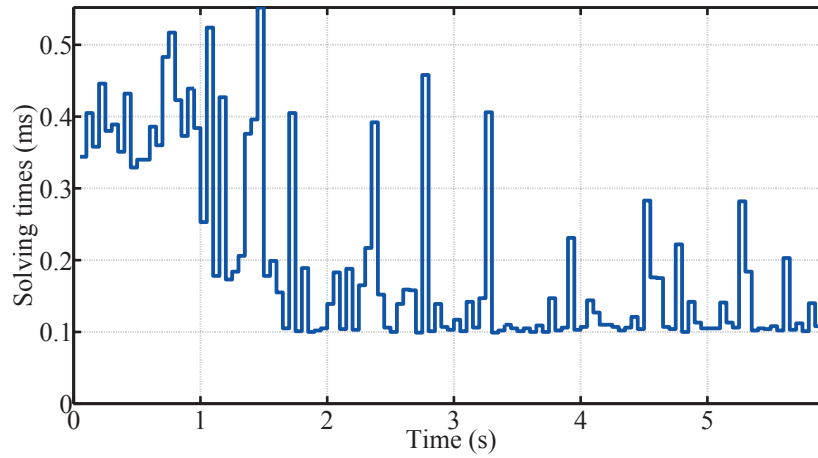
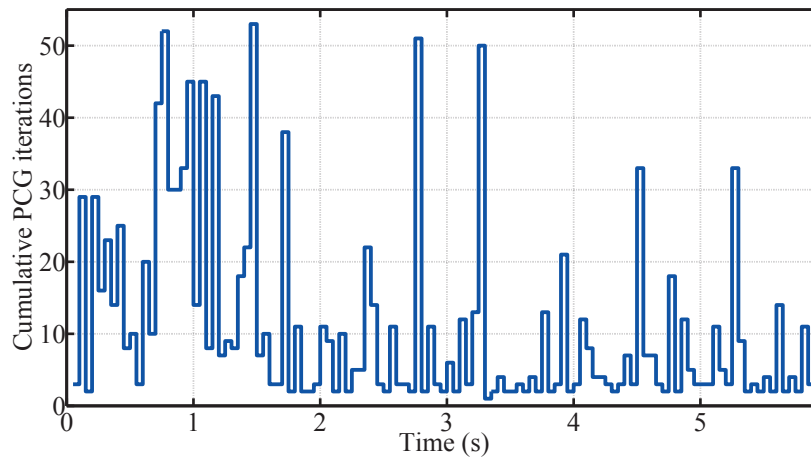


Figure 4.9: KKT satisfaction on augmented Lagrangian subproblem using MUTRAL with $T = 1$ sec and $N = 20$.

	MUTRAL	ACADO-FORCES	ACADO-QPDUNES	ACADO-QPOASES
Average over first 4 sec	$230\mu s$	$199\mu s$	$97\mu s$	$80\mu s$

Table 4.2: Computation times of different online NMPC software on the inverted pendulum NMPC problem with horizon $T = 1$ sec and $N = 20$ shooting intervals.

trajectory and input are shown in Fig. 4.12. It is worth noting that the error with respect to the full NMPC trajectory with the same horizon is smaller than with horizon $T = 1$ sec and $N = 20$

Figure 4.10: Solving times using MUTRAL with $T = 1$ sec and $N = 20$.Figure 4.11: Cumulative PCG iterations using MUTRAL with $T = 1$ sec and $N = 20$.

shooting intervals.

MUTRAL yields an average computation time of $352\mu\text{s}$ and a worst case time of $771\mu\text{s}$. A comparison with the other NMPC codes is given in Tab. 4.3 below. It appears that MUTRAL performs better on average than ACADO-FORCES and ACADO-QPOASES, and is very close to ACADO-QPDUNES.

At this point, a natural question should be raised: does increasing the number of trust region iterations help reducing the error with respect to the full NMPC trajectory? Given the same penalty ρ and number of bands in the preconditioner, we vary the maximum number of trust region it-

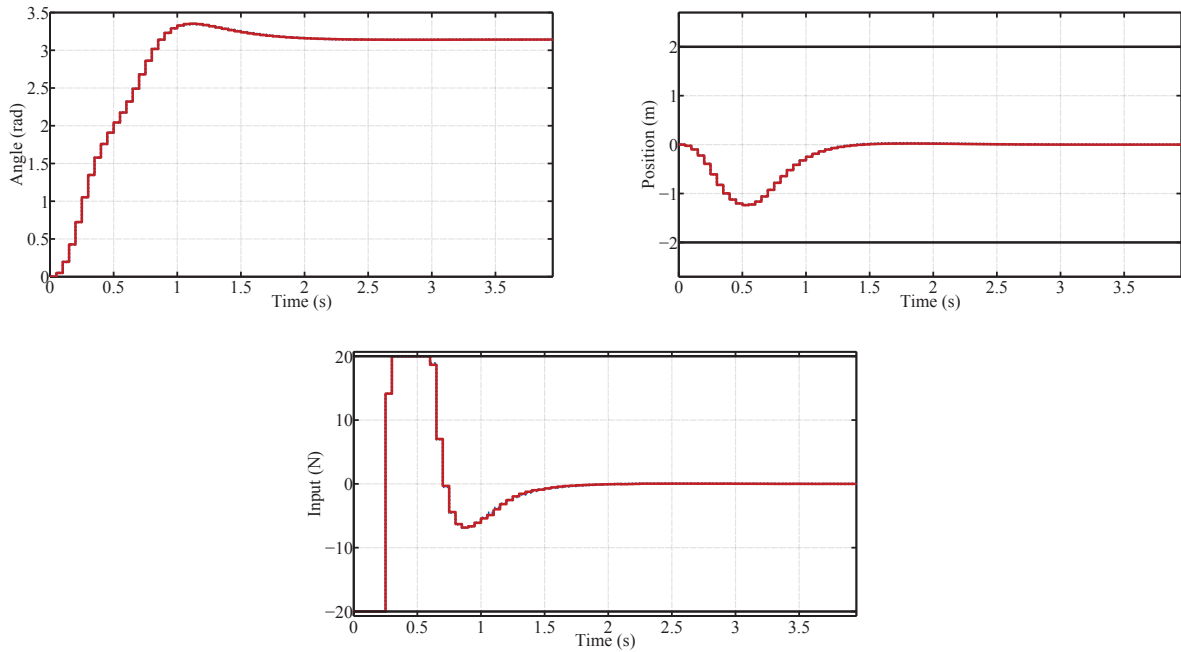


Figure 4.12: Angular, horizontal position and control input of inverted pendulum using MUTRAL with $T = 3$ sec and $N = 60$. The sub-optimal trajectories obtained with Algorithm 9 are plotted in dashed red, while the full NMPC trajectories obtained using a complete augmented Lagrangian dual loop are plotted in blue.

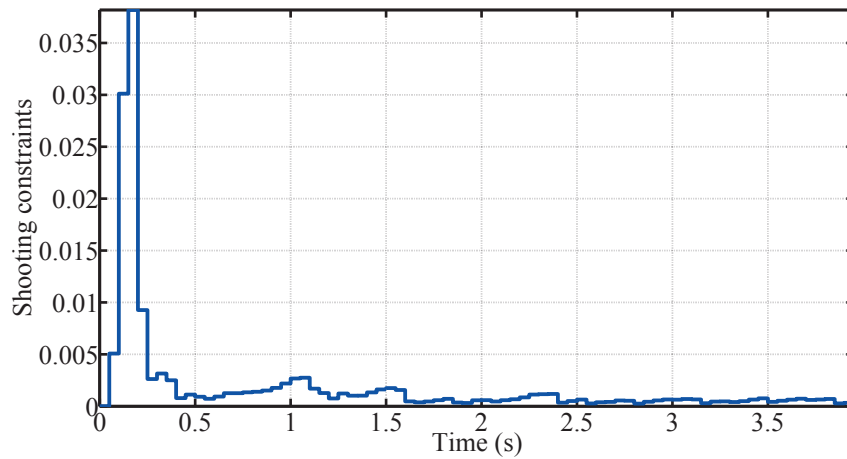


Figure 4.13: Euclidean norm of shooting constraints using MUTRAL with horizon $T = 3$ sec and $N = 60$ shooting intervals.

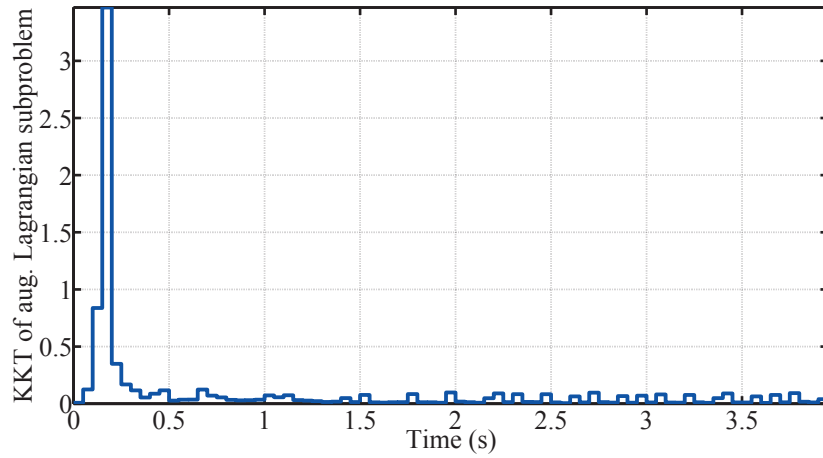


Figure 4.14: KKT satisfaction on augmented Lagrangian subproblem using MUTRAL with horizon $T = 3$ sec and $N = 60$ shooting intervals.

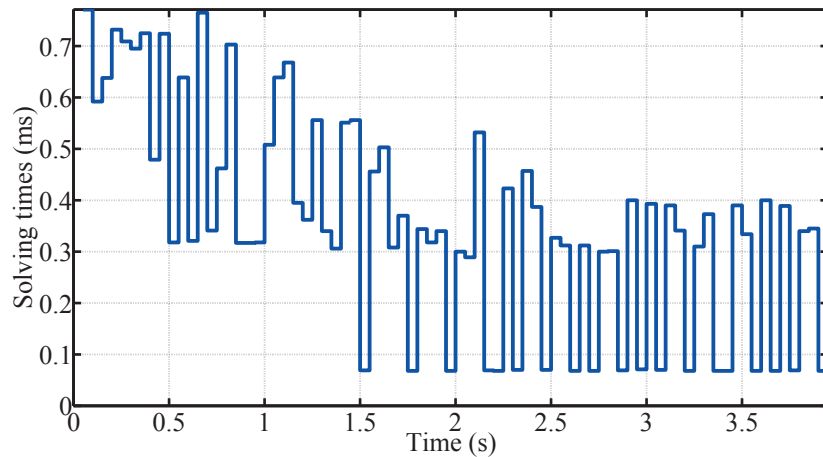


Figure 4.15: Solving times using MUTRAL with horizon $T = 3$ sec and $N = 60$ shooting intervals.

	MUTRAL	ACADO-FORCES	ACADO-QPDUNES	ACADO-QPOASES
Average over 4 sec	$352\mu\text{s}$	$745\mu\text{s}$	$312\mu\text{s}$	$407\mu\text{s}$

Table 4.3: Computation times of different online NMPC softwares on the inverted pendulum NMPC problem with horizon 3 sec and 60 shooting intervals.

erations and record the tracking error with respect to the optimal NMPC input and closed-loop NMPC trajectory. The tolerance on the KKT conditions of the parametric augmented Lagrangian

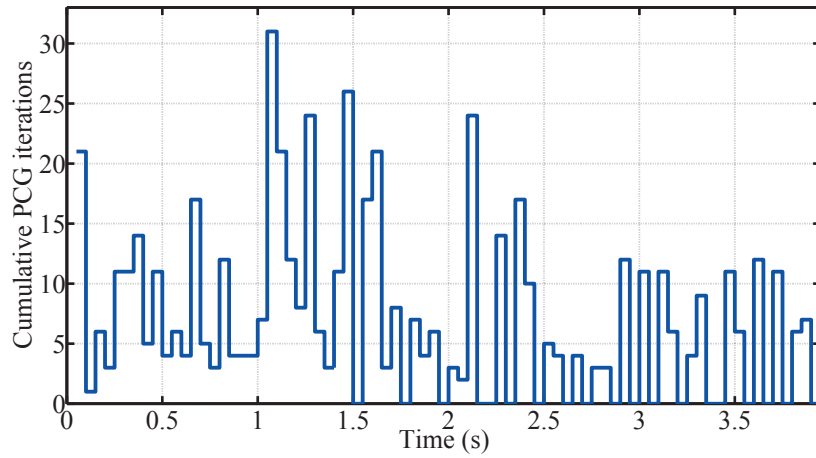


Figure 4.16: Cumulative PCG iterations using MUTRAL with horizon $T = 3$ sec and $N = 60$ shooting intervals.

subproblem is deliberately set to a low level (10^{-4}). Results are plotted in Fig. 4.17.

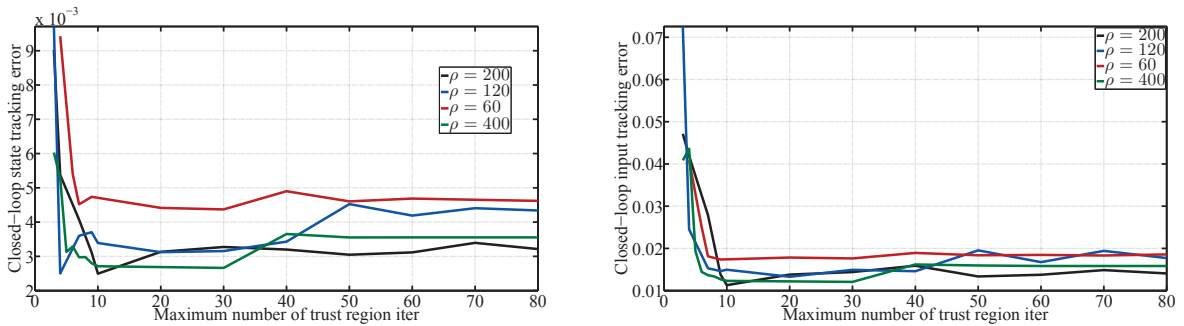


Figure 4.17: Closed-loop state and input tracking errors versus number of trust region iterations.

After a few trust region iterations, the closed-loop state and input error drop quickly. Further increasing the number of trust region iterations does not help in improving the performance of the tracking scheme. Such a behaviour, which is very different from the results obtained in Chapter 3, is a consequence of the fast local convergence rate of the quasi-Newton method and the fast activity detection properties of the gradient projection.

4.1.4.2 Real-time economic NMPC on a bioreactor

In this paragraph, we consider a nonlinear continuous-time model of bioreactor for culture fermentation. The system has five states and one input. The system dynamics is given in Eq. (4.30).

$$\dot{x}_1 = -Dx_1 + \mu(x) x_1 \quad (4.30a)$$

$$\dot{x}_2 = D(u - x_2) - \frac{\mu(x) x_1}{Y_{xs}} \quad (4.30b)$$

$$\dot{x}_3 = -Dx_3 + (\alpha\mu(x) + \beta) x_1 \quad (4.30c)$$

$$\dot{x}_4 = \frac{u}{T} \quad (4.30d)$$

$$\dot{x}_5 = \frac{x_1}{T} \quad (4.30e)$$

where

$$\mu(x) = \mu_m \frac{\left(1 - \frac{x_3}{P_m}\right) x_2}{K_m + x_2 + \frac{x_2^2}{K_i}} .$$

For the sake of brevity, we do not discuss details about the model and numerical values of parameters and refer to [129]. We consider the following economic stage-cost

$$l(x, u) = \frac{-Dx_3}{T} .$$

The input is subject to lower and upper bound,

$$\underline{u} = 28.7 \text{ g/L} \leq u \leq \bar{u} = 40.0 \text{ g/L} .$$

The control objective is to maximise the average productivity. It is known that for system (4.30), the maximum productivity is obtained when operating in periodic mode [121]. Therefore, we enforce periodicity constraints in the ENMPC problem, as follows

$$x(0) = x(T) .$$

Our software MUTRAL has been tested on a processor with 2.5 GHz and 8 GB of RAM. For this simulation, the maximum number of trust region M was set to 3. The penalty ϱ was set to 100 and a banded preconditioner with 10 bands was used. In the ENMPC problem, a prediction horizon of $T = 48$ hours with $N = 20$ shooting intervals was set. In order to simulate the system, we applied a fourth-order explicit Runge-Kutta scheme (RK4) with 20 steps. The evaluation of the shooting

constraints was performed with 1 step of RK4.

The closed-loop trajectory and the suboptimal ENMPC input are shown in Fig 4.18. An average productivity of $3.08 \text{ g/L}\cdot\text{h}$ is obtained, which is a bit lower than the productivity given by the periodic trajectory proposed in [92] ($3.11 \text{ g/L}\cdot\text{h}$), but larger than the steady-state productivity ($3.0 \text{ g/L}\cdot\text{h}$). It is worth noting that the system operates in almost periodic mode under our suboptimal ENMPC control law.

Computational results are shown in Fig. 4.19. An average solving time of $213\mu\text{s}$ was obtained. It appears that the cumulative number of sCG iterations per time step is quite low, which is a result of our preconditioning. This is interesting, as sCG iterations are one of the main computational burdens in MUTRAL.

In conclusion, on small scale tracking NMPC problems, MUTRAL does not appear to be as efficient as the ACADO toolkit [91]. However, when the problem dimension increases, MUTRAL shows superior performance to ACADO coupled with the interior-point convex QP solver FORCES or the parametric active-set convex QP solver QPOASES. Nevertheless, the combination between ACADO and the dual Newton strategy QPDUNES leads to faster computation times than MUTRAL on NMPC problems with long horizons. These results should be tempered in two ways. First, concerning the real-time algorithm, MUTRAL can be applied to a broader range of NMPC problems than ACADO, which is still limited to NMPC problems with least-squares objectives due to the Gauss-Newton approximation, as shown in paragraph 4.1.4.2. Secondly, the ACADO code has reached a more advanced stage of development than our software package MUTRAL, which is still prototypical.

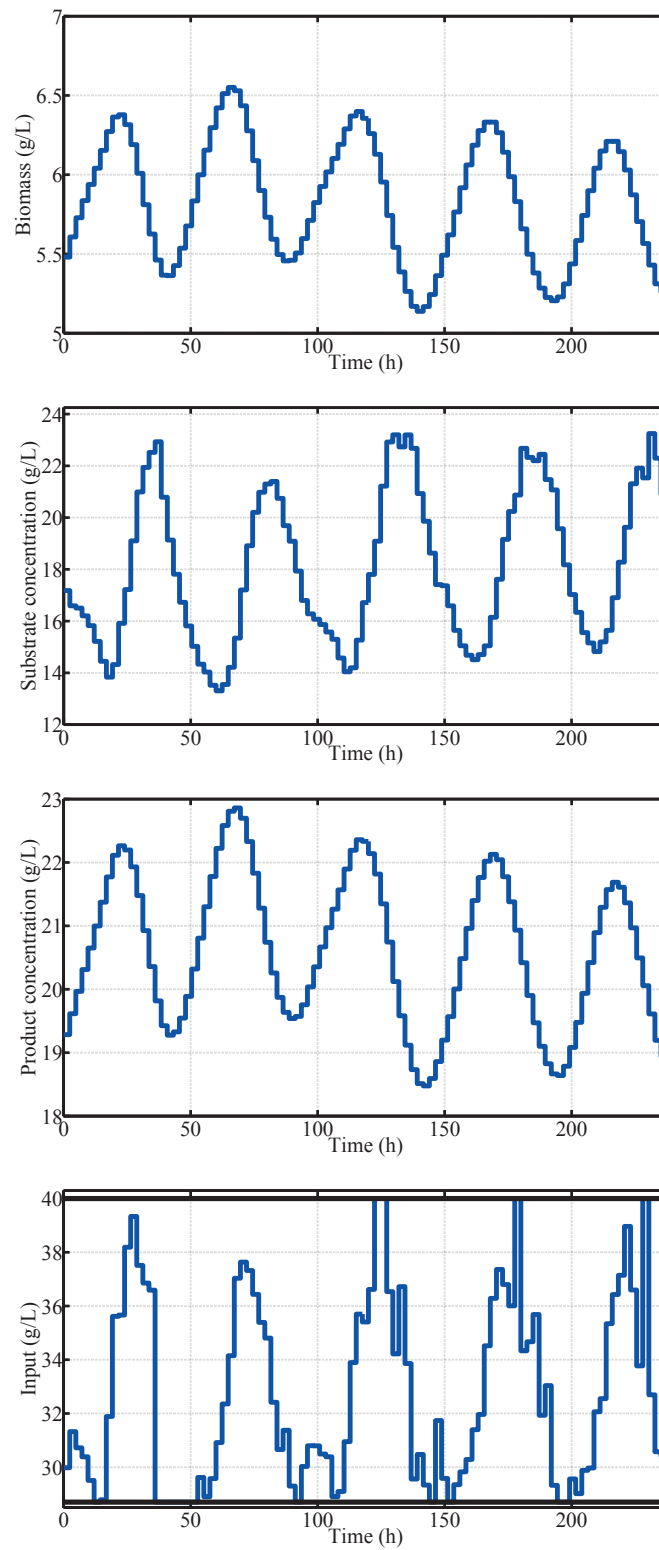


Figure 4.18: Closed-loop trajectories for the bioreactor example.

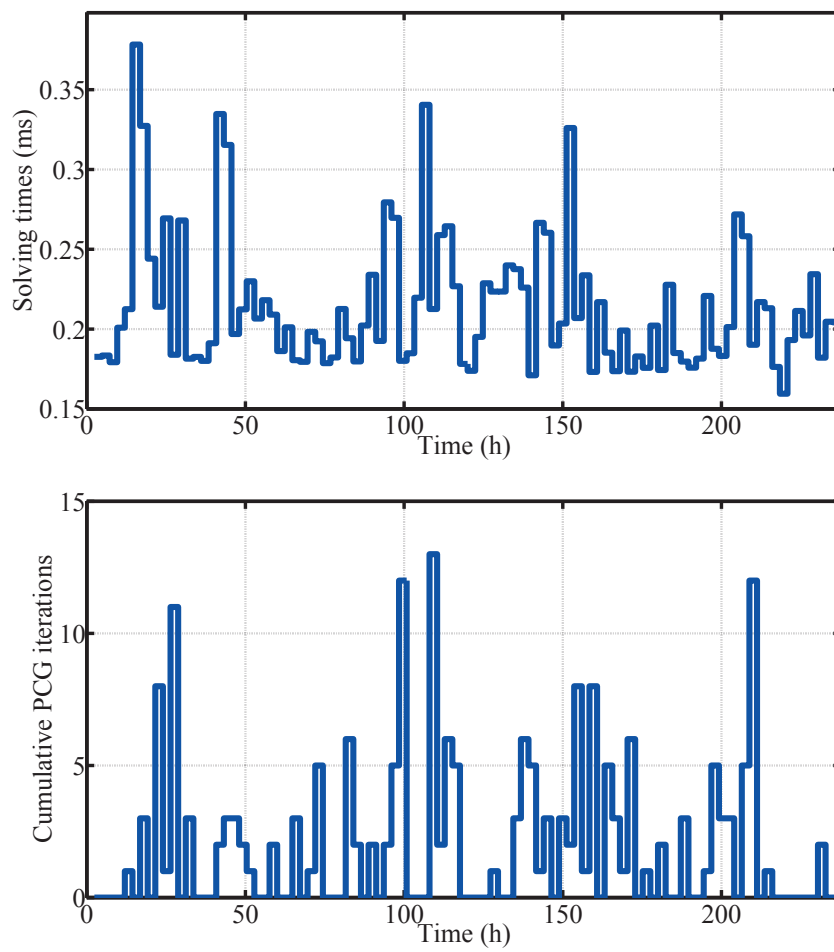


Figure 4.19: Solving times and cumulative PCG iterations for the bioreactor example.

4.2 Multi-stage Optimal AC Power Flow Problems

The AC-OPF problem has been considered in Chapter 2 in order to assess the performance of the distributed algorithm TRAP. We now intend to analyse the practical performance of the distributed optimality tracking algorithm pTRAP (Algorithm 11). For this, we recast a single instance of the AC-OPF problem into a multi-stage AC-OPF problem over a prediction horizon by incorporating dynamical entities in the network, namely batteries. The dynamics of the storage elements introduces coupling between time instants over the prediction horizon. Subsequently, we obtain a finite-horizon discrete-time optimal control problem with nonlinear coupling constraints. The varying parameters are the demand at every node of the network and the battery states. The solution of the multi-stage AC-OPF problem provides power set-points for the generating units in the network as well as injections at the storage elements, which minimise the overall generation cost.

This study focuses on real-world distribution networks [55]. In particular, we carry out numerical experiments on a 7-bus network and a 47-bus network, from which the 7-bus network is extracted. The network topologies are shown in Fig. 4.20 and 4.21 below. The network data is given in Tab. A.1 in Appendix A.

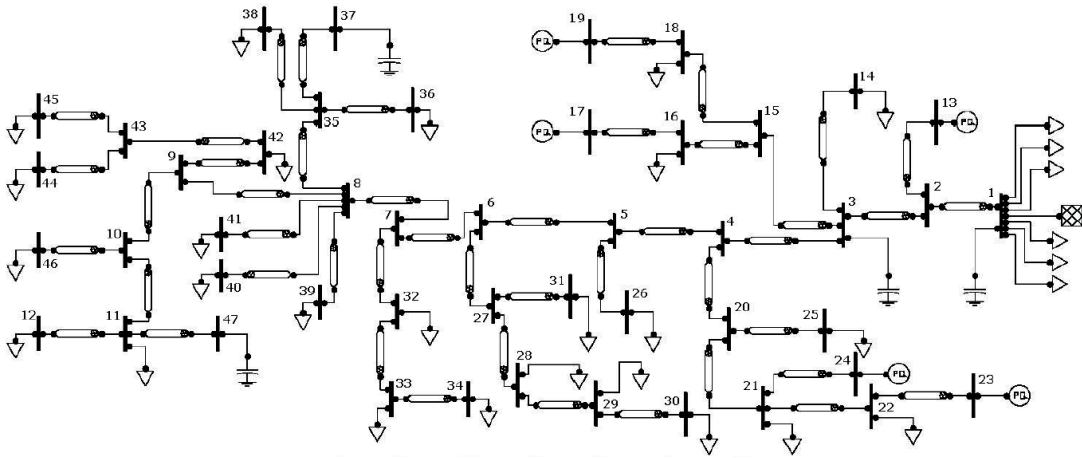


Figure 4.20: Topology of the 47-bus network [55].

Distribution networks are radial networks that are composed of buses and lines. They have a tree topology. The root is a substation, which is connected to a transmission network and has a fixed voltage. A distribution network is represented by graph $(\mathcal{N}, \mathcal{E})$, where \mathcal{N} denotes the set of vertices or nodes, and \mathcal{E} represents the set of edges or lines. A bus is attached to every node in \mathcal{N} . Each node is indexed by an integer $i \in \{0, \dots, n\}$, where 0 is the substation index and $1, \dots, n$ are the indices of the other nodes. A line is represented by a pair of indices $(i, j) \in \mathcal{E}$, where j lies

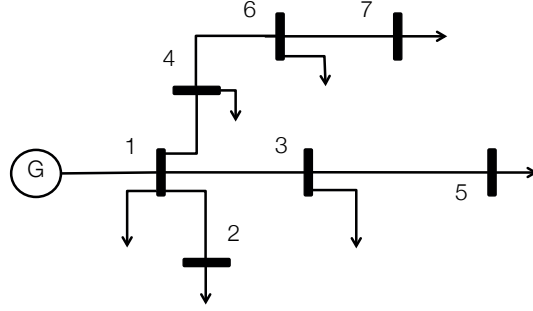


Figure 4.21: Topology of the 7-bus network.

on the unique path from bus i to the substation vertex 0. The impedance of line (i, j) is denoted by

$$z_{ij} = r_{ij} + \mathbf{i}x_{ij} \ ,$$

where the real part r_{ij} and the imaginary part x_{ij} are expressed in Ohms (Ω). The physics of radial networks is described by the *branch-and-flow* model, which consists of the following set of equations:

$$\left\{ \begin{array}{l} \forall (i, j) \in \mathcal{E} : S_{ij} = s_i + \sum_{h: h \rightarrow i} (S_{hi} - z_{hi}\mathcal{I}_{ij}) \ , \\ 0 = s_0 + \sum_{h: h \rightarrow 0} (S_{h0} - z_{h0}\mathcal{I}_{h0}) \ , \\ \forall (i, j) \in \mathcal{E} : v_i - v_j = 2\Re\{\bar{z}_{ij}S_{ij}\} - (r_{ij}^2 + x_{ij}^2)\mathcal{I}_{ij} \ , \\ \forall (i, j) \in \mathcal{E} : \mathcal{I}_{ij} = \frac{|S_{ij}|^2}{v_i} \ , \end{array} \right. \quad (4.31)$$

where S_{ij} is the complex power flowing through line (i, j) , s_i is the complex power injected at bus i , v_i denotes the squared magnitude of its complex voltage and \mathcal{I}_{ij} represents the squared magnitude of the complex current through line (i, j) . The nonlinearity in the branch-and-flow model comes from the last equality, which relates the complex current \mathcal{I}_{ij} and the complex power through line ij with the squared voltage magnitude v_i at the origin of the line. We define the global variables

$$\mathbf{S} := (S_{ij})_{(i,j) \in \mathcal{E}} \ , \ \mathcal{I} := (\mathcal{I}_{ij})_{(i,j) \in \mathcal{E}} \ , \ \mathbf{s} := (s_i)_{i \in \mathcal{N}} \ \text{and} \ \mathbf{v} := (v_i)_{i \in \mathcal{N}} \ . \quad (4.32)$$

Next, the branch-and-flow model is represented by the nonlinear equality constraint

$$\mathcal{B}(\mathbf{S}, \mathcal{I}, \mathbf{s}, \mathbf{v}) = 0 \ , \quad (4.33)$$

where \mathbf{S} , \mathcal{I} , \mathbf{s} and \mathbf{v} have been defined in Eq. (4.32). The voltages v_i and complex power injections s_i are subject to bound constraints

$$\underline{v}_i \leq v_i \leq \bar{v}_i, \Re\{\underline{s}_i\} \leq \Re\{s_i\} \leq \Re\{\bar{s}_i\}, \Im\{\underline{s}_i\} \leq \Im\{s_i\} \leq \Im\{\bar{s}_i\} .$$

An important point is that there are a few generating units in the distribution network. The subset of generators is denoted by \mathcal{G} . As claimed earlier, the storage elements are batteries with linear dynamics [69]

$$b(t + \Delta t) = b(t) + r(t) \cdot \Delta t , \quad (4.34)$$

where $b(t)$ is the state of charge at time t , $r(t)$ is the rate of charge at time t and Δt is the sampling period. The state and rate of charge are constrained as follows

$$\underline{b} \leq b(t) \leq \bar{b}, \underline{r} \leq r(t) \leq \bar{r} .$$

It is with noting that $-r(t)$ corresponds to the complex power injected into the network by the storage element. The subset of nodes equipped with batteries is denoted by \mathbb{B} . The vector of injections at time t is denoted by $\mathbf{r}(t)$. Some nodes in the network consume active and reactive power. The vector of predicted active and reactive power demands at all nodes at time t is denoted by $(\mathbf{P}_d(t)^\top, \mathbf{Q}_d(t)^\top)^\top$. In order to control the batteries so as to minimise the overall generation cost and meet the power demand, we create the multi-stage AC-OPF problem

$$\underset{s, v, \mathcal{I}, \mathbf{S}, r, b}{\text{minimise}} \sum_{t=0}^{T \cdot \Delta t} \sum_{i \in \mathcal{G}} c_i(s_i(t)) \quad (4.35)$$

s.t.

$$\forall t \in \{1, \dots, T\}, \mathcal{B}(S(t), \mathcal{I}(t), s(t), v(t)) + \mathcal{T}_r \mathbf{r}(t) + \mathcal{T}_d \begin{pmatrix} \mathbf{P}_d(t) \\ \mathbf{Q}_d(t) \end{pmatrix} = 0 ,$$

$$\mathcal{B}(S(0), \mathcal{I}(0), s(0), v(0)) + \mathcal{T}_r \mathbf{r}(0) + \mathcal{T}_d \begin{pmatrix} \widehat{\mathbf{P}_d(0)} \\ \widehat{\mathbf{Q}_d(0)} \end{pmatrix} = 0 ,$$

$$\forall t \in \{0, \dots, T-1\}, \forall i \in \mathbb{B}, b_i(t + \Delta t) = b_i(t) + r_i(t) \cdot \Delta t ,$$

$$\forall i \in \mathbb{B}, b_i(0) = \hat{b}_i ,$$

$$\forall t \in \{0, \dots, T\}, \forall i \in \mathcal{N}, \Re\{\underline{s}_i\} \leq \Re\{s_i(t)\} \leq \Re\{\bar{s}_i\}, \Im\{\underline{s}_i\} \leq \Im\{s_i(t)\} \leq \Im\{\bar{s}_i\} ,$$

$$\forall t \in \{0, \dots, T\}, \forall i \in \mathcal{N}, \underline{v}_i \leq v_i(t) \leq \bar{v}_i ,$$

$$\forall t \in \{0, \dots, T\}, \forall i \in \mathbb{B}, \underline{b}_i \leq b_i(t) \leq \bar{b}_i, \underline{r}_i \leq r_i(t) \leq \bar{r}_i ,$$

where T is a prediction horizon, the functions $c_i : \mathbb{R}^2 \rightarrow \mathbb{R}$ correspond to generation costs at specific buses, \mathcal{T}_r and \mathcal{T}_d are matrices of appropriate dimensions, $\{\hat{b}_i\}_{i \in \mathbb{B}}$ are the initial battery states and

$$\begin{pmatrix} \widehat{\mathbf{P}_d(0)} \\ \widehat{\mathbf{Q}_d(0)} \end{pmatrix}$$

corresponds to the real demand varying over time.

In practice, the multi-stage AC-OPF program is a large-scale NLP due to the size of the network and the prediction horizon T . Moreover, a limited amount of time is generally allocated to the computation of the OPF solutions. Therefore, parametric distributed optimisation algorithms such as pTRAP are relevant. We consider two multi-stage AC-OPF problems: the 7-bus network with batteries at all nodes over a prediction horizon of 12 hours and the 47-bus network with batteries at all nodes over a prediction horizon of 6 hours. The real-world demand data is taken from the European Network of Transmission System Operators for Electricity (www.entsoe.eu). We introduce a mismatch of 2% between the predicted demand that appears in the multi-stage AC-OPF and the actual demand. The demand curves are plotted in Fig. 4.22 in the case of the 7-bus network and Fig.4.23 regarding the 47-bus network.

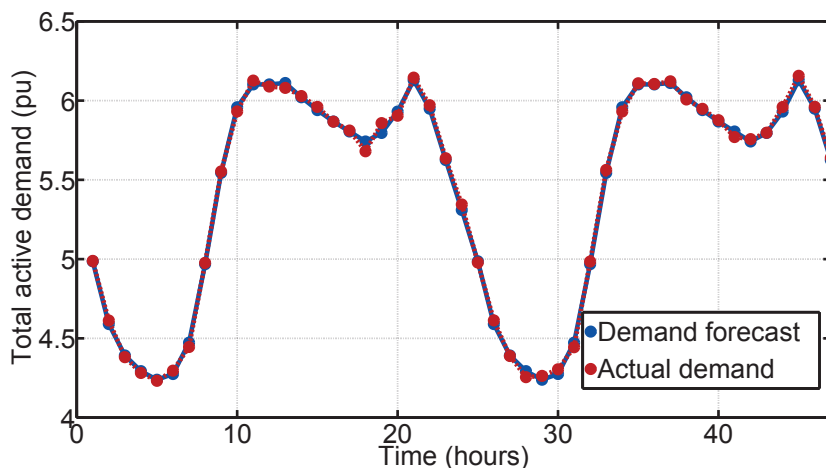


Figure 4.22: Total predicted and actual demand curves over the 7-bus distribution network.

For the 7-bus network with $T = 12$ hours, which results in an NLP with 643 variables and 415 equality constraints, we compare the generation obtained by solving the AC-OPF with IPOPT without batteries to the generation yielded by IPOPT applied to NLP (4.35) and the generation provided by Algorithm 7 coupled with 150 and 300 trust region iterations in pTRAP. The generation

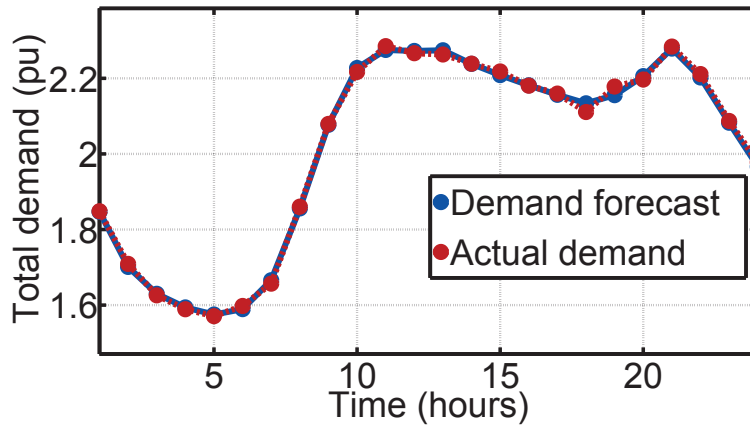


Figure 4.23: Total predicted and actual demand curves over the 47-bus distribution network.

curves are plotted in Fig. 4.24. As expected, the storage integration has a peak-shaving effect. The solutions obtained with our suboptimal algorithm have the same shape as the one yielded by IPOPT and move closer to the optimal solution as the number of trust region iterations increases. This conclusion is corroborated by the storage profiles shown in Fig. 4.25.

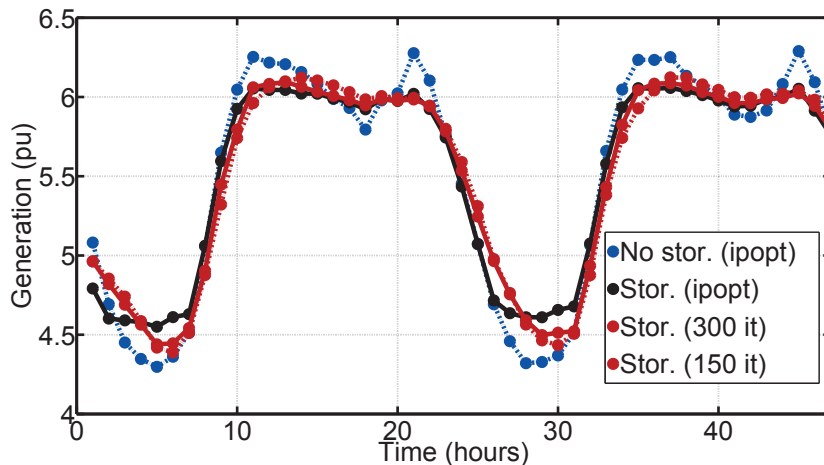


Figure 4.24: Total generation curves for the 7-bus distribution network. Comparison between the generation obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP with batteries.

As our algorithm outputs a suboptimal solution, the nonlinear power flow constraints are not satisfied. However, in order to make the suboptimal power set-points and injections applicable in practice, a sufficient level of feasibility is required, otherwise voltage collapse can occur. From

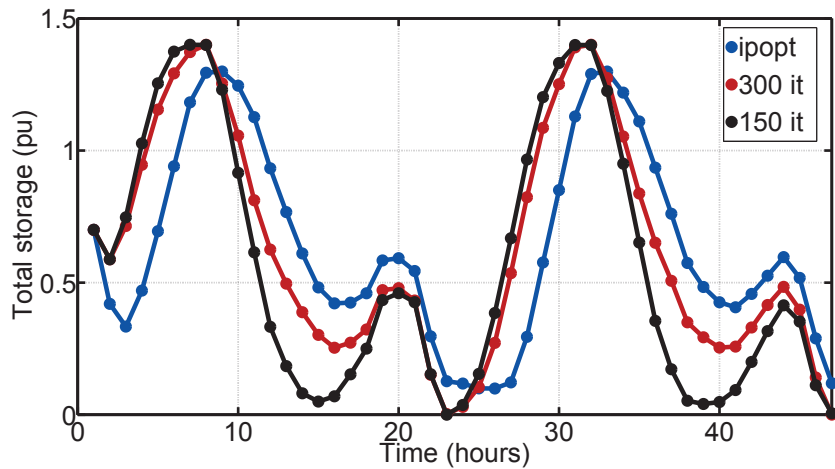


Figure 4.25: Total state of charge over the 7-bus network. Comparison between the storage obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP.

Fig. 4.26, we can conclude that this requirement is almost met by increasing the number of trust region iterations. In theory, one can further improve the feasibility level by adjusting the penalty coefficient, as shown in Chapter 3

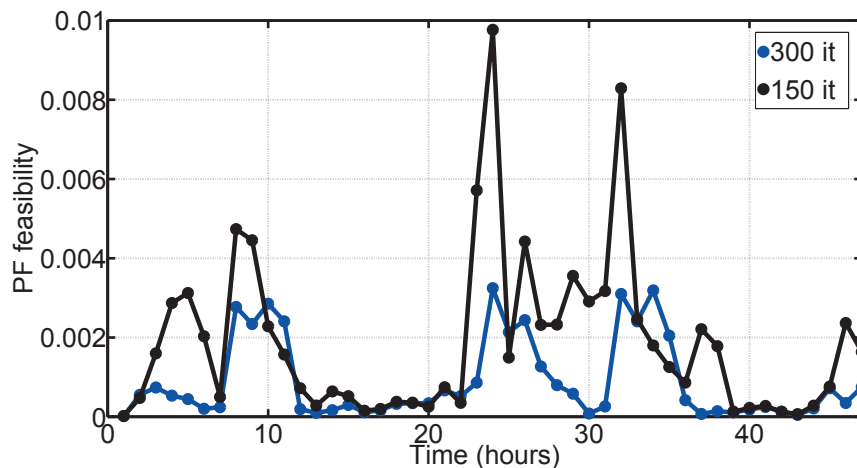


Figure 4.26: Euclidean norm of AC power flow constraints of the 7-bus network obtained by means of Algorithm 7 coupled with 150 and 300 iterations of pTRAP.

The 47-bus distribution network encompasses 9 generating units. Over a horizon of 6 hours, it results in an NLP with 2285 variables and 1451 constraints. Per time step, 100 iterations of

pTRAP are carried out along with on dual update. The penalty is set to 180. Results are shown in Fig. 4.27, 4.28 and 4.29. They are in line with the results obtained on the 7-bus network, although the storage dynamics are different, as the storage elements have a tendency to discharge even when demand is low.

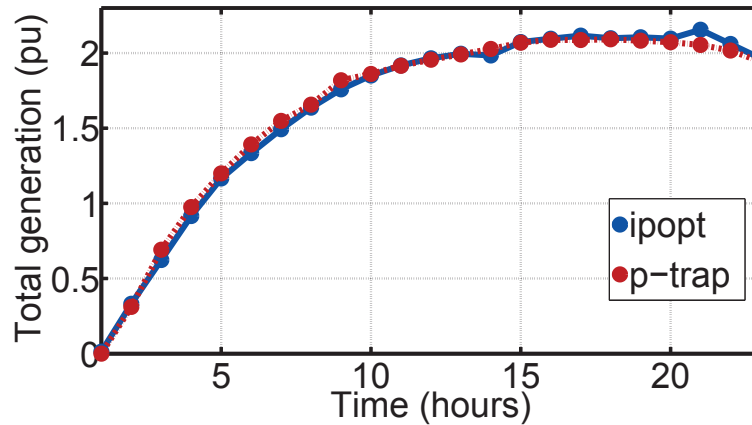


Figure 4.27: Total generation curves for the 47-bus distribution network. Comparison between the generation obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP with batteries.

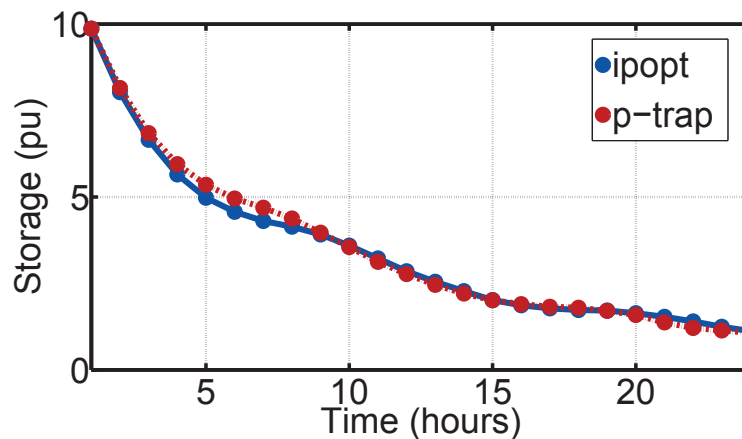


Figure 4.28: Total state of charge over the 47-bus network. Comparison between the storage obtained via IPOPT without batteries, IPOPT with batteries and via Algorithm 7 coupled with 150 and 300 iterations of pTRAP.

In conclusion, when tuned appropriately, pTRAP yields sub-optimal solutions to the AC-OPF problem, which seem to be suitable in terms of satisfaction of the nonlinear power flow con-

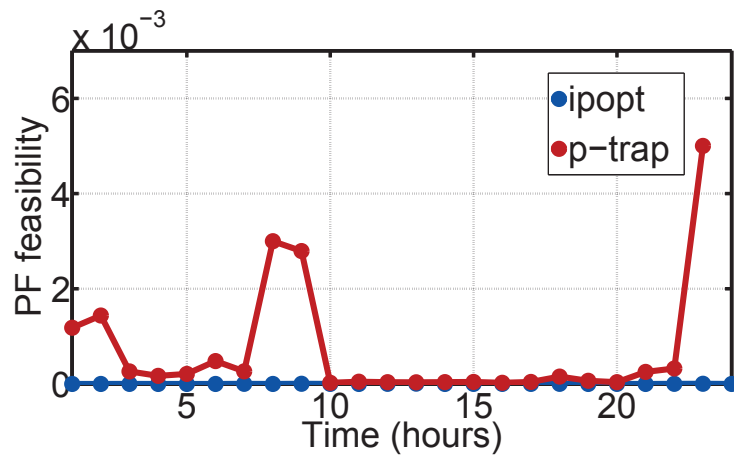


Figure 4.29: Euclidean norm of AC power flow constraints of the 47-bus network obtained by means of IPOPT and Algorithm 7 coupled with 100 iterations of pTRAP.

straints. From our numerical experiments, it can be concluded that the number of trust region iterations can be reduced to a certain level without threatening stability of the tracking scheme. This can be regarded as an advantage for computing an AC-OPF solution in real-time when a sufficiently good warm-start is provided to pTRAP.

Conclusions

This thesis has been focused on the development of numerical methods for solving parametric nonconvex problems based on decomposition and continuation strategies.

In Chapter 2, a novel decomposition algorithm applicable to nonconvex programs has been presented and analysed. It is based on alternating projected gradient steps computed on a sequence of augmented Lagrangian relaxations with nonconvex sublevel sets. A proof of local convergence to a critical point of the NLP has been derived. One of its salient ingredient is the proximal regularisation of the alternating subproblems. We have tested several stopping criteria for the primal alternating minimisations. Even though Eckstein and Silva's stopping criterion appears to be successfully applicable in a nonconvex setting, a heuristic-based criterion shows superior performance in terms of the total number of projected gradient steps on a specific class of nonconvex QPs. In the second part of Chapter 2, we have introduced a novel trust region algorithm, named TRAP, which is applicable to linearly constrained nonlinear problems. In comparison with existing trust region methods, the key difference lies in the Cauchy point computation. The Cauchy point is not generated by a centralised projected search, but by means of alternating projected gradient steps on the model. This ingredient makes the Cauchy phase implementable in a distributed setting. From a distributed perspective, the bottleneck of the algorithm is still the update of the trust region radius, which requires one centralised evaluation of the objective per iteration. However, it is important to stress that in a distributed framework, this can be carried out more easily than a line-search globalisation. Efficacy of the algorithm has been successfully demonstrated on nonconvex optimal power flow problems.

Augmented Lagrangian methods consist of solving inexactly a nonconvex problem at every dual iteration. As a consequence, despite their advantages in large-scale and distributed settings, their applicability in a real-time context seems to be hampered. As an attempt to remedy this issue, Chapter 3 has been devoted to the development and theoretical analysis of continuation algorithms based on the augmented Lagrangian. We have introduced a novel parametric augmented Lagrangian scheme, which consists of a fixed number of primal descent steps and a first-order dual update. The primal steps can be performed via a distributed algorithm, a first-order method or an

active-set trust region method. Four descent algorithms have been introduced. Two of them are first-order schemes, based on either a centralised projected gradient or distributed alternating gradient projections. The two other techniques are a centralised trust-region Newton method and the TRAP algorithm of Chapter 2 for the distributed case. A novelty of the trust region methods presented in this chapter is the proximal regularisation of the subproblem with respect to the Cauchy point, which appears to play a role in the convergence analysis. By combining the Kurdyka-Lojasiewicz with some ingredients of [10], a novel local convergence rate has been derived for the two trust region algorithms. In comparison with the existing literature on trust region Newton methods, its key novelty is that it does not rely on the finite detection of an optimal active-set. This allowed us to establish a local contraction inequality for the parametric augmented Lagrangian algorithm. From this inequality, stability of the tracking scheme has been proven. A novel homotopy mechanism for improving the performance of the optimality-tracking technique has also been proposed and justified by means of the theory. Numerical experiments have been performed on a centralised as well as a distributed example. The theoretical developments explain the numerical results quite well. In particular, the effect of the sampling period on the optimality-tracking performance given a fixed computational power can be explained from the contraction inequality.

In Chapter 4, the efficacy of the continuation algorithms of Chapter 3 has been demonstrated on two examples in the field of optimal control. First, a novel multiple-shooting algorithm has been developed by combining an augmented Lagrangian relaxation of the shooting constraints with a trust region quasi-Newton method. In particular, it has been shown that the augmented Lagrangian offers interesting possibilities from the perspective of sensitivity generation. Using adjoint sensitivity analysis, its gradient with respect to the shooting variables can be generated more efficiently than via the sensitivity analysis techniques applied in SQP-based multiple-shooting. Moreover, we have shown that the complexity of all the computational phases allow for a good scalability when the prediction horizon increases. Our multiple-shooting algorithm has been fully implemented in a C++ software package, named MUTRAL. Its performance has been compared to the ACADO toolkit on a challenging NMPC problem. The results show that MUTRAL outperforms the combinations ACADO-FORCES and ACADO-QPOASES on NMPC problems with long horizons, and provides similar computational performance to ACADO-QPDUNES. It is worth pointing out that the parametric algorithm of MUTRAL can be applied to a larger number of NMPC problems, including economic NMPC programs, as shown in paragraph 4.1.4.2, than ACADO, which is still limited to NMPC problems with least-squares objectives. As a second example, we have tested a parametric version of the distributed algorithm TRAP on multi-stage AC-OPF problems. Results show that the number of trust region iterations per time step can be kept relatively low, while ensuring a reasonable tracking performance. This conclusion is interesting from the perspective of distributed optimisation, as each primal iteration has a cost in terms of communications, which cannot be

neglected in a practical implementation.

Bibliography

- [1] A. Pavlov and N. van de Wouw and H. Nijmeijer. Frequency response functions for nonlinear convergent systems. *IEEE Transactions on Automatic Control*, 52(6):1159–1165, 2007.
- [2] Absil, P.-A. and Mahony, R. and Andrews, B. Convergence of the iterates of descent methods for analytic cost functions. *SIAM Journal on Optimization*, 16(2):531–547, 2005.
- [3] Allgower, E.L. and Georg, K. *Introduction to numerical continuation methods*. SIAM, 1990.
- [4] Alter, T.D. 3D pose from three corresponding points under weak-perspective projection. Technical report, Cambridge, MA, USA, 1992.
- [5] Amberg, B. and Blake, A. and Fitzgibbon, A. and Romdhani, S. and Vetter, T. Reconstructing high quality face-surfaces using model based stereo. In *Proceedings of the International Conference on Computer Vision*, 2007.
- [6] Amestoy, P.R. and Duff, I.S. and Koster, J. and L'Excellent, J.-Y. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.
- [7] Anderson, B.D.O. and Moore, J.B. *Optimal control, Linear quadratic methods*. Prentice-Hall International, 1989.
- [8] Andreani, R. and Birgin, E.G. and Martínez, J.M. and Schuverdt, M.L. On augmented Lagrangian methods with general lower-level constraints. *SIAM Journal on Optimization*, 18(4):1286–1309, 2007.
- [9] Arrillaga, J. and Watson, N.R. *Power system harmonics*. Wiley, 2003.
- [10] Attouch, H. and Bolte, J. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming*, 116:5–16, 2009.

- [11] Attouch, H. and Bolte, J. and Redont, P. and Soubeyran, A. Proximal alternating minimisation and projection methods for nonconvex problems: an approach based on the Kurdyka-Lojasiewicz inequality. *Mathematics of Operations Research*, 35:438–457, 2010.
- [12] Attouch, H. and Bolte, J. and Svaiter, B.F. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting and regularised Gauss-Seidel methods. *Mathematical Programming*, 137:91–129, 2013.
- [13] B. Grünbaum. *Convex polytopes*. Springer, 2003.
- [14] Benders, J.F. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.
- [15] Bernstein, A. and Reyes-Chamorro, L. and Le Boudec, J.-Y. and Paolone, M. A composable method for real-time control of active distribution networks with explicit power setpoints. Part I: Framework. *Electric Power Systems Research*, 125:254–264, 2015.
- [16] Bertsekas, D.P. *Constrained optimization and Lagrange multiplier methods*. Athena Scientific, 1982.
- [17] Bertsekas, D.P. and Tsitsiklis, J.N. *Parallel and distributed computation: numerical methods*. Athena Scientific, 1997.
- [18] Biedl, T. and Hasan, M. and López-Ortiz, A. Reconstructing convex polygons and polyhedra from edge and face counts in orthogonal projections. In *Proceedings of the 27th international conference on Foundations of software technology and theoretical computer science*, FSTTCS’07, pages 400–411, Berlin, Heidelberg, 2007. Springer-Verlag.
- [19] Bock, H.-G. and Plitt, K.-J. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the 9th World Congress*. International Federation of Automatic Control, July 1984.
- [20] Bolte, J. and Daniilidis, A. and Lewis, A. The Lojasiewicz inequality for nonsmooth sub-analytic functions with applications to subgradient dynamical systems. *SIAM Journal on Optimization*, 17:1205–1223, 2007.
- [21] Bolte, J. and Sabach, S. and Teboulle, M. Proximal alternating linearised minimisation for nonconvex and nonsmooth problems. *Mathematical Programming*, 2013.
- [22] Borchardt-Ott, Walter. *Kristallographie*. Springer, Heidelberg, 2008.

- [23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.
- [24] Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge University Press, New York, NY, USA, 2004.
- [25] Buksh, W.A. and Grothey, A. and McKinnon, K.I.M. and Trodden, P.A. Local solutions of the optimal power flow problem. *IEEE Transactions on Power Systems*, 28(4), November 2013.
- [26] J.V. Burke, J.J. Moré, and G. Toraldo. Convergence properties of trust region methods for linear and convex constraints. *Mathematical Programming*, 47:305–336, 1990.
- [27] Byrd, R.H. and Curtis, F.E. and Nocedal, J. An inexact SQP method for equality constrained optimization. *SIAM Journal on Optimization*, 19(1):351–369, 2008.
- [28] Calamai, P.H. and Moré, J.J. Projected gradient methods for linearly constrained problems. *Mathematical Programming*, 39:93–116, 1987.
- [29] M. Chiang, S. Low, A. Calderbank, and J.C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, January 2007.
- [30] Cohen, G. Auxiliary problem principle and decomposition of optimization problems. *Journal of Optimization Theory and Applications*, 32(3):277–305, November 1980.
- [31] Conn, A.R. and Gould, N. and Sartenaer, A. and Toint, P.L. Convergence properties of an augmented Lagrangian algorithm for optimization with a combination of general equality constraints and linear constraints. *SIAM Journal on Optimization*, 6(3):674–703, August 1996.
- [32] Conn, A.R. and Gould, N.I.M and Toint, P.L. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis*, 25(2), 1988.
- [33] Conn, A.R. and Gould, N.I.M. and Toint, P.L. Testing a class of methods for solving minimization problems with simple bounds on the variables. *Mathematics of Computation*, 50(182):399–430, April 1988.

- [34] Conn, A.R. and Gould, N.I.M. and Toint, P.L. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, 28(2):545–572, April 1991.
- [35] Conn, A.R. and Gould, N.I.M. and Toint, P.L. *LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A)*, volume 17 of *Springer Series in Computational Mathematics*. Springer Verlag, Heidelberg, New York, 1992.
- [36] Conn, A.R. and Gould, N.I.M. and Toint, P.L. *Trust Region Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
- [37] Conte, C. and Voellmy, N.R. and Zeilinger, M.N. and Jones, C.N. and Morari, M. Distributed synthesis and control of constrained linear systems. In *Proceedings of the 2012 American Control Conference*, pages 6017–6022, June 2012.
- [38] Cortés, P. and Rodríguez, J. and Quevedo, D.E. and Silva, C. Predictive current control strategy with imposed load current spectrum. *IEEE Transactions on Power Electronics*, 23(2):612–618, March 2008.
- [39] Curtis, F.E. and Gould, N.I.M. and Jiang, H. and Robinson, D.P. Adaptive augmented Lagrangian methods: algorithms and practical numerical experience. Technical Report 14T-006, COR@L Laboratory, Department of ISE, Lehigh University, 2014. To appear in *Optimization Methods and Software*.
- [40] Curtis, F.E. and Johnson, T.C. and Robinson, D.P. and Wächter, A. An inexact sequential quadratic optimization algorithm for nonlinear optimization. *SIAM Journal on Optimization*, 24(3):1041–1074, 2014.
- [41] D. D’Acunto and K. Kurdyka. Effective Lojasiewicz gradient inequality for polynomials. *Annales Polonici Mathematici*, 87, 2005.
- [42] Daniel-Berhe, S. and Unbehauen, H. Experimental physical parameter estimation of a thyristor driven DC-motor using the HMF-method. *Control Engineering Practice*, 6:615–626, 1998.
- [43] Dantzig, G.B. and Wolfe, P. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, January-February 1960.
- [44] Davis, P.F. and Rabinowitz, P. *Methods of numerical integration*. Academic Press, 1984.

-
- [45] D’Azevedo, E. and Eijkhout, V. and Romine, C. LAPACK Working Note 56: Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Technical report, University of Tennessee, Knoxville, TN, USA, 1993.
- [46] Diehl, M. and Bock, H.G. and Schloeder, J.P. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736, 2005.
- [47] Diehl, M. and Ferreau, H.J. and Haverbeke, N. Efficient numerical methods for nonlinear MPC and moving horizon estimation. In *Proceedings of the international workshop on assessment and future directions of NMPC*, Pavia, Italy, September 2008.
- [48] Domahidi, A. and Zraggen, A. and Zeilinger, M.N. and Morari, M. and Jones, C.N. Efficient Interior Point Methods for Multistage Problems Arising in Receding Horizon Control. In *IEEE Conference on Decision and Control*, pages 668–674, Maui, HI, USA, December 2012.
- [49] A.L. Dontchev and R.T. Rockafellar. Characterizations of strong regularity for variational inequalities over polyhedral convex sets. *SIAM Journal on Optimization*, 6(4):1087–1105, 1996.
- [50] Dontchev, A. and Rockafellar, R.T. Parametric stability of solutions in models of economic equilibrium. *Journal of Convex Analysis*, 19:975–993, 2012.
- [51] Dontchev, A.L. and Krastanov, M.I. and Rockafellar, R.T. and Veliov, V.M. An Euler-Newton continuation method for tracking solution trajectories of parametric variational inequalities. *SIAM Journal on Control and Optimization*, 51(3):1823–1840, 2013.
- [52] Dontchev, A.L. and Rockafellar, R.T. *Implicit functions and solution mappings*. Springer, New-York, 2009.
- [53] Douglas, D. and Peucker, T. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10:112–122, 1973.
- [54] Eckstein, J. and Silva, P.J.S. A practical relative error criterion for augmented Lagrangians. *Mathematical Programming*, 141:319–348, 2013.
- [55] Farivar, M. and Clarke, C.R. and Low, S.H. and Chandy, K.M. Inverter VAR control for distribution systems with renewables. *IEEE SmartGrid-Comm*, pages 457–462, 2011.

- [56] Fei, Y. and Guodong, R. and Wang, B. and Wang, W. Parallel L-BFGS-B algorithm on GPU. *Computers and Graphics*, 40:1–9, 2014.
- [57] Fernández, D. and Solodov, M.V. Local convergence of exact and inexact augmented Lagrangian methods under the second-order sufficient optimality condition. *SIAM Journal on Optimization*, 22(2):384–407, 2012.
- [58] Ferreau, H.J. and Bock, H.G. and Diehl, M. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.
- [59] Fletcher, R. and Leyffer, S. Nonlinear programming without a penalty function. *Mathematical Programming, Ser. A*, 91:239–269, 2002.
- [60] Franke, R. and Arnold, E. Applying new numerical algorithms to the solution of discrete-time optimal control problems. In Warwick, K. and Kárný, M., editor, *Computer-intensive methods in control and signal processing: the curse of dimensionality*, chapter 6, pages 105–118. Springer Science and Business Media, New-York, 1997.
- [61] Frasch, J. *Parallel algorithms for optimization of dynamic systems in real-time*. PhD thesis, KU Leuven, 2014.
- [62] Frasch, J.V. and Vukov, M. and Ferreau, H.J. and Diehl, M. A new quadratic programming strategy for efficient sparsity exploitation in SQP-based nonlinear MPC and MHE. In *Proceedings of the 19th IFAC World Congress*, 2014 (to appear).
- [63] Fukuda, K. and Liebling, T.M. and Margot, F. Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron. *Computational Geometry: Theory and Applications*, 8(1):1–12, June 1997.
- [64] Gan, L. and Li, N. and Topcu, U. and Low, S.H. Exact convex relaxation of optimal power flow in radial network. *IEEE Transactions on Automatic Control*, 2014. Accepted for publication.
- [65] Gardner, R. *Geometric Tomography*. Cambridge University Press, New-York, 1995.
- [66] Gardner, R.J. and Gronchi, P. and Theobald, T. Determining a rotation of a tetrahedron from a projection. *Discrete Computational Geometry*, 48:749–765, 2012.
- [67] Gardner, R.J. and Milanfar, P. Reconstruction of convex bodies from brightness function. *Discrete Computational Geometry*, 29:279–303, 2003.

- [68] Garey, M.R. and Johnson, D.S. *Computers and intractability: a guide to the theory of NP-completeness*. W.H. Freeman, San Francisco, 1979.
- [69] Gayme, D. and Topcu, U. Optimal power flow with large-scale storage integration. *IEEE Transactions on Power Systems*, 28(2):709–717, May 2013.
- [70] Giselsson, P. and Boyd, S. Diagonal scaling in Douglas-Rachford splitting and ADMM. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 5040–5045, Los Angeles, CA, December 2014.
- [71] Gondhalekar, R. and Jones, C.N. and Besselmann, T. and Hours, J.-H. and Mercangöz, M. Constrained spectrum control using MPC. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1219–1226, Orlando, USA, 2011.
- [72] Gravin, N. and Lasserre, J. and Pasechnik, D.V. and Robins, S. The inverse moment problem for convex polytopes. *Discrete Computational Geometry*, 48:596–621, 2012.
- [73] Griewank, A. and Toint, P.L. Local convergence analysis of partitioned quasi-Newton updates. *Numerische Mathematik*, 39:429–448, 1982.
- [74] Griewank, A. and Toint, P.L. Partitioned variable metric updates for large structured optimization problems. *Numerische Mathematik*, 39:119–137, 1982.
- [75] Grigore, O. and Veltkamp, R.C. On the implementation of polygonal approximation algorithms. Technical Report UU-CS-2003-005, Department of Information and Computing Sciences, Utrecht University, 2003.
- [76] Grippo, L. and Sciandrone, M. On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. *Operations Research Letters*, 26:127–136, 2000.
- [77] Grüne, L. and Pannek, J. *Nonlinear Model Predictive Control*. Springer-Verlag London, 2011.
- [78] Hairer, E. and Wanner, G. *Solving ordinary differential equations I*. Springer, 2008.
- [79] Hamdi, A. and Mishra, S.K. Decomposition methods based on augmented Lagrangian: a survey. In *Topics in nonconvex optimization*. Mishra, S.K., 2011.
- [80] Herceg, M. and Kvasnica, M. and Jones, C.N. and Morari, M. Multi-parametric toolbox 3.0. In *Proceedings of the European Control Conference*, 2013.

- [81] Hong, M. and Luo, Z.-Q. On the linear convergence of the alternating direction method of multipliers. <http://arxiv.org/abs/1208.3922>, 2013.
- [82] Hornegger, J. and Tomasi, C. Representation issues in the ML estimation of camera motion. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 640–647, 1999.
- [83] Hours, J.-H. and Jones, C.N. A parametric multi-convex technique with application to real-time NMPC. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 5052–5057, Los Angeles, CA, USA, December 2014.
- [84] Hours, J.-H. and Jones, C.N. An augmented Lagrangian coordination-decomposition algorithm for solving distributed nonconvex programs. In *Proceedings of the 2014 American Control Conference*, pages 4312–4317, Portland, Oregon, USA, June 2014.
- [85] Hours, J.-H. and Jones, C.N. An alternating trust region algorithm for distributed linearly constrained nonlinear programs, application to the AC optimal power flow. *Journal of Optimization Theory and Applications*, 2015. Accepted.
- [86] Hours, J.-H. and Jones, C.N. A parametric nonconvex decomposition algorithm for real-time and distributed NMPC. *IEEE Transactions on Automatic Control*, 61(2), February 2016. To appear.
- [87] Hours, J.-H. and Schorsch, S. and Jones, C.N. Parametric polytope reconstruction, an application to crystal shape estimation. *IEEE Transactions on Image Processing*, 23(10):4474–4485, October 2014.
- [88] Hours, J.-H. and Shukla, H. and Jones, C.N. A parametric augmented Lagrangian algorithm for real-time economic NMPC. 2015. Submitted to the 2016 European Control Conference.
- [89] Hours, J.-H. and Zeilinger, M.N. and Gondhalekar, R. and Jones, C.N. Spectrogram-MPC: Enforcing hard constraints on system’s output spectra. In *Proceedings of the American Control Conference*, pages 2010–2017, Montreal, CA, 2012.
- [90] Hours, J.-H. and Zeilinger, M.N. and Gondhalekar, R. and Jones, C.N. Constrained spectrum control. *IEEE Transactions on Automatic Control*, 60(7):1969–1974, July 2015.
- [91] Houska, B. and Ferreau, H.J. and Diehl, M. An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. *Automatica*, 47:2279–2285, 2011.

-
- [92] Houska, B. and Logist, F. and Van Impe, J. and Diehl, M. Approximate robust optimization of time-periodic stationary states with application to biochemical processes. In *Proceedings of the joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P.R. China, December 2009.
- [93] Huneault, M. and Galiana, F.D. An investigation of the solution to the optimal power flow problem incorporating continuation methods. *IEEE Transactions on Power Systems*, 5(1):103–110, February 1990.
- [94] Huttenlocher, D.P. and Klanderman, G.A. and Rucklidge, W.J. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9), September 1993.
- [95] Jakubiec, F.Y. and Ribeiro, A. D-MAP: Distributed Maximum a Posteriori Probability Estimation of Dynamic Systems. *IEEE Transactions on Signal Processing*, 61(2):450–466, 2013.
- [96] Jones, C.N. *Polyhedral tools for control*. PhD thesis, University of Cambridge, 2005.
- [97] Jozs, C. and Maeght, J. and Panciatici, P. and Gilbert, J.-C. Application of the moment-SOS approach to global optimization of the OPF problem. *IEEE Transactions on Power Systems*, 30(1), January 2015.
- [98] Kameswaran, S. and Biegler, L.T. Convergence rates for direct transcription of optimal control problems using collocation at Radau points. *Computational Optimization and Applications*, 41:81–126, 2008.
- [99] Kiehl, M. Parallel multiple shooting for the solution of initial value problems. *Parallel Computing*, 20:275–295, 1994.
- [100] Kim, B.H. and Baldick, R. Coarse-grained distributed optimal power flow. *IEEE Transactions on Power Systems*, 12(2), May 1997.
- [101] Kirches, C. and Wirsching, L. and Bock, H.G. and Schlöder, J.P. Efficient direct multiple shooting for nonlinear model predictive control on long horizons. *Journal of Process Control*, 22(3):540–550, 2012.
- [102] Kögel, M. and Findeisen, R. Stability of NMPC with cyclic horizons. In *Proceedings of the 9th IFAC Symposium on Nonlinear Control Systems*, pages 809–814, September 2013.

- [103] Kozma, A. and Klintberg, E. and Gros, S. and Diehl, M. An improved distributed dual Newton-CG method for convex quadratic programming problems. In *Proceedings of the 2014 American Control Conference*, pages 2324–2329, Portland, OR, June 2014.
- [104] Lam, A.Y.S. and Zhang, B. and Tse, D.N. Distributed algorithms for optimal power flow. In *Proceedings of the 51st Conference on Decision and Control*, pages 430–437, December 2012.
- [105] Lam, A.Y.S. and Zhang, B. and Tse, D.N. Distributed algorithms for optimal power flow problem. In *Proceedings of the 51st Conference on Decision and Control*, pages 430–437, December 2012.
- [106] Larsen, P.A. and Patience, D.B. and Rawlings, J.B. Industrial crystal size, shape and structure. *IEEE Control Systems Magazine*, August 2006.
- [107] Larsen, P.A. and Rawlings, J.B. and Ferrier, N.J. Model-based object recognition to measure crystal size and shape distributions from in-situ video images. *Chemical Engineering Science*, 62:1430–1441, 2007.
- [108] Lavaei, J. and Low, S. Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 27(1):92–107, February 2012.
- [109] Leineweber, D.B. Analyse und Restrukturierung eines Verfahrens zur direkten Lösung von Optimal-Steuerungsproblemen. Master’s thesis, Universität Heidelberg, 1995.
- [110] Lin, C.-J. and Moré, J.J. Newton’s method for large bound-constrained optimization problems. *SIAM Journal on Optimization*, 9(4):1100–1127, 1999.
- [111] Mangasarian, O.L. Solution of symmetric linear complementarity problems by iterative methods. *Journal of Optimization Theory and Applications*, 22(4), August 1977.
- [112] Marlin, B. and Toussaint, G. Constructing convex 3-polytopes from two triangulations of a polygon. *Comput. Geom. Theory Appl.*, 28(1):41–47, May 2004.
- [113] Mayne, D.Q. and Rawlings, J.B. and Rao, C.V. and Scokaert, P.O.M. Constrained model predictive control: stability and optimality. *Automatica*, 36(6):789–814, 2000.
- [114] Michalska, H. and Mayne, D.Q. Robust receding horizon control of constrained systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.
- [115] Moré, J.J. *Trust regions and projected gradients*, volume 113 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1988.

-
- [116] Moré, J.J. and Toraldo, G. On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1(1):93–113, February 1991.
- [117] J.J. Moreau. Décomposition orthogonale d'un espace hilbertien selon deux cônes mutuellement polaires. *C.R. Académie des Sciences*, 255:238–240, 1962.
- [118] Necoara, I. and Savorgnan, C. and Tran Dinh, Q. and Suykens, J. and Diehl, M. Distributed nonlinear optimal control using sequential convex programming and smoothing techniques. In *Proceedings of the 48th Conference on Decision and Control*, 2009.
- [119] Nocedal, J. and Wright, S. *Numerical optimization*. Springer, New-York, 2006.
- [120] Oppenheim, A.V. and Schaffer, R.W. *Discrete-time signal processing*. Prentice Hall, 1999.
- [121] Parulekar, S.J. Analysis of forced periodic operations of continuous bioprocesses - single input variations. *Chemical Engineering Science*, 53(14):2481–2502, 1998.
- [122] Peng, Z.K. and Lang, Z.Q. and Billings, S.A. Resonances and resonant frequencies for a class of nonlinear systems. *Journal of Sound and Vibration*, 300:993–1014, 2007.
- [123] Powell, M.J. On search directions for minimization algorithms. *Mathematical Programming*, 4:193–201, 1973.
- [124] Presles, B. and Debayle, J. and Pinoli, J.-C. Size and shape estimation of 3-D convex objects from their 2-D projections: applications to crystallisation processes. *Journal of Microscopy*, 248:140–155, 2012.
- [125] Presles, B. and Debayle, J. and Rivoire, A. and Févotte, G. and Pinoli, J.C. Monitoring the particle size distribution using image analysis during batch crystallization processes. In *9th IEEE/SPIE International Conference on Quality Control by Artificial Vision (QCAV)*, Wels, Austria, May 2009.
- [126] Preumont, A. *Vibration control of active structures: An introduction*. Springer, 2002.
- [127] Quevedo, D.E. and Bölcskei, H. and Goodwin, G.C. Quantisation of filter bank frame expansions through moving horizon optimisation. *IEEE Transactions on Signal Processing*, 57(2):503–515, 2009.
- [128] Quirynen, R. and Houska, B. and Vallerio, M. and Telen, D. and Logist, F. and Van Impe, J. and Diehl, M. Symmetric algorithmic differentiation based exact hessian SQP method and software for economic MPC. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 2752–2757, Los Angeles, CA, USA, December 2014.

- [129] Quirynen, R. and Houska, B. and Vallerio, M. and Telen, D. and Logist, F. and Van Impe, J. and Diehl, M. Symmetric algorithmic differentiation based exact hessian SQP method and software for economic MPC. In *Proceedings of the 53rd Conference on Decision and Control*, pages 2752–2757, Los Angeles, CA, USA, December 2014.
- [130] Quirynen, R. and Vukov, M. and Zanon, M. and Diehl, M. Autogenerating microsecond solvers for nonlinear MPC: A tutorial using ACADO integrators. *Optimal Control Applications and Methods*, 2014.
- [131] Rangarajan, A. and Chui, H. and Bookstein, F.L. The softassign Procrustes matching algorithm. In *Information Processing in Medical Imaging*, pages 29–42. Springer, 1997.
- [132] Rawlings, J.B. and Mayne, D.Q. *Model predictive control: Theory and design*. Nob Hill Publishing, 2009.
- [133] Reddien, G.W. Collocation at Gauss points as a discretization in optimal control. *SIAM Journal on Control and Optimization*, 17(2), March 1979.
- [134] Richter, S. and Mariéthoz, S. and Morari, M. High-speed online MPC based on a fast gradient method applied to power converter control. In *Proceedings of the 2010 American Control Conference*, pages 4737–4743, Baltimore, MD, USA, June-July 2010.
- [135] Robinson, D.P. *Primal-dual methods for nonlinear optimization*. PhD thesis, UC San Diego, 2007.
- [136] Robinson, S.B. and Hemler, P.F. and Webber, R. A geometric problem in medical imaging. In Wilson, D.C. and Tagare, H.D. and Bookstein, F.L. and Preteux, F.J. and Dougherty, E.R., editor, *Mathematical Modeling, Estimation and Imaging*, pages 208–217. SPIE, 2000.
- [137] Robinson, S.M. Strongly regular generalised equations. *Mathematics of Operations Research*, 5(1):43–62, 1980.
- [138] Rockafellar, R.T. and Wets, R.J.-B. *Variational analysis*. Springer, 2009.
- [139] Saad, Y. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.
- [140] Schmidt, J. and Niemann, H. Using quaternions for parametrizing 3-D rotations in unconstrained nonlinear optimization. In *Proceedings of the vision modeling and visualization conference 2001, VMV '01*, pages 399–406. Aka GmbH, 2001.

-
- [141] Schorsch, S. and Hours, J.-H. and Vetter, T. and Mazzotti, M. and Jones, C.N. An optimization-based approach to extract faceted crystal shapes from stereoscopic images. *Computers and Chemical Engineering*, 75:171–183, 2015.
- [142] Shi, W. and Ling, Q. and Yuan, K. and Wu, G. and Yin, W. On the linear convergence of the ADMM in decentralised consensus optimization. *IEEE Transactions on signal processing*, 62(7), April 2014.
- [143] Singh, M.R. and Chakraborty, J. and Nere, N. and Tsung, H.-H. and Bordawekar, S. and Ramkrishna, D. Image-analysis-based method for 3D crystal morphology measurement and polymorph identification using confocal microscopy. *Crystal growth and design*, 12:3735–3748, December 2012.
- [144] Soussen, C. and Mohammad-Djafari, A. Polygonal and polyhedral contour reconstruction in computed tomography. *IEEE Transactions on Image Processing*, 13:1507–1523, 2004.
- [145] Steihaug, T. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983.
- [146] Suzuki, S. and Abe, K. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics and image processing*, 30:32–46, 1985.
- [147] Tran Dinh, Q. and Necoara, I. and Diehl, M. A dual decomposition algorithm for separable nonconvex optimization using the penalty framework. In *Proceedings of the 52nd Conference on Decision and Control*, 2013.
- [148] Tran Dinh, Q. and Savorgnan, C. and Diehl, M. Adjoint-based predictor-corrector sequential convex programming for parametric nonlinear optimization. *SIAM Journal on Optimization*, 22(4):1258–1284, 2012.
- [149] Tran-Dinh, Q. and Savorgnan, C. and Diehl, M. Combining Lagrangian decomposition and excessive gap smoothing technique for solving large-scale separable convex optimization problems. *Computational Optimization and Applications*, 55(1):75–111, 2013.
- [150] Tseng, P. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, June 2001.
- [151] Verschoor, M. and Jalba, A.C. Analysis and performance estimation of the Conjugate Gradient method on multiple GPUs. *Parallel Computing*, 38(10-11):552–575, 2012.

- [152] Wächter, A. and Biegler, L.T. On the implementation of a primal-dual interior point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.
- [153] Wirsching, L. and Ferreau, H.J. and Bock, H.G. and Diehl, M. An online active set strategy for fast adjoint based nonlinear model predictive control. In *Proceedings of the 7th Symposium on Nonlinear Control Systems*, pages 164–169, 2007.
- [154] Xue, D. and Sun, W. and Qi, L. An alternating structured trust-region algorithm for separable optimization problems with nonconvex constraints. *Computational Optimization and Applications*, 57:365–386, 2014.
- [155] Yamashita, N. Sparse quasi-Newton updates with positive definite matrix completion. *Mathematical Programming*, 115(1):1–30, September 2008.
- [156] Yan, Q.-W. and Chen, C.L.P. and Tang, Z. Efficient algorithm for the reconstruction of 3D objects from orthographic projections. *Computer-aided Design*, 26:699–717, 1994.
- [157] Zavala, V.M. and Anitescu, M. Real-time nonlinear optimization as a generalised equation. *SIAM Journal on Control and Optimization*, 48(8):5444–5467, 2010.
- [158] Zavala, V.M. and Anitescu, M. Scalable nonlinear programming via exact differentiable penalty functions and trust-region Newton methods. *SIAM Journal on Optimization*, 24(1):528–558, 2014.
- [159] Zavala, V.M. and Biegler, L.T. The advanced-step NMPC controller: optimality, stability and robustness. *Automatica*, 45(1):86–93, 2009.
- [160] Zavala, V.M. and Laird, C.D. and Biegler, L.T. Fast solvers and rigorous models: Can both be accommodated in NMPC ? In *Proceedings of the IFAC Workshop on NMPC for Fast Systems*, 2006.
- [161] Zavala, V.M. and Laird, C.D. and Biegler, L.T. Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems. *Chemical Engineering Science*, 63:4834–4845, 2008.
- [162] Zhu, J. *Optimization of Power System Operation*. IEEE Press, Piscataway, NJ, USA, 2009.
- [163] Ziegler, G.M. *Lectures on polytopes*. Springer-Verlag, 1995.

- [164] Zimmerman, Z. and Murillo-Sánchez, C.E. and Thomas, R.J. Matpower: Steady-state operations, planning and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26:12–19, 2011.

Appendix A

Network data

Origin bus	End bus	Real part of line impedance r (Ω)	Imaginary part of line impedance x (Ω)
1	2	0.259	0.808
2	13	0	0.001
2	3	0.031	0.092
3	4	0.046	0.092
3	14	0.092	0.031
3	15	0.214	0.046
4	20	0.336	0.061
4	5	0.107	0.183
5	26	0.061	0.015
5	6	0.015	0.031
6	27	0.168	0.061
6	7	0.031	0.046
7	32	0.076	0.015
7	8	0.015	0.015
8	40	0.046	0.015
8	39	0.244	0.046
8	41	0.107	0.031
8	35	0.076	0.015
8	9	0.031	0.031
9	10	0.015	0.015
9	42	0.153	0.046
10	11	0.107	0.076
10	46	0.229	0.122
11	47	0.031	0.015
11	12	0.076	0.046
15	18	0.046	0.015
15	16	0.107	0.015
16	17	0	0.001
18	19	0	0.001
20	21	0.122	0.092
20	25	0.214	0.046
21	24	0	0.001
21	22	0.198	0.046
22	23	0	0.001
27	31	0.046	0.015
27	28	0.107	0.031
28	29	0.107	0.031
29	30	0.061	0.015
32	33	0.046	0.015
33	34	0.031	0.010
35	36	0.076	0.015
35	37	0.076	0.046
35	38	0.107	0.015
42	43	0.061	0.015
43	44	0.061	0.015
43	45	0.061	0.015

Table A.1: Line data of 47-bus network [55].

Appendix B

Constrained Spectrum Control

B.1 Introduction

Many methods for system analysis and controller design commonly applied in industry are typically based on the system's response to exogenous harmonic excitations at specific frequencies. In the *linear time-invariant* (LTI) case, the closed-loop behavior of the system in terms of performance and robustness is closely related to its harmonic response. However, in the case of constrained and nonlinear systems, even though it is possible to achieve specific control objectives such as tracking or stabilisation, the system's response to excitations at specific frequencies is difficult to characterise and even harder to control. Recently, the harmonic response of convergent nonlinear systems was analysed via the Frequency Response Function in [1], yet to the authors' knowledge, this approach does not provide any controller design method.

A standard approach for the control of constrained systems is *Model Predictive Control* (MPC) [132], but most MPC approaches do not facilitate designing the harmonic response of the closed-loop system. Recent work on power converters has shown that frequency information can be incorporated into an MPC optimisation problem for the purpose of reducing the harmonics level in an output signal. In [38], the spectrum of the load current is shaped by using a band-pass filter and by penalising the filter output in the cost function of an MPC problem. In this chapter, we propose an MPC method for shaping the harmonic response of a constrained nonlinear system. We extend the idea of loop-shaping *linear-quadratic regulator* (LQR) techniques [7] by defining spectrum constraints, which are enforced within a receding-horizon optimal control problem. Our approach is targeted towards band-wise spectrum constraints. With the proposed method, a given frequency band can be kept below a certain level while the system is operating in closed-loop and the typically considered pointwise-in-time input and output constraints are guaranteed to also be enforced. The damping effect can be captured by computing the time-localised spectrum of the output signal using the STFT, for instance. Therefore, in this chapter, the frequency shaping is performed by con-

straining the squared magnitude of the STFT, called the *spectrogram*. In this context the notion of a system's harmonic response is based solely on the system's output trajectory. Thus, constrained nonlinear systems can be accommodated, despite the standard notion of a transfer function not being appropriate.

In this chapter, the constrained spectrum control approach first proposed in [71, 89] for LTI systems is extended to constrained nonlinear systems. Conditions for recursive feasibility and stability of the proposed spectrum constrained NMPC scheme are derived via an ellipsoidal invariant set that ensures satisfaction of the constraints on the spectrogram as well as the standard pointwise-in-time state and input constraints, and that can be computed using *semidefinite-programming* (SDP). Finally, the efficacy of the proposed approach is demonstrated on a nonlinear oscillator with hard constraints on the spectrogram as well as the usual pointwise-in-time state and input constraints.

B.2 Notation

We denote by $\rho(A)$ the spectral radius of a matrix A , and by $\mathbb{1}$ a vector with all elements equal to 1. Both the Euclidian 2-norm in \mathbb{R}^n and the induced 2-norm in $\mathbb{R}^{n \times n}$ are denoted by $\|\cdot\|_2$. The open ball centred at a point $a \in \mathbb{R}^n$ with radius $r > 0$ is denoted as $\mathcal{B}(a, r)$. Given a positive definite matrix M and a positive scalar β , we define the ellipsoid $\mathcal{E}(M, \beta) := \{x \in \mathbb{R}^n : \langle x, Mx \rangle \leq \beta\}$. The set of square-integrable functions from a segment $[a, b]$ to \mathbb{C} is denoted as $L^2([a, b], \mathbb{C})$. The sets of strictly positive and strictly negative integers are denoted by \mathbb{Z}_+ and \mathbb{Z}_- , respectively.

B.3 Spectrum constrained NMPC

In this section we demonstrate how frequency features can be incorporated into a receding horizon optimal control problem by constraining the magnitude of a filter output. The proposed approach is targeted at nonlinear systems and builds upon the ideas introduced in [89] for the LTI case.

B.3.1 Problem formulation

Consider a discrete-time constrained nonlinear system

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) , \\ x_k &\in \mathbb{X} , u_k \in \mathbb{U} , \end{aligned} \tag{B.1}$$

where $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$. The constraint sets \mathbb{X} and \mathbb{U} are assumed to be polyhedral and to contain the origin in their interiors.

Assumption B.1. *The mapping f is twice continuously differentiable and $f(0, 0) = 0$.*

Let the linearisation about the origin be defined as

$$x_{k+1}^{(L)} = A_L x_k^{(L)} + B_L u_k . \quad (\text{B.2})$$

Assumption B.2 (Stabilisability). *The pair (A_L, B_L) is stabilisable.*

Let K be a linear state-feedback gain that stabilises the linearised system (B.2), implying that the origin is locally exponentially stable under $x_{k+1} = f(x_k, Kx_k)$, that is

$$\exists r > 0, \exists c_1 > 0, \exists \gamma \in]0, 1[\text{ such that } \|x_0\|_2 < r \implies \forall k \geq 0, \|x_k\|_2 \leq c_1 \gamma^k \|x_0\|_2 . \quad (\text{B.3})$$

In the sequel, we denote $\bar{A}_L := A_L + B_L K$. Note that a gain K such that $\rho(\bar{A}_L) < 1$ exists by Assumption B.2. The system's output, whose frequency components are to be constrained, is defined as follows:

$$\begin{cases} \forall k \in \mathbb{Z}_+, z_k := Cx_k + Du_k \\ \forall k \in \mathbb{Z}_-, z_k := 0 , \end{cases} \quad (\text{B.4})$$

where $C \in \mathbb{R}^{1 \times n}$ and $D \in \mathbb{R}^{1 \times m}$.

Remark B.1. *For clarity of the presentation, we consider constraints involving a single output, although the extension to multiple outputs is direct. Note that the output (B.4) may describe an actual system output, or any linear combination of actual outputs, system states, and control inputs.*

The spectral content of the output signal $\{z_k\}$ is constrained via a design parameter

$$\mathcal{F} \in L^2([-\pi, \pi], \mathbb{C})$$

called a *frequency profile*, as shown in (B.7). It is assumed that such a frequency profile $\mathcal{F}(\omega)$ can be defined as the Fourier transform of the impulse response of an LTI filter

$$\begin{cases} \xi_{k+1} = \mathcal{A}\xi_k + \mathcal{B}z_k \\ \psi_k = \mathcal{C}\xi_k + \mathcal{D}z_k , \end{cases} \quad (\text{B.5})$$

where $\mathcal{A} \in \mathbb{R}^{q \times q}$, $\mathcal{B} \in \mathbb{R}^{q \times 1}$, $\mathcal{C} \in \mathbb{R}^{1 \times q}$, and $\mathcal{D} \in \mathbb{R}$.

Assumption B.3 (Stability and observability). *The pair $(\mathcal{C}\mathcal{A}, \mathcal{A})$ is observable and $\rho(\mathcal{A}) < 1$.*

Remark B.2. For the LTI filter (B.5), either a Finite Impulse Response (FIR) filter or a stable Infinite Impulse Response (IIR) filter can be chosen. While FIR filters are simple to implement, and stable, filters with higher selectivity require higher filter orders, which results in larger state dimensions. In contrast, IIR filters can be designed to be more selective for smaller filter orders, i.e. state dimension. However, in both cases, increasing the frequency resolution requires observing the signal for an increasing time length. In the remainder of the chapter we use FIR filters, although the methodology can be applied immediately when using stable IIR filters.

Time-localised spectrum constraints are based on a windowing of the output signal $\{z_k\}$. A window is defined by its length $M \in \mathbb{Z}_+$ and a windowing signal $\{f_p\}$, $p \in \mathbb{Z}$ that satisfies $f_p = 0$ if $|p| > M$. Choosing an appropriate time-domain window allows one to mitigate spectral leakage caused by the finite signal length [9]. Windows that tend to zero at the boundaries of the selected time interval, such as the Hamming window, are a good way to mitigate this problem.

The salient ingredient of the spectrum constrained MPC formulation derived in the sequel is the STFT $\mathcal{Z}(\omega, \tau)$ of the windowed signal $\{z_k\}$ at time $\tau \in \mathbb{Z}$:

$$\mathcal{Z}(\omega, \tau) := \sum_{i=-\infty}^{+\infty} z_i f_{i-\tau} e^{-j\omega i} . \quad (\text{B.6})$$

The goal is to constrain the amplitude of frequency components of the signal $\{z_k\}$ to lie in a given frequency band $[\omega_L, \omega_U]$. This is achieved by enforcing hard constraints on the STFT $\mathcal{Z}(\omega, \tau)$ weighted by the frequency profile $\mathcal{F}(\omega)$ in a receding horizon optimal control problem. Such constraints are described via the *spectrogram* of $\{z_k\}$, or more precisely via

$$S(\tau) := \frac{1}{2\pi} \int_{-\pi}^{\pi} |\mathcal{F}(\omega) \mathcal{Z}(\omega, \tau)|^2 d\omega , \quad (\text{B.7})$$

where $\tau \in \mathbb{Z}$. A spectrogram constraint at prediction time τ involves samples from time $\tau - M$ until time $\tau + M$. Compared to a standard MPC set-up, the model prediction is therefore extended before prediction time 0 and after prediction time N . The resulting spectrum constrained NMPC problem is formulated as follows

$$\underset{u_0, \dots, u_{N-1}}{\text{minimise}} \sum_{p=0}^{N-1} l(x_p, u_p) + V_N(x_N) \quad (\text{B.8a})$$

subject to :

System dynamics on $\{0, \dots, N + 2M\}$

$$x_{p+1} = f(x_p, u_p) \quad \forall p \in \{0, \dots, N - 1\} \quad (\text{B.8b})$$

$$x_{p+1} = f(x_p, Kx_p) \quad \forall p \in \{N, \dots, N + 2M\} \quad (\text{B.8c})$$

$$z_p = Cx_p + Du_p \quad \forall p \in \{0, \dots, N - 1\} \quad (\text{B.8d})$$

$$z_p = (C + DK)x_p \quad \forall p \in \{N, \dots, N + 2M\} \quad (\text{B.8e})$$

Spectrogram constraints on $\{-M, \dots, N + M\}$

$$S(p) \leq \alpha \quad \forall p \in \{-M, \dots, N + M\} \quad (\text{B.8f})$$

Polyhedral constraints on $\{0, \dots, N - 1\}$

$$x_p \in \mathbb{X}, u_p \in \mathbb{U} \quad \forall p \in \{0, \dots, N - 1\} \quad (\text{B.8g})$$

Terminal constraint

$$x_N \in \mathbb{S}, \quad (\text{B.8h})$$

where $l(\cdot, \cdot)$ is a continuous positive-definite stage-cost and $\mathbb{S} \subset \mathbb{R}^n$ is an appropriate compact invariant set under the nonlinear dynamics (B.1), derived in Section B.4. The terminal penalty V_N is assumed to be continuous positive-definite and to satisfy the following standard terminal cost decrease assumption

Assumption B.4 (Terminal cost decrease). *For all x in the terminal set \mathbb{S} ,*

$$V_N(f(x, Kx)) - V_N(x) \leq -l(x, Kx) . \quad (\text{B.9})$$

Remark B.3. *The solution of the spectrum constrained NMPC program (B.8) depends on the pre-designed stabilising control law K .*

Remark B.4. *Throughout the rest of the text, prediction steps are indexed using the p , while steps of the closed-loop system are indexed by k .*

The two main challenges of the proposed spectrum constrained NMPC scheme are the derivation of a tractable formulation of program (B.8), specifically the spectrum constraint (B.8f), and the computation of the terminal constraint set (B.8h). Next, it is shown that a convex quadratic formulation of the constraints (B.8f) can be derived. As this step has been described in detail in [89], only the main result is stated in this chapter.

Remark B.5. *At first glance it may seem that the spectrogram constraint leads to a time-dependent control law, as for each given state the spectrogram is a function of past states. However, the spectrum constrained NMPC problem can be reformulated to provide a time invariant control law by considering an augmented system with a state including both the actual system state and the relevant portion of the output history.*

B.3.2 Properties of the closed-loop spectrum

The spectrogram constraint (B.8f) enforces that the predicted output trajectory $\{z_p\}$, $p = 0, \dots, N$, contributes to the *entire*, past and present, output trajectory, in such a manner that the spectrogram constraint $S(\tau)$ of (B.7) is satisfied for all $\tau \in \mathbb{Z}$. Thus, the spectrum of the entire closed-loop system output $\{z_k\}$, $k \in \mathbb{Z}_+$, is constrained in a time-localised fashion. Assuming that the spectrum constrained NMPC problem (B.8) is recursively feasible, which is proven later in Section B.4, this notion of closed-loop spectrum shaping is formalised in Theorem B.1.

Theorem B.1. *If Problem (B.8) is recursively feasible, then for any $\tau \geq 0$, $S(\tau) \leq \alpha$, where $S(\cdot)$ is computed on the output of system (B.1) in closed-loop with the optimal control law obtained from (B.8).*

Proof. This is a direct consequence of the spectrogram constraint $S(-M) \leq \alpha$ in (B.8), which incorporates only the first state in the model prediction, and recursive feasibility of (B.8), proven in Theorem B.3. \square

B.3.3 Tractability of spectrum constraints

The key ingredient for the convex quadratic reformulation of the spectrogram constraint is Parseval's theorem [120], which is applied to the output signal of the filter at every prediction instant. This allows for the transformation of a spectrogram constraint into an infinite horizon time-domain constraint, of which a finite horizon formulation can be obtained. Such a technique has been applied in a filter weighting context in [127].

Theorem B.2 ([89], Quadratic spectrum constraints). *For any time $\tau \geq 0$,*

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} |F(\omega) \mathcal{Z}(\omega, \tau)|^2 d\omega = \sum_{k=\tau-M}^{\tau+M} \left\langle \begin{pmatrix} \xi_k^\tau \\ z_k \end{pmatrix}, P_k \begin{pmatrix} \xi_k^\tau \\ z_k \end{pmatrix} \right\rangle + \langle \xi_{\tau+M}^\tau, \mathcal{P} \xi_{\tau+M}^\tau \rangle \quad (\text{B.10})$$

where

- $\{\xi_k^\tau\}$ is the sequence of states of the filter (B.5) under input $\{f_{k-\tau} z_k\}$, the output signal windowed around time τ . Without loss of generality, we assume that $\xi_{\tau-M}^\tau = 0$.

- $\forall k \in \{\tau - M, \dots, \tau + M\}$,

$$P_k := \begin{pmatrix} \mathcal{C}^\top \\ f_{k-\tau} \mathcal{D}^\top \end{pmatrix} \begin{pmatrix} \mathcal{C} & f_{k-\tau} \mathcal{D} \end{pmatrix} \succeq 0 . \quad (\text{B.11})$$

- $\mathcal{P} \succ 0$ is the unique solution of the discrete-time Lyapunov equation

$$\mathcal{P} = (\mathcal{C}\mathcal{A})^\top \mathcal{C}\mathcal{A} + \mathcal{A}^\top \mathcal{P}\mathcal{A} , \quad (\text{B.12})$$

that exists by Assumption B.3.

As a result, the spectrum constrained NMPC problem (B.8) can be reformulated as a quadratically constrained nonlinear program, which can subsequently be solved using nonlinear interior-point solvers, such as IPOPT [152].

B.4 Recursive feasibility of spectrum constrained NMPC

In this section, an invariant set for the dynamics $x_{k+1} = f(x_k, Kx_k)$ is derived. This set ensures recursive feasibility of NMPC problem (B.8). The standard notion of an invariant set must be adapted to the added requirements imposed by the spectrogram, since the terminal constraint should not only be invariant in order to ensure recursive feasibility, but also, containment of the predicted state x_p in the terminal set \mathbb{S} at time $p \geq N$ should guarantee satisfaction of the spectrogram constraint at time $p + M$, as a spectrogram constraint involves M samples backwards in time [89]. In the sequel, we propose that an invariant set for the nonlinear dynamics (B.1) with these properties, can be computed by solving an SDP. The derivation of such an invariant set is performed in Lemmas B.1, B.2, B.3 and B.4. The main result is stated in Theorem B.3. Stability of the closed-loop system under the spectrum constrained NMPC control law then follows from a standard optimal cost decrease argument.

The following Lemma shows that the spectrogram computed at time $p + M$ for the nonlinear dynamics is upper-bounded by the sum of a quadratic function of $\|x_p\|_2$, depending on the filter matrices $(\mathcal{A}, \mathcal{B})$ and the linearised model. By choosing x_p small enough, the difference between the spectrogram of the output of the linearised system and the output of the nonlinear dynamics can be made arbitrarily small. Let $g(x) := f(x, Kx) - \bar{A}_L x$. By Assumption B.1, g satisfies $g(0) = 0$, and $\|g(x)\|_2 / \|x\|_2 \rightarrow 0$ when $x \rightarrow 0$.

Lemma B.1. *Given r satisfying (B.3), there exists a constant $c > 0$ such that for all $p \geq N$*

$$x_p \in \mathcal{B}(0, r) \implies S(p + M) \leq \langle x_p, \mathcal{R}x_p \rangle + c \|x_p\|_2^2 , \quad (\text{B.13})$$

where \mathcal{R} is defined as

$$\mathcal{R} := \mathcal{H}_{2M}^\top \begin{pmatrix} \mathcal{P} & 0 \\ 0 & 0 \end{pmatrix} \mathcal{H}_{2M} + \sum_{l=0}^{2M-1} \mathcal{H}_l^\top P_l \mathcal{H}_l \quad (\text{B.14})$$

with P_l defined in (B.11) and

$$\mathcal{H}_l := \begin{pmatrix} \sum_{k=0}^{l-1} \mathcal{A}^k \mathcal{B} (C + DK) \bar{A}_L^{l-k-1} \\ (C + DK) \bar{A}_L^l \end{pmatrix} .$$

for $l \in \{0, \dots, 2M\}$.

Proof. It holds that $x_p \in \mathcal{B}(0, r)$. For $i \in \mathbb{N}$, $i \geq 1$, define

$$h(x_p, \dots, x_{p+i-1}) := \sum_{j=0}^{i-1} \bar{A}_L^j g(x_{p+i-1-j}) , \quad (\text{B.15})$$

where the sequence of states $\{x_p, \dots, x_{p+i-1}\}$ is obtained by applying the nonlinear dynamics $x_{k+1} = f(x_k, Kx_k)$ to x_p . Since $x_p \in \mathcal{B}(0, r)$, $\|g(x_{p+i-1-j})\|_2$ can be bounded by a linear function in $\|x_p\|_2$, using exponential stability (B.3) of the origin under the control law K and applying the triangle inequality:

$$\|g(x_{p+i-1-j})\|_2 \leq c_1 \gamma^{i-1-j} (1 + \|\bar{A}_L\|_2) \|x_p\|_2 , \quad (\text{B.16})$$

where $\gamma \in]0, 1[$ is defined in (B.3). From the triangle inequality,

$$\|h(x_p, \dots, x_{p+i-1})\|_2 \leq \sum_{j=0}^{i-1} \|\bar{A}_L^j\|_2 \|g(x_{p+i-1-j})\|_2 ,$$

which implies that for all $i \geq 1$, there exists $\eta(i) > 0$ such that

$$\|h(x_p, \dots, x_{p+i-1})\|_2 \leq \eta(i) \|x_p\|_2 . \quad (\text{B.17})$$

From Theorem B.2, there exist matrices $\tilde{P}_k \in \mathbb{R}^{n(k-p+1) \times n(k-p+1)}$ such that

$$S(p+M) = \sum_{k=p}^{p+2M} \left\langle X_{p:k}, \tilde{P}_k X_{p:k} \right\rangle , \quad (\text{B.18})$$

where $X_{p:k} := (x_p^\top, \dots, x_k^\top)^\top$ for $k \in \{p, \dots, p+2M\}$. For $k \in \{p+1, \dots, p+2M\}$, by writing

$x_k = \bar{A}_L^{k-p} x_p + h(x_p, \dots, x_{k-1})$, one obtains

$$S(p+M) - \langle x_p, \mathcal{R}x_p \rangle = \sum_{k=p+1}^{p+2M} 2 \langle X_{p:k}^{(\bar{A}_L)}, \tilde{P}_k X_{p:k}^{(h)} \rangle + \sum_{k=p+1}^{p+2M} \langle X_{p:k}^{(h)}, \tilde{P}_k X_{p:k}^{(h)} \rangle ,$$

where

$$\begin{aligned} X_{p:k}^{(\bar{A}_L)} &:= \left(x_p^\top, (\bar{A}_L x_p)^\top, \dots, (\bar{A}_L^{k-p} x_p)^\top \right)^\top , \\ X_{p:k}^{(h)} &:= \left(0^\top, h(x_p)^\top, \dots, h(x_p, \dots, x_{k-1})^\top \right)^\top \end{aligned}$$

and \mathcal{R} is defined in (B.14). It then follows that

$$S(p+M) - \langle x_p, \mathcal{R}x_p \rangle \leq \max_k \left\| \tilde{P}_k \right\|_2 \sum_{k=p+1}^{p+2M} \left\| X_{p:k}^{(h)} \right\|_2 \left(2 \left\| X_{p:k}^{(\bar{A}_L)} \right\|_2 + \left\| X_{p:k}^{(h)} \right\|_2 \right) . \quad (\text{B.19})$$

From (B.17) and (B.19), we can directly deduce the existence of $c > 0$ such that

$$S(p+M) - \langle x_p, \mathcal{R}x_p \rangle \leq c \|x_p\|_2^2 .$$

Note that the constant c does not depend on x_p . □

In the sequel, the matrix \mathcal{R} is assumed to be positive definite. Such an assumption is satisfied, for instance by the linear system presented in [89]. The following Lemma shows that by choosing x_p appropriately in a neighbourhood of the origin, the spectrogram constraint at time $p+M$ is satisfied.

Lemma B.2. *Let $p \geq N$. For all $\delta \in (0, \alpha)$,*

$$x_p \in \mathcal{E}(\mathcal{R}, \alpha - \delta) \cap \mathcal{B} \left(0, \min \left\{ \sqrt{\frac{\delta}{c}}, r \right\} \right) \implies S(p+M) \leq \alpha ,$$

where r is defined via (B.3).

Proof. Let $\delta \in]0, \alpha[$ and $x_p \in \mathcal{E}(\mathcal{R}, \alpha - \delta) \cap \mathcal{B} \left(0, \min \left\{ \sqrt{\frac{\delta}{c}}, r \right\} \right)$. Hence

$$\begin{aligned} S(p+M) &\leq \langle x_p, \mathcal{R}x_p \rangle + c \|x_p\|_2^2 \leq \alpha - \delta + c \left(\sqrt{\frac{\delta}{c}} \right)^2 \\ &\leq \alpha . \end{aligned}$$

□

In the remainder, we fix $\delta \in]0, \alpha[$. For less conservatism, δ should be chosen as close as possible to α . First, a set, which is invariant under the linearised closed-loop dynamics $x_{k+1} = \bar{A}_L x_k$, is computed, guaranteeing satisfaction of the spectrogram constraint at time $p + M$, by enforcing containment in the neighbourhood of the origin defined in Lemma B.2.

Lemma B.3. *There exists a matrix $\mathcal{S} \succ 0$ such that $\mathcal{E}(\mathcal{S}, 1)$ is invariant under the linearised dynamics $x_{k+1}^{(L)} = \bar{A}_L x_k^{(L)}$ and*

$$\mathcal{E}(\mathcal{S}, 1) \subseteq \mathcal{E}(\mathcal{R}, \alpha - \delta) \cap \mathcal{B}\left(0, \min\left\{\sqrt{\frac{\delta}{c}}, r\right\}\right) . \quad (\text{B.20})$$

Proof. The proof of existence is constructive. A matrix \mathcal{S} guaranteeing (B.20) can be computed by solving an SDP analogous to the one given in Theorem 3 in [89] with two additional constraints:

- the ‘spectrogram-ellipsoid’ is shrunk, resulting in the containment constraint

$$\mathcal{E}(\mathcal{S}, 1) \subseteq \mathcal{E}(\mathcal{R}, \alpha - \delta) ,$$

which can be formulated as an LMI in \mathcal{S}^{-1} .

- the containment

$$\mathcal{E}(\mathcal{S}, 1) \subseteq \mathcal{B}\left(0, \min\left\{\sqrt{\frac{\delta}{c}}, r\right\}\right) ,$$

which can also be expressed as an LMI in \mathcal{S}^{-1} .

□

The following Lemma guarantees invariance of a sub-level set of $\mathcal{E}(\mathcal{S}, 1)$ under the nonlinear dynamics. Its proof follows the arguments described in [114].

Lemma B.4. *There exists $\kappa \in]0, 1[$ such that $\mathcal{E}(\mathcal{S}, \kappa)$ is invariant under the nonlinear dynamics $x_{k+1} = f(x_k, Kx_k)$.*

Proof. Define

$$d(x) := \langle f(x, Kx), \mathcal{S}f(x, Kx) \rangle - \langle x, \mathcal{S}x \rangle - 2\langle x, \mathcal{S}\bar{A}_L g(x) \rangle - \langle g(x), \mathcal{S}g(x) \rangle .$$

From the invariance of $\mathcal{E}(\mathcal{S}, 1)$, it is clear that for all $x \in \mathcal{E}(\mathcal{S}, 1)$, $d(x) < 0$. Note that the function d is continuous and that $\mathcal{E}(\mathcal{S}, 1)$ is compact. Hence one can define

$$d_\infty := \max_{x \in \mathcal{E}(\mathcal{S}, 1)} d(x) < 0 . \quad (\text{B.21})$$

For all $x \in \mathcal{E}(\mathcal{S}, 1)$,

$$\langle f(x, Kx), \mathcal{S}f(x, Kx) \rangle \leq d_\infty + \langle g(x), \mathcal{S}g(x) \rangle + 2\langle g(x), \mathcal{S}\bar{A}_L x \rangle + \langle x, \mathcal{S}x \rangle .$$

From the definition of g , $\langle g(x), \mathcal{S}g(x) \rangle + 2\langle g(x), \mathcal{S}\bar{A}_L x \rangle \rightarrow 0$ when $x \rightarrow 0$. Then, there exists $\sigma > 0$ such that

$$\forall x \in \mathcal{B}(0, \sigma), \left| \langle g(x), \mathcal{S}g(x) \rangle + 2\langle g(x), \mathcal{S}\bar{A}_L x \rangle \right| \leq -d_\infty .$$

Let $\kappa > 0$ such that $\mathcal{E}(\mathcal{S}, \kappa) \subseteq \mathcal{E}(\mathcal{S}, 1) \cap \mathcal{B}(0, \sigma)$. Hence,

$$x \in \mathcal{E}(\mathcal{S}, \kappa) \implies \langle f(x, Kx), \mathcal{S}f(x, Kx) \rangle \leq \kappa ,$$

which proves that $\mathcal{E}(\mathcal{S}, \kappa)$ is invariant under the nonlinear dynamics (B.1). \square

Theorem B.3. *The spectrogram-MPC problem formulated onto the nonlinear system (B.1) with terminal constraint $x_N \in \mathcal{E}(\mathcal{S}, \kappa)$ is recursively feasible.*

Proof. The proof follows the same lines as in the linear case. As the set $\mathcal{E}(\mathcal{S}, \kappa)$ is invariant under the nonlinear dynamics, shifting the optimal sequence from the current step and appending the LQR solution $u = Kx$ provides a feasible solution to problem (B.8) at the next time instant. Satisfaction of the spectrogram constraint computed on the nonlinear dynamics at time $N + M$ is guaranteed by Lemmas B.1 and B.2 and the appropriate choice of the terminal constraint formulated in Lemmas B.3 and B.4. \square

Theorem B.4. *The closed-loop nonlinear system under the spectrogram-MPC control law is locally asymptotically stable, with basin of attraction equal to the feasible set of the spectrum constrained NMPC problem (B.8).*

Proof. Stability of the closed-loop system follows from recursive feasibility and the fact that the terminal cost satisfies the standard decrease Assumption (B.4). \square

B.5 Numerical Example

Oscillations are very common in mechanical systems and are responsible, e.g., for fatigue and failure of engines, and thus control techniques for active vibration damping are required [126]. In this section, an example illustrating the efficacy of the proposed spectrum constrained NMPC approach for damping resonance frequencies in constrained nonlinear systems is presented. In practice, many oscillatory dynamical systems can be modelled as linear resonators with a nonlinear restoring force [122]. Therefore, we consider the constrained nonlinear system

$$\left\{ \begin{array}{l} \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} 0 & x_2 \\ -\omega_0^2(x_1 + \epsilon x_1^2) & -2\nu\omega_0 x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 100 \end{pmatrix} u \\ z = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ |x_1| \leq 15, |x_2| \leq 100 \\ |u| \leq 100 \end{array} \right. \quad (\text{B.22})$$

where $\epsilon = 0.1$, $\omega_0 = 2\pi \cdot 12$ rad/sec and $\nu = 2 \cdot 10^{-4}$. System (B.22) is first controlled to track a piecewise constant reference signal $z_{\text{ref}} = \pm 0.5$ using a standard NMPC formulation without spectrogram constraints. The continuous dynamics are sampled at 50 Hz and discretised by applying a Runge-Kutta method of order four. The linearised model around the origin is given by (B.2) with

$$A_L = \begin{pmatrix} 0 & 1.00 \\ -5.68 \cdot 10^3 & 0.0030 \end{pmatrix}, \quad B_L = \begin{pmatrix} 0 \\ 100 \end{pmatrix}.$$

The stabilising control law used in the spectrum constrained NMPC problem is

$$K = \begin{pmatrix} -0.87 & -0.14 \end{pmatrix}.$$

The stage cost of (B.8) is defined as a quadratic function $l(x, u) := \langle x, Qx \rangle + \langle u, Ru \rangle$ with $Q = 100 \cdot I$ and $R = 1$.

When the system output tracks the upper constant reference $+0.5$, a resonance can be observed around 12.1 Hz, whereas when tracking the lower reference -0.5 , the resonance is obtained around 10.5 Hz, as shown in the time domain trajectory in Fig. B.1(a) and the spectrogram in Fig. B.2(a). Spectrogram constraints are then incorporated into the NMPC problem. A 3rd order Butterworth filter has been chosen with a window length $M = 25$, the prediction horizon being $N = 30$. The spectrogram constraint parameter α is set to 0.1. A constraint is first enforced at 10.5 Hz, which re-

sults in the spectrogram in Fig. B.2(b). The constraint on the first resonance is then removed and a constraint at 12.1 Hz is added, resulting in the spectrogram in Fig. B.2(c). Finally, both resonances are constrained, as shown in the spectrogram of Fig. B.2(d). The corresponding closed-loop trajectories are shown in Fig. B.1(a), (b), (c) and (d) respectively. The spectrum constrained NMPC strategy proves effective at damping nonlinear resonances. It should be noted that a waterbed effect can be observed in spectrograms (b) and (d), where damping the first resonance seems to amplify the second one, and damping both resonances results in some energy transfer to lower and higher frequencies.

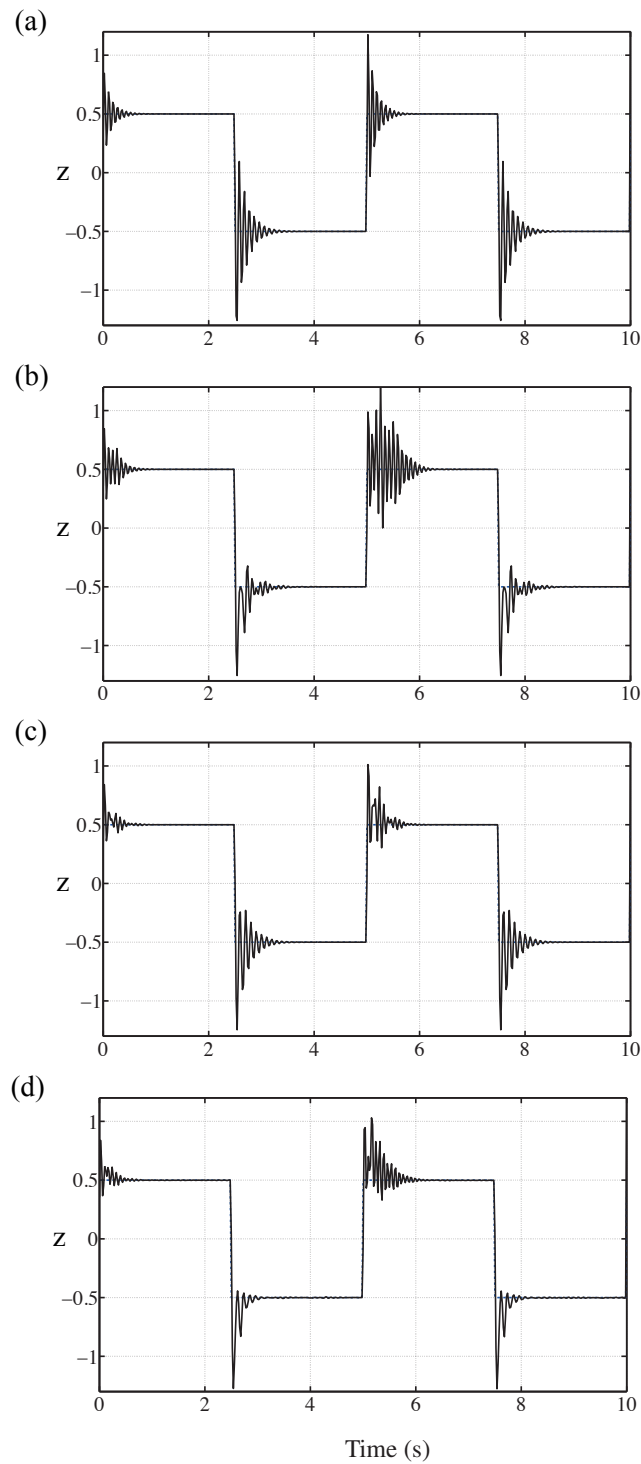


Figure B.1: Closed-loop output trajectories: Without spectrum constraint (a), with spectrum constraints at 10.5 Hz (b), with spectrum constraints at 12.1 Hz (c), and with spectrum constraints at both frequencies (d).

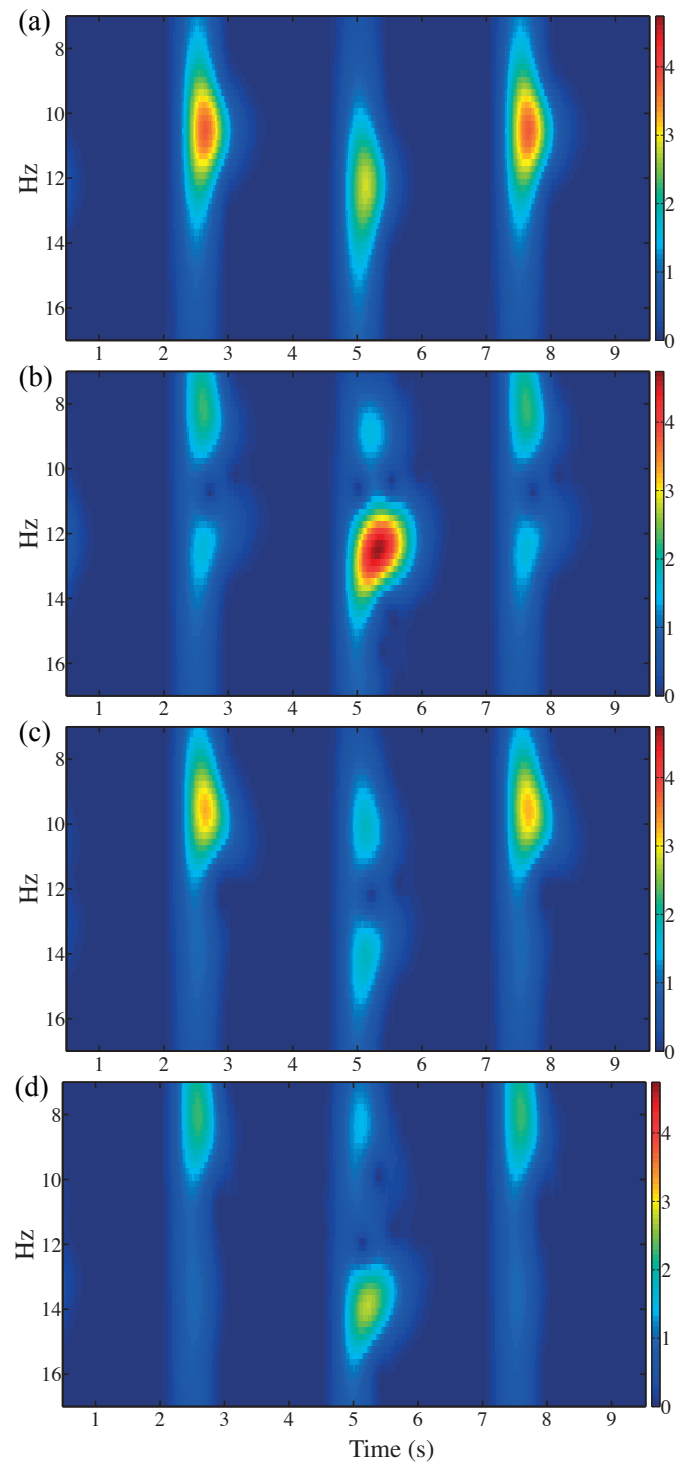


Figure B.2: Spectrograms of the output signal of the closed-loop system: No spectrum constraint (a), spectrum constraint at 10.5 Hz (b), at 12.1 Hz (c), and at both resonances (d).

Appendix C

Parametric Polytope Reconstruction, an Application to Crystal Shape Estimation

C.1 Introduction

Monitoring the CSD (Crystal Size Distribution) during a crystallisation process is of critical importance for the quality of the end product, the chemical properties of the crystal, and the efficiency of the manufacturing process. In order to estimate crystal shapes in-situ, several techniques exist such as laser backscattering, yet these techniques rely on the assumption that the particles are spherical and can thus not be applied in the case of highly non-spherical particles such as needles for instance. Recently, the interest in in-situ imaging-based methods for crystal shape estimation has increased [106, 143, 125]. Such techniques allow one to record the sizes and shapes of the crystals rapidly. Yet obtaining quantitative information about the crystal shape generally requires image segmentation.

In order to estimate the shape of a three-dimensional object from multiple views when the object pose is unknown, model-based methods have been successfully applied [5]. Such approaches rely on fitting a shape prior with images obtained from multiple views by minimising a re-projection error, providing an estimate of the object pose and shape. A few model-based approaches exist for crystal shape estimation from images, a survey is given in Section C.2 of this Chapter. Yet they all have drawbacks in terms of accuracy or in-situ applicability. In this Chapter, we propose a novel method to estimate crystal shapes from two orthogonal microscope views. The salient ingredient is a modelling of crystals as convex parametric polytopes. Moreover, the ‘weak-perspective’ assumption [4] allows one to take images of crystals as projections of the parametric polytope. Crystal shape estimation can thus be formalised as a polytope reconstruction problem.

Reconstructing a polytope from its projections onto hyperplanes is a long-standing problem that appears in different forms. More generally, the polytope reconstruction problem is part of a

field of mathematics called geometric tomography, which deals with ‘the retrieval of information about a geometric object from data about its sections, or projections, or both’ [65]. Polytope reconstruction problems appear in medical imaging [136], computer-aided design [156], computer vision [144] and computational geometry [18, 66, 112, 72]. The reconstruction technique always depends on the set-up and the type of projection data provided. From a theoretical point of view, in [112], the authors addressed the problem of reconstructing a 3-polytope¹ given one of its projections and two associated triangulations, and derive conditions under which such a 3-polytope exists. With a limited amount of information in the projections data, namely the number of visible edges, [18] derived some conditions under which a polytope can produce the given set of projections. From a more applied point of view, a field in which 3D reconstruction appears quite often is computer tomography, in which one seeks to recover the shape of an object from X-ray images. An algorithm for reconstructing any convex body in \mathbb{R}^n from its brightness function, the function giving the volume of its projections onto hyperplanes has been provided by [67].

In some cases, the polytope reconstruction problem is also closely related to the estimation of the polytope spatial orientation. Regarding this question, a few studies exist. In special cases, assuming that the correspondence between the vertices of the 3-polytope and the vertices of the 2-polytope is known, it is possible to estimate the rotation of the 3-polytope by applying standard results in projective geometry [136]. The case where the correspondence is unknown is more involved and conditions under which the computation of the rotation is possible are explored in [66] based on Gröbner bases. The method proposed in this Chapter avoids computing correspondence points between the crystal model and data projections, which makes it quite promising for a real-time application.

In the first part of this Chapter, a brief survey of existing approaches to crystal shape estimation from images is provided and our vision set-up is presented. Then the proposed technique is presented and it is shown that the shape estimation problem can be recast as nonlinear least-squares. Technical details regarding the parameterisation of the vertices of the parametric polytope are also exposed. Finally, the effectiveness of our approach is demonstrated on artificially generated images as well as real images produced by a real-world vision set-up.

C.2 Vision methods for crystal shape estimation

C.2.1 State of the art

Existing vision methods for estimating three-dimensional crystal shapes range from complex techniques such as tomography or laser backscattering to more basic ones such as in-situ video mi-

¹An n -polytope is a polytope in \mathbb{R}^n .

croscopy. A survey of existing techniques is provided by [106]. Details about imaging instruments used for the monitoring of crystallisation processes are given in [143]. Recently there has been a lot of interest in image-based methods, which allow one to easily visualise the crystal shape and acquire data quickly, but require a significant amount of processing in order to extract quantitative information. Several algorithms for crystal shape estimation from in-situ images have been proposed [143, 107]. Both [143] and [107] are model-based approaches that estimate a shape parameter from image data and a prior model of the crystal. Such approaches have been commonly applied in computer vision to measure shapes of complex objects using stereoscopic imaging [5]. Stereo-imaging techniques have been applied to estimate simple crystal shapes such as spheres or cubes, yet no systematic method to estimate complex shapes exist yet.

In [143], the three-dimensional crystal morphology is estimated from tomographic images of crystals obtained via confocal microscopy. Such a tomographic approach provides very accurate results, yet crystals need to be fluorescent coated, which makes the approach difficult to apply in-situ. The M-SHARC (Model-Based Shape Recognition for Crystals) algorithm proposed by [107] extracts shape information from a single image and a wire-frame model consisting of a set of vertices and a set of lines. The salient ingredient of the M-SHARC algorithm is linear feature detection and matching. One of its main advantages is speed (10 images per minute), which makes it applicable in a real-time in-situ context. More recently, a stereological method has been proposed by [124] in order to estimate the shape of any 3D convex body from several 2D projections. It has been successfully tested in a crystallisation process. The essential ingredient of the method is a maximum likelihood estimator based on an appropriate shape descriptor.

C.2.2 Proposed approach

The main challenge of in-situ microscopic imaging techniques is to infer three-dimensional information from two-dimensional data in an efficient way. Stereoscopic imaging techniques are generally applied to solve such 3D reconstruction problems. In this Chapter, the stereoscopic imaging set-up consists of two cameras photographing suspended particles which are pumped through a glass-walled cell that avoids optical distortion effects from two perpendicular directions, as shown in Figure C.1. The magnification is such that 1 pixel corresponds to $1.15\mu\text{m}$. Images are captured at a rate of 5 Hz and have a resolution of 5 MP. Xenon flash lamps with a very short decay time assure that no motion blur, which might occur due to the movement of crystals, is visible on the recorded images. As the size of the crystals is small compared to the length of the cameras to the flow-through cell, the ‘weak-perspective’ projection model [4] is chosen for the image formation, so that the obtained images can be taken as scaled projections of the crystal.

Similarly to [143] and [107], the approach described in this Chapter is a model-based proce-

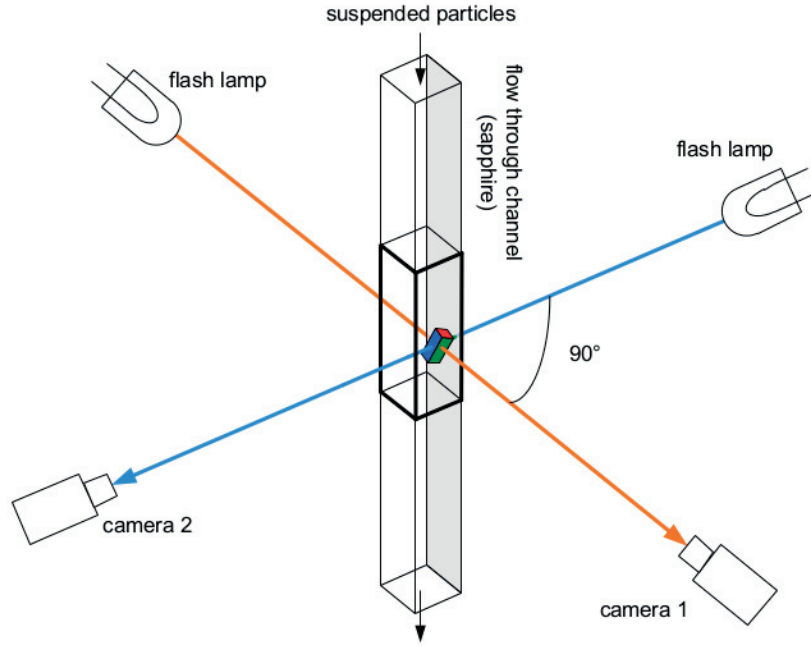


Figure C.1: Schematic drawing of the flow through cell. The light from the xenon flash lamps passes through the cell. Orthogonal projections of crystals in suspension are captured by the two cameras. A pre-processing is applied to each pair of images so as to extract pairs of single crystals. Hence crystals are analysed one by one.

dure. A faceted crystal is modelled by a parametric polytope as defined in Section C.4. Particle models are based on crystallographic data, i.e. crystal unit cell parameters, and a set of experimentally observed facets that are to be included. Crystal facets are commonly identified by their Miller indices [22]. Using the geometry of the unit cell, a normal vector in Cartesian space, which defines a facet plane, can be calculated for each Miller index. These facet vectors make the *parameter matrix* A as used in the parametric polytope definition (Eq. (C.1)). As not all crystal facets are considered to grow independently, a matrix B is defined which relates the growth of some faces and reduces the degrees of freedom in the model. Finally, the relative distance of each group of facets to the origin of the polytope is defined by the *shape parameter* t as introduced in Eq. (C.1). Thus estimating the three-dimensional crystal shape amounts to computing the shape parameter t from data points extracted from the set of two images.

As often in stereoscopic imaging techniques, evaluating the orientation of the crystal is an important issue. Contrary to the M-SHARC algorithm, which requires certain assumptions on the crystal orientation, the technique proposed in this Chapter allows one to automatically estimate the

crystal orientation and shape from the pair of images.

C.3 Notation

We denote by $\text{conv}E$ the convex hull of a set E . We denote by $d_H(E, F)$ the Hausdorff distance between two sets E and F . $\mathbb{1}_n$ is the vector in \mathbb{R}^n with all coordinates equal to 1. Similarly $\mathbb{0}_n$ stands for the vector in \mathbb{R}^n with all coordinates equal to 0. I_n is the identity matrix in \mathbb{R}^n .

C.4 Basic definitions in polyhedral geometry

In this section, we present some basic definitions in polyhedral geometry, which can be found in [163].

Definition C.1 (Polyhedron). *A polyhedron P in \mathbb{R}^d is the intersection of finitely many closed half-spaces in \mathbb{R}^d .*

$$P := \{x \in \mathbb{R}^d : Ax \leq b\} \quad ,$$

where $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$ and the inequality \leq is row-wise. The rows of A are denoted by a_i^\top , $i \in \{1, \dots, m\}$.

In the sequel, we denote the polyhedral set $\{x \in \mathbb{R}^d : Ax \leq b\}$ by $P(A, b)$.

Definition C.2 (Polytope). *A polytope is a bounded polyhedron. A polytope in \mathbb{R}^d is called a d -polytope.*

Definition C.3 (Parametric polytope). *A parametric polytope $P(A, Bt)$, where $t \in \mathbb{R}^p$, is the polyhedral set*

$$P(A, Bt) := \{x \in \mathbb{R}^d : Ax \leq Bt\} \quad . \tag{C.1}$$

Assumption C.1. *The shape parameter t lies in a p -polytope T .*

Remark C.1. *Assumption C.1 ensures that the shape parameter is bounded. In practice, this models the fact that crystals can be measured up to a fixed maximal size.*

Remark C.2. *In the sequel, \mathbb{R}^d is sometimes referred to as the data space and \mathbb{R}^p as the parameter space.*

Remark C.3 (Rotated parametric polytope). *The parametric polytope $P(A, Bt)$ can be rotated by $R \in SO_d(\mathbb{R})$, resulting in the rotated parametric polytope*

$$P(AR^\top, Bt) := \{x \in \mathbb{R}^d : AR^\top x \leq Bt\} .$$

Definition C.4 (Projection of a polytope onto a hyperplane). *The projection of a polytope $P \subset \mathbb{R}^n$ onto a hyperplane $\mathcal{H} = \{x \in \mathbb{R}^n : a^\top x = b\}$ is denoted by $\pi_{\mathcal{H}}P$ and is defined as*

$$\pi_{\mathcal{H}}P := \{x \in \mathcal{H} : \exists \lambda \in \mathbb{R}, x + \lambda a \in P\} .$$

Similarly, the projection from \mathbb{R}^{d+p} onto \mathbb{R}^d is defined by the function $\pi_{\mathbb{R}^d}$, and onto \mathbb{R}^p by $\pi_{\mathbb{R}^p}$.

Remark C.4. *The $n \times n$ matrix defining the projection onto \mathcal{H} is denoted by $P_{\mathcal{H}}$.*

Definition C.5 (Affine hull). *Let $\mathcal{S} := \{x_i\}_{i=0}^{N-1}$ be a set of points in \mathbb{R}^d . The affine hull of \mathcal{S} , denoted by $\text{aff } \mathcal{S}$, is the smallest affine set that contains \mathcal{S} . It can be shown that $\text{aff } \mathcal{S}$ is the set of all affine combinations of elements of \mathcal{S}*

$$\text{aff } \mathcal{S} := \left\{ \sum_{i=0}^{N-1} \lambda_i x_i : \sum_{i=0}^{N-1} \lambda_i = 1, x_i \in \mathcal{S}, i \in \{0, \dots, N-1\} \right\} .$$

Definition C.6 (Polyhedral partition). *Let P be a p -polytope. A finite family of p -polytopes*

$$\{P_1, \dots, P_r\}$$

is a polyhedral partition of P if

$$\begin{cases} P = \cup_{i=1}^r P_i \\ \forall i \neq j, \text{int } P_i \cap \text{int } P_j = \emptyset \end{cases} ,$$

where $\text{int } P$ stands for the interior of P . In the remainder of the Chapter, when referring to a polyhedral partition, we use the symbol \sqcup instead of \cup to show that the union is disjoint.

C.5 Description of the shape parameter estimation technique

In this section, we address the general problem of estimating the shape parameter \tilde{t} of a rotated parametric d -polytope $P(A\tilde{R}^\top, B\tilde{t})$ from N of its projections onto hyperplanes

$$\{\mathcal{H}_1, \dots, \mathcal{H}_N\}$$

in \mathbb{R}^d . As it appears later in Section C.8, the hyperplanes $\{\mathcal{H}_1, \dots, \mathcal{H}_N\}$ model the N different views of a crystal, represented as a parametric polytope. Following Definition C.4, the projections of $P(A\tilde{R}^\top, B\tilde{t})$ onto $\mathcal{H}_1, \dots, \mathcal{H}_{N-1}$ and \mathcal{H}_N are denoted by

$$\left\{ \pi_{\mathcal{H}_1} P(A\tilde{R}^\top, B\tilde{t}), \dots, \pi_{\mathcal{H}_N} P(A\tilde{R}^\top, B\tilde{t}) \right\} .$$

A quick look at Fig. C.2 shows that estimating the shape parameter \tilde{t} goes together with estimat-

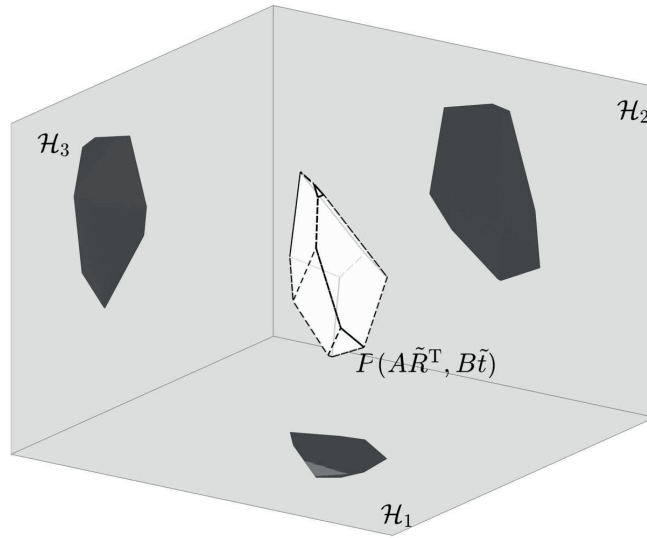


Figure C.2: Illustration of the parametric polytope reconstruction problem: Estimate the shape parameter \tilde{t} from projections onto hyperplanes \mathcal{H}_1 , \mathcal{H}_2 and \mathcal{H}_3 . The polytope projections $\pi_{\mathcal{H}_1} P(A\tilde{R}^\top, B\tilde{t})$, $\pi_{\mathcal{H}_2} P(A\tilde{R}^\top, B\tilde{t})$ and $\pi_{\mathcal{H}_3} P(A\tilde{R}^\top, B\tilde{t})$ are plotted as dark areas.

ing the orientation \tilde{R} of $P(A\tilde{R}^\top, B\tilde{t})$. The main concept of our shape estimation algorithm is to calculate the pair of parameters (\hat{t}, \hat{R}) , which result in the best fitting of the projections of the model polytope

$$\left\{ \pi_{\mathcal{H}_1} P(A\hat{R}^\top, B\hat{t}), \dots, \pi_{\mathcal{H}_N} P(A\hat{R}^\top, B\hat{t}) \right\}$$

with the data polytopes

$$\left\{ \pi_{\mathcal{H}_1} P(A\tilde{R}^\top, B\tilde{t}), \dots, \pi_{\mathcal{H}_N} P(A\tilde{R}^\top, B\tilde{t}) \right\} . \quad (\text{C.2})$$

In the sequel, the data polytopes $\pi_{\mathcal{H}_i} P \left(A\tilde{R}^\top, B\tilde{t} \right)$ are denoted by D_i , for $i \in \{1, \dots, N\}$.

Remark C.5. *The data polytopes $\{D_i\}_{i=1}^N$ are extracted from a single set of N different projections of the same parametric polytope, or N different 2D images of the same crystal, as explained in Section C.8.*

Such an approach is very frequent in Computer Vision for estimating the shape parameter of a three-dimensional object from multiple-view images. The key idea to minimise the re-projection error between a parametric model of the object and data points extracted from the images. Hence a metric measuring the discrepancy between the projected model polytope and the data polytopes should be defined in order to obtain an accurate estimate \hat{t} of the shape parameter \tilde{t} in the l^2 -sense, that is minimising the error $\|\tilde{t} - \hat{t}\|_2$. In the remainder, it is shown how an appropriate metric can be constructed in order to measure and then minimise the discrepancy between the parametric polytopes

$$\left\{ \pi_{\mathcal{H}_1} P \left(A\hat{R}^\top, B\hat{t} \right), \dots, \pi_{\mathcal{H}_N} P \left(A\hat{R}^\top, B\hat{t} \right) \right\}$$

and the data polytopes

$$\left\{ \pi_{\mathcal{H}_1} P \left(A\tilde{R}^\top, B\tilde{t} \right), \dots, \pi_{\mathcal{H}_N} P \left(A\tilde{R}^\top, B\tilde{t} \right) \right\} .$$

Remark C.6. *Note that the data polytopes $\{D_i\}_{i=1}^N$ are in \mathbb{R}^{d-1} , whereas the parametric polytope lies \mathbb{R}^d . Thus the polytopic shape estimation problem consists in inferring complex geometric information on a d -dimensional polytopic object from $(d - 1)$ -dimensional data, with very few assumptions on the problem structure.*

C.5.1 Choice of the re-projection error

We propose defining the re-projection error as the average distance of the projected vertices of the parametric polytope to the data polytopes plus the average distance of the vertices of the data polytopes to the projected parametric model polytope. It is shown later that this comes from the definition of the re-projection error as an averaged version of the Hausdorff distance between the projected parametric model polytope and data polytopes. Thus the re-projection error should be interpreted as an approximation of the distance between a model set and data sets.

The sets of vertices of the parametric model $P(A, Bt)$ and the data D_i for $i \in \{1, \dots, N\}$, are defined as

$$\text{extr } P(A, Bt) := \{v_1(t), \dots, v_{N_v(t)}(t)\}$$

and

$$\text{extr } D_i := \left\{ w_1^{(i)}, \dots, w_{M_i}^{(i)} \right\} .$$

The proposed re-projection error is then given by

$$\begin{aligned} \Delta \left(P (AR^\top, Bt), \{D_i\}_{i=1}^N \right) &:= \frac{1}{N} \sum_{i=1}^N \left(\frac{\beta_i}{N_v(t)} \sum_{j=1}^{N_v(t)} d(\pi_{\mathcal{H}_i} R v_j(t), D_i)^2 \right. \\ &\quad \left. + \frac{\gamma_i}{M_i} \sum_{k=1}^{M_i} d \left(w_k^{(i)}, \pi_{\mathcal{H}_i} P (AR^\top, Bt) \right)^2 \right) , \end{aligned} \quad (\text{C.3})$$

where N is the number of hyperplanes, the coefficients $\beta_i > 0$ and $\gamma_i > 0$ such that $\beta_i + \gamma_i = 1$ are relative weighting coefficients.

Remark C.7. *The relative weighting coefficients β_i and γ_i may help tuning the re-projection error in some practical cases.*

Finally, obtaining an estimate of \tilde{R} and \tilde{t} consists in minimising the re-projection error

$$\Delta \left(P (AR^\top, Bt), \{D_i\}_{i=1}^N \right) ,$$

which results in the nonlinear program

$$\text{minimise}_{t,R} \Delta \left(P (AR^\top, Bt), \{D_i\}_{i=1}^N \right) \quad (\text{C.4a})$$

Constraints on the shape parameter

$$t \in T \quad (\text{C.4b})$$

Rotation matrix

$$R \in SO_d(\mathbb{R}) . \quad (\text{C.4c})$$

where T is defined in Assumption C.1. At this point, two challenges appear. First, the vertices $v_j(t)$ of the parametric polytope $P(A, Bt)$ should be expressed as a function of the shape parameter t . It is shown in Section C.6 that the vertices $v_j(t)$ are *piecewise affine* (PWA) functions of the shape parameter t defined over a polyhedral partition $\sqcup_{i=1}^L T_i$ of the parameter polytope T . Secondly, by using an appropriate parameterisation of the rotation matrix R , the nonlinear constraint $R \in SO_d(\mathbb{R})$ can be transformed into box constraints on a parameter α , as addressed in Sec-

tion C.7. Once these two problems have been resolved, the nonlinear program can be recast as a constrained nonlinear least-squares problem.

Remark C.8 (Re-projection error as a Hausdorff distance). *The re-projection error appearing in the objective (C.4) can be viewed as a sum of pseudo-Hausdorff distances between the projections of the model polytope and data polytopes, via the following Lemma, of which the proof can be derived easily.*

Lemma C.1. *Let $P_1 = \text{conv}\{v_1, \dots, v_n\}$ and $P_2 = \text{conv}\{w_1, \dots, w_m\}$.*

$$d_H(P_1, P_2) = \max \left\{ \max_{i=1, \dots, n} d(v_i, P_2), \max_{j=1, \dots, m} d(w_j, P_1) \right\}, \quad (\text{C.5})$$

where $d(v, P) = \min_{x \in P} d(v, x)$.

From (C.5), the objective of (C.4) is obtained by replacing the max operator with the L^2 -norm. Lemma C.1 essentially means that the distance between two polytopic sets can be expressed as a function of the distances of the vertices of one polytope to the other polytope and vice-versa. Therefore, parametric vertices $v_i(t)$ and data vertices $w_k^{(i)}$ appear in the expression of the re-projection error, and constraints guaranteeing containment in the model or data polytopes need to appear in the minimisation of the re-projection error. The Hausdorff metric has been successfully employed for comparing a model set and an image set in model matching algorithms such as [94].

C.5.2 A nonlinear least-squares problem for parametric polytope shape estimation

After deriving a parameterisation of the vertices of the parametric polytope $P(A, Bt)$, as explained in Section C.6 and of the rotation R , as in Section C.7, the nonlinear program (C.4) can be transformed into a finite set of constrained nonlinear least-squares problems. More precisely, an estimate of the shape parameter is obtained by solving

$$J^* = \text{minimise}_{l \in \{1, \dots, L\}} J_l, \quad ,$$

where L is the number of polytopes in the polyhedral partition of T and J_l is defined over each partition polytope T_l as

$$J_l := \min_{\substack{t, \alpha, \\ \{y_i^{(1)}\}, \dots, \{y_i^{(N)}\} \\ \{z_j^{(1)}\}, \dots, \{z_j^{(N)}\}}} \frac{1}{N} \sum_{k=1}^N \left(\frac{\beta_k}{N^l} \sum_{i=0}^{N_v^l - 1} \left\| P_{\mathcal{H}_k} R(\alpha) v_i(t) - y_i^{(k)} \right\|_2^2 + \frac{\gamma_k}{M_k} \sum_{j=0}^{M_k - 1} \left\| w_j^{(k)} - P_{\mathcal{H}_k} R(\alpha) z_j^{(k)} \right\|_2^2 \right) \quad (\text{C.6a})$$

Containment in $(d - 1)$ -dimensional data polytopes:

$$\begin{aligned} y_i^{(1)} &\in D_1 \\ &\dots \\ y_i^{(N)} &\in D_N, \quad i \in \{0, \dots, N_v - 1\} \end{aligned} \tag{C.6b}$$

Containment in parametric polytope $P(t)$:

$$\begin{aligned} Az_j^{(1)} &\leq Bt, \quad j \in \{0, \dots, M_1 - 1\} \\ &\dots \\ Az_j^{(N)} &\leq Bt, \quad j \in \{0, \dots, M_N - 1\} \end{aligned} \tag{C.6c}$$

Polyhedral constraint on shape parameter:

$$t \in T_l \tag{C.6d}$$

Box constraints on rotation parameter:

$$\alpha \in [\alpha_U, \alpha_L] \quad , \tag{C.6e}$$

where $\{P_{\mathcal{H}_k}\}_{k=1}^N$ are the matrices of the projections onto $\{\mathcal{H}_1, \dots, \mathcal{H}_N\}$ respectively.

The special orthogonal constraint $R^\top R = I$ vanishes, but a nonlinear expression of the rotation matrix $R(\alpha)$ appears in the objective of the nonlinear program. Another important change compared to (C.4) is that the number of vertices N_v^l of the parametric polytope $P(A, Bt)$ does not depend on t anymore. This results from the fact that the shape parameter t lies in a fixed polytope T_l of the polyhedral partition, as clarified in Section C.6.

One of the key aspects of the proposed shape estimation procedure is that the minimisation of the re-projection error does not involve correspondences between points of the parametric model polytope and points in the data polytopes, which is generally the case in most of the reconstruction techniques based on model matching [131, 5], and often leads to binary optimisation problems, which are notoriously hard to solve. This is a direct consequence of the choice of the re-projection error, which is basically taken as an approximated distance between a model set and data sets, and Lemma C.1.

C.6 A partition of the parameter polytope

In this section, we clarify the motivation for computing a polyhedral partition of the parameter polytope T and present basic results in polyhedral geometry, which are essential to the proposed partition generating algorithm. First, one can note that

$$\pi_{\mathcal{H}_i} P(AR^\top, Bt) = \text{conv} \{ \pi_{\mathcal{H}_i} Rv_k(t) \}_{k=1}^{N_v(t)},$$

where $\{v_k(t)\}_{k=1}^{N_v(t)}$ are the vertices of $P(A, Bt)$. As the shape parameter t varies in T , the vertices $\{v_k(t)\}_{k=1}^{N_v(t)}$ of $P(A, Bt)$ can split or merge. Thus the first step of our method is to identify regions of the parameter polytope T in which the set of parametric vertices does not change. More precisely, every parametric vertex $v_k(t)$ can be represented as a PWA function of the shape parameter t , that is

$$v_k(t) := M_k^{(l)} t, \text{ for } t \in T_l. \quad (\text{C.7})$$

We aim at identifying the regions T_l , appearing in (C.7) of the parameter polytope T , which we call *critical regions*, in which the vertices of $P(A, Bt)$ can be expressed via a fixed set of matrices $\{M_k^{(l)}\}$ in $\mathbb{R}^{d \times p}$.

It is shown that each critical region is a polyhedron in the parameter space \mathbb{R}^p . The main concept of the partition generating algorithm is to enumerate the faces of a cone defined in mixed data-parameter space, project them onto the parameter space and generate a partition of the parameter polytope from the set of projected faces intersected with the parameter polytope T . After introducing some basic facts about faces of polyhedra, the partition generating algorithm is presented and it is proven that its output is a partition of the parameter polytope T .

C.6.1 Preliminaries

Most of the following definitions can be found in [163] and [13].

Definition C.7 (Face). *A subset F in \mathbb{R}^d is called a p -face of a polyhedron $P(A, b)$ if there exists a supporting hyperplane \mathcal{H} of $P(A, b)$ such that*

$$\begin{cases} F = P(A, b) \cap \mathcal{H} \\ \dim \text{aff } F = p \end{cases}. \quad (\text{C.8})$$

0-faces are called vertices, 1-faces are called edges. In the case of a p -polytope, $(p - 1)$ -faces are called facets.

Definition C.8 (Sub-matrix). *Let $A \in \mathbb{R}^{m \times n}$ and $I \subseteq \{1, \dots, m\}$. The sub-matrix of A built by stacking the rows of A of indices contained in I is noted A_I .*

Given a subset $I \subseteq \{1, \dots, m\}$, we note

$$P_I := \{x \in P(A, b) : A_I x = b_I\} .$$

The notion of equality set introduced in [96] plays an important role in the computational aspects of our algorithm.

Definition C.9 (Equality set). *Let $P(A, b)$ be a d -polytope defined by the intersection of m hyperplanes. Let $E \subseteq \{1, \dots, m\}$ and*

$$G(E) := \{i \in \{1, \dots, m\} : \forall x \in P_E, \langle a_i, x \rangle = b_i\} .$$

E is an equality set of $P(A, b)$ if and only if $E = G(E)$.

The following Theorem, proven in [96], states that there is a one-to-one correspondence between faces of a polyhedron and its equality sets.

Theorem C.1 (Equality set to face correspondence). *If E is an equality set of a d -polytope $P(A, b)$, then P_E is a face of $P(A, b)$. Furthermore, if F is a face of $P(A, b)$, then there exists a unique equality set E such that $F = P_E$.*

A polyhedral cone C in data-parameter space can be defined from a parametric polytope $P(A, Bt)$, as follows:

$$C := \left\{ \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^{d+p} : \begin{bmatrix} A & -B \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq 0 \right\} .$$

In the remainder, the polyhedral cone C is assumed to be full-dimensional, that is $\dim \text{aff } C = d + p$.

C.6.2 Basic results

In this paragraph, basic results about faces of the cone C and vertices of the parametric polytope $P(A, Bt)$ are presented. The main idea is that each parametric vertex of $P(A, Bt)$ can be associated with a unique p -face of the cone C . This fact is clarified and proven in Lemma C.2, where it is shown that the set of parameters for which a parametric vertex exists is given by the projection of a p -face onto the parametric space. A polyhedral partition of the parameter polytope T can then be built from the set of faces of C associated with the set of vertices of $P(A, Bt)$.

Lemma C.2. *For all $t \in T$ and for all $v \in P(A, Bt)$, there exists a unique p -face F of C such that*

$$v = \pi_{\mathbb{R}^d}(F \cap S(t)) \quad ,$$

where

$$S(t) := \left\{ \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^{d+p} : y = t \right\} .$$

If F is a p -face of C such that $\dim \text{aff } F \cap S(0) = p$, then the set of parameters t such that $\pi_{\mathbb{R}^d}(F \cap S(t)) \in \text{extr } P(A, Bt)$ is $\pi_{\mathbb{R}^d}(F) \cap T$.

Proof. Let $t \in T$ and $v \in \text{extr } P(A, Bt)$. By definition of a vertex as a 0-face of $P(A, Bt)$, there exists a supporting hyperplane H_t of $P(A, Bt)$ such that $v = P(A, Bt) \cap H_t$. This implies that we can find a supporting hyperplane H of C such that

$$\begin{aligned} v &= \pi_{\mathbb{R}^d}(C \cap S(t)) \cap \pi_{\mathbb{R}^d}(H \cap S(t)) \\ &= \pi_{\mathbb{R}^d}(C \cap H \cap S(t)) \quad . \end{aligned} \tag{C.9}$$

As $\dim \text{aff } v = 0$ and v is obtained as a projection from \mathbb{R}^{d+p} onto \mathbb{R}^d , where p independent components are removed, $\dim \text{aff } C \cap H \leq p$. From the last equality in (C.9), one can deduce that

$$\left\{ \begin{pmatrix} x \\ t \end{pmatrix} \in \mathbb{R}^{p+1} : t \in T, x = v \right\} = C \cap H \cap \bigcup_{t \in \mathbb{R}^p} S(t) \quad .$$

As

$$\dim \text{aff} \left\{ \begin{pmatrix} v \\ t \end{pmatrix} \in \mathbb{R}^{p+1} : t \in T, x = v \right\} = p \quad ,$$

it follows that $\dim \text{aff } C \cap H \geq p$. Finally $\dim \text{aff } C \cap H = p$. As H is a supporting hyperplane of C , $C \cap H$ is a p -face of C . Assume that there exists two different p -faces of C , F_1 and F_2 such that

$$v = \pi_{\mathbb{R}^d}(F_1 \cap S(t)) = \pi_{\mathbb{R}^d}(F_2 \cap S(t)) \quad . \tag{C.10}$$

Thus $\pi_{\mathbb{R}^d}((F_1 \cup F_2) \cap S(t)) = v$. As previously observed, this implies that $\dim \text{aff } F_1 \cap F_2 \leq p$. Yet, $\dim \text{aff } F_1 \cap F_2 > p$, which is a contradiction. In conclusion, $F_1 = F_2$, meaning that the p -face associated with v is unique.

We then prove the second statement of the Lemma. Let F be a p -face of C such that $\dim \text{aff } F \cap S(0) = p$. So, if $\pi_{\mathbb{R}^d}(F \cap S(t))$ is non-empty, then it is a vertex of $P(A, Bt)$.

$$\begin{aligned} \{t \in T : F \cap S(t) \neq \emptyset\} &= \left\{ t \in T : \exists x \in \mathbb{R}^d, \begin{pmatrix} x \\ t \end{pmatrix} \in F \right\} \\ &= \pi_{\mathbb{R}^p}(F) \cap T . \end{aligned} \tag{C.11}$$

□

In conclusion, a unique p -face of C is associated to each vertex of the parametric polytope $P(A, Bt)$. We are only interested in p -faces such that the dimension of their intersection with $S(0)$ is p , so that they correspond to vertices of the parametric polytope $P(A, Bt)$. The number of such p -faces of C is denoted by n_F and their set is

$$\{F_1, \dots, F_{n_F}\} .$$

The vertex of $P(A, Bt)$ associated with the j -th p -face of C is denoted by v_{F_j} .

Definition C.10. Let $J \subseteq \{1, \dots, n_F\}$.

$$T_J := \left\{ t \in T : \text{extr } P(A, Bt) = \{v_{F_j}\}_{j \in J} \right\} .$$

Theorem C.2 (Polyhedral partition of T). $\{T_J\}_{J \in 2^{\{1, \dots, n_F\}}}$ is a polyhedral partition of the parameter polytope T .

Proof. Let $J \in 2^{\{1, \dots, n_F\}}$.

$$T_J = T \cap \left(\bigcap_{j \in J} \pi_{\mathbb{R}^p}(F_j) \right) .$$

As the projection of a polyhedron is a polyhedron and an intersection of polyhedra is a polyhedron, the set T_J is a polyhedron. Let $J \in 2^{\{1, \dots, n_F\}}$ and $K \in 2^{\{1, \dots, n_F\}}$ such that $J \neq K$. Assume for the sake of contradiction that $\text{int } T_J \cap \text{int } T_K \neq \emptyset$ and take $t \in \text{int } T_J \cap \text{int } T_K$. It follows that

$$\begin{aligned} \text{extr } P(A, Bt) &= \{\pi_{\mathbb{R}^d}(F_j \cap S(t))\}_{j \in J} \\ &= \{\pi_{\mathbb{R}^d}(F_k \cap S(t))\}_{k \in K} , \end{aligned}$$

which leads to a contradiction, since $\pi_{\mathbb{R}^d}(F_j \cap S(t)) = \pi_{\mathbb{R}^d}(F_k \cap S(t))$ implies $F_j = F_k$, by Lemma C.2, which states that the pre-image of each vertex is a unique p -face of the cone C . Sub-

sequently, the sets T_J are all disjoint. It remains to show that every $t \in T$ belongs to a polytope T_J . This follows again from Lemma C.2. \square

Once a polyhedral partition $T = \sqcup_{l=1}^L T_l$ has been computed, a parametric representation of the vertices of $P(A, Bt)$ can be derived. More precisely, the PWA function (C.7) defining the vertices of $P(A, Bt)$ can be explicitly computed. For a partition polytope T_l , the set of parameterisation matrices $M_k^{(l)}$ does not change, since the vertices are obtained by projecting the same set of faces of C . More precisely, the matrices $M_k^{(l)}$ are derived by calculating the Chebychev center $t_C^{(l)}$ of the polytope T_l [24], and extracting d active constraints for each vertex of $P(A, Bt_C^{(l)})$, which results in a matrix $M_k^{(l)} \in \mathbb{R}^{d \times p}$ after stacking the active constraints.

C.6.3 Computational geometry aspects

It has been shown that the parameter polytope T can be partitioned into a family of polytopes corresponding to a fixed parameterisation of the vertices of the parametric polytope $P(A, Bt)$. The salient ingredient for computing such a partition is the set of p -faces of the cone C in data-parameter space such that the dimension of their intersection with $S(0)$ is p . Subsequently, Algorithm 13 is made of two main steps, enumerate all p -faces F of C such that $\dim \text{aff } F \cap S(0) = p$ and check whether T_J is empty for $J \in 2^{\{1, \dots, n_F\}}$. The efficiency of each step can be improved further, as detailed below.

Algorithm 13 Partition generating algorithm

Input:

- Matrices A and B ,
- Parameter polytope T in half-space representation.

Enumerate all p -faces of the polyhedral cone C such that $\dim \text{aff } F \cap S(0) = p$.
 Check emptiness of polyhedra T_J for all $J \in 2^{\{1, \dots, n_F\}}$.

Output: Polyhedral partition $T = \sqcup_{l=1}^L T_l$.

C.6.3.1 Enumerating p -faces of C

In order to compute all p -faces of the polyhedral cone C , all faces of C are first listed by applying the algorithm proposed in [63]. In this framework, faces are represented by their associated equality sets (Theorem C.1), which corresponds to the maximum set of inequalities that are active at all points in the face, as formalised by Definition C.9. The algorithm then consists in applying a back-track search over the set of faces of C . The underlying idea is to partition the set of faces into m disjoint sets of faces, where m is the number of inequalities of the polyhedral cone C . Each of these m

sets is then built recursively. Once all faces of C have been enumerated, the p -faces can be extracted by checking all equality sets I in $\{1, \dots, m\}$ such that $\text{rank} \begin{bmatrix} A_I & B_I \end{bmatrix} = (d + p) - p = d$. Furthermore, the p -faces F satisfying $\dim \text{aff } F \cap S(0) = p$ can be extracted by checking whether $\text{rank } A_I = d$. The appropriate p -faces are then projected on the parameter space \mathbb{R}^p by means of a polytope projection algorithm such as the Equality Set Projection [96], and the resulting polytope is intersected with the data polytope T , which is a trivial operation, since all polytopes are in half-space representation.

C.6.3.2 Check emptiness of polyhedra T_I

This step can be performed efficiently by checking whether the diameter of the inscribed ball is below a pre-specified tolerance. This amounts to solving a linear program, which can be done efficiently by means of interior point methods. In practice, we use the MPT 3.0 function `isEmptySet` [80].

C.7 Rotation parameterisation

In this section, we address the parameterisation of the rotation R of the parametric polytope $P(A, Bt)$ in \mathbb{R}^d . In the context of crystal shape estimation, the data space has dimension $d = 3$. The problem very often appears in vision and robotics of finding an ‘optimal’ rotation, for instance the pose of the camera accounting for observed image points, which is very similar to our problem. In order to apply an optimisation procedure and obtain an estimate of the optimal rotation, it is relevant to use a parameterisation of the group of three-dimensional rotations $SO_3(\mathbb{R})$. Yet not all parameterisations are apt and several requirements should be met. According to [82] regarding estimation problems in vision, one of the key requirements for a rotation parameterisation is *fairness*, which basically means that the parameterisation should not bias the sensitivity results. This property is guaranteed if a rigid transformation of the space results in an orthogonal transformation of the space of parameters. Three rotation representations usually prevail: Euler angles, angle-axis and quaternions [140]. According to [140], the Euler angles representation is not fair and is thus numerically unstable, whereas the quaternions and angle-axis representations are fair parameterisations. In this Chapter, we opt for the quaternion parameterisation.

Remark C.9. *Comparisons with the angle-axis parameterisation seem to show better global convergence of the nonlinear optimisation algorithm results for the quaternion representation, which confirms the observation made by [140].*

The quaternion parameterisation of rotations is achieved via a mapping of S^3 , the unit sphere

in \mathbb{R}^4 into the special orthogonal group $SO_3(\mathbb{R})$:

$$R(q) = \begin{pmatrix} r_1(q) & r_2(q) & r_3(q) \end{pmatrix}, \quad (\text{C.12})$$

where

$$r_1(q) = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 \\ 2(q_1q_2 + q_0q_3) \\ 2(q_1q_3 - q_0q_2) \end{pmatrix}, \quad r_2(q) = \begin{pmatrix} 2(q_1q_2 - q_0q_3) \\ q_0^2 - q_1^2 + q_2^2 - q_3^2 \\ 2(q_2q_3 + q_0q_1) \end{pmatrix},$$

$$r_3(q) = \begin{pmatrix} 2(q_1q_3 + q_0q_2) \\ 2(q_2q_3 - q_0q_1) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix}.$$

Yet using the mapping (C.12) implies having a unit norm constraint on the quaternion in the optimisation problem. Instead we propose a slight refinement, assuming that the quaternion is non-zero, and take the following parameterisation:

$$R(q) = \frac{1}{\|q\|_2^2} \begin{pmatrix} r_1(q) & r_2(q) & r_3(q) \end{pmatrix}.$$

It can be verified that $R(q)$ is in $SO_3(\mathbb{R})$ for all $q \neq 0$. The quaternion can then be constrained to lie in $[-1, 1]$. Finally, when applying the quaternion formulation, the re-projection error minimisation turns into a box-constrained nonlinear least-squares of the form (C.6).

C.8 Application to crystal shape estimation

The algorithm described in Sections C.5, C.6 and C.7 has been tested on artificially generated images of crystals and real images recorded by the set-up depicted in Fig. C.1. This corresponds to the particular case, in which there are $N = 2$ views and the data dimension is $d = 3$. Several types of crystals have been considered such as Acetaminophen, Ascorbic acid, Ibuprofen, L-glutamic acid α and L-glutamic acid β . The simpler case of a cube has also been studied. Thus the proposed approach has been tested on simple and complex crystal shapes, demonstrating its efficiency.

C.8.1 Pre-processing: Extracting matching contours

The first step is a pre-processing of each of the two images in order to sample relevant data points. This pre-processing follows three steps:

1. Thresholding. Global thresholding is possible due to the even brightness distribution and high image quality.
2. Contour extraction from the binary image by applying a border following algorithm [146]. This is performed using the `openCV` function `cv::findContours`.
3. Find pairs of matching blobs on the two images based on the coordinates of the centroids of each blob.

The output contours of the pre-processing are depicted in blue in Fig. C.3. After normalizing, the

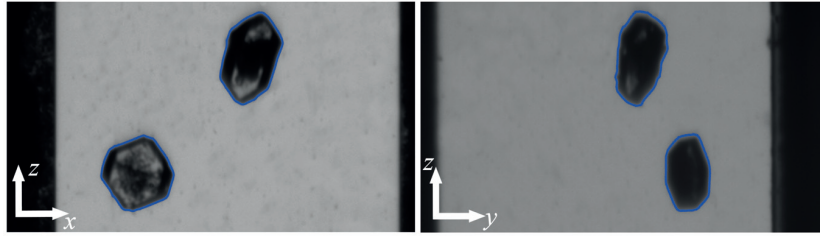


Figure C.3: Pair of images of two Ibuprofen crystal in water and extracted contours, as a blue lines.

pre-processing stage yields two sets of two-dimensional data points. From the ‘weak perspective’ hypothesis, it can be assumed that the data points are samples on the boundaries of the projections of the parametric polytope $P(A\tilde{R}^\top, B\tilde{t})$. The projections can be taken as projections onto the xy -plane and the xz -plane. We denote the obtained two sets of data points by $\mathcal{D}_1 = \{d_i^{(1)}\}_{i=0}^{M_1-1}$ for the xy -projection and $\mathcal{D}_2 = \{d_i^{(2)}\}_{i=0}^{M_2-1}$ for the xz -projection.

Remark C.10. *An additional pre-processing is to be applied to both sets of data points in order to remove outliers from the sets of data points \mathcal{D}_1 and \mathcal{D}_2 . Are considered as outliers, data points, which are unlikely to be the vertices of the projected polytope $P(A\tilde{R}^\top, B\tilde{t})$. The pre-processing stage consists in computing the vertices of the convex hull of the polygonal line produced by the Douglas-Peucker algorithm [53] applied to \mathcal{D}_1 and \mathcal{D}_2 . The resulting vertices are the points $w_k^{(i)}$ in the definition of the re-projection error (C.3). In Fig. C.4, it appears that the resulting polygon visually matches the data points quite well. In practice, the polygonal simplification procedure provides visually good approximations, yet no guarantees can be made that the vertices of the convex hull of the polygonal line produced by the Douglas-Peucker algorithm correspond to vertices of the projections of the rotated parametric polytopes. Several aspects of polygonal approximation algorithms are addressed in [75].*

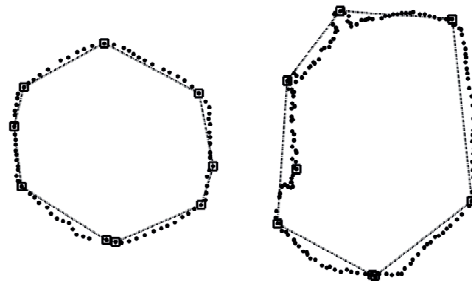


Figure C.4: Output of the Douglas-Peucker applied to a pair of images of Ibuprofen. The points extracted from the images appear as black dots. The black squares correspond to the vertices of the polygonal line produced by the Douglas-Peucker algorithm. The convex polygon corresponds to the convex hull of the output points of the Douglas-Peucker procedure.

C.8.2 Numerical results

The first step of our method is to compute a polyhedral partition of the parameter polytope, which can be done offline for each family of crystals. The polyhedral computations are performed using the toolbox MPT 3.0 [80]. Pictures of some polyhedral partitions obtained by applying the proposed algorithm to different crystal models are shown in Fig. C.5. Some partition polytopes along with the associated parametric polytopes are plotted in Fig. C.6. It clearly appear that from one critical region to its neighbour the shape of the parametric polytope is very different. For each of the five

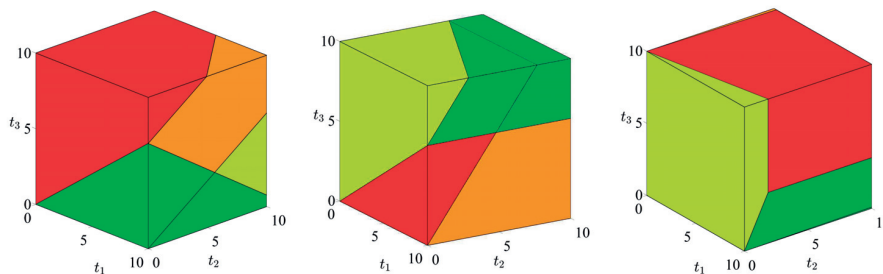


Figure C.5: Polyhedral partitions obtained for Acetaminophen, L-glutamic acid β and Ascorbic acid.

crystal families, data sets of 200 artificial and real images have been generated and the algorithm run on each of them. Examples of obtained fits are shown in Fig. C.7. As explained in the first part of the Chapter, the estimated shape parameter \hat{t} is multiplied with a scaling constant s yielding the size of the reconstructed polytope in μm .

For artificially generated images, the shape parameters are known, therefore the estimation error ϵ_t can be evaluated by computing the 2-norm of the difference. Statistics for artificially gen-

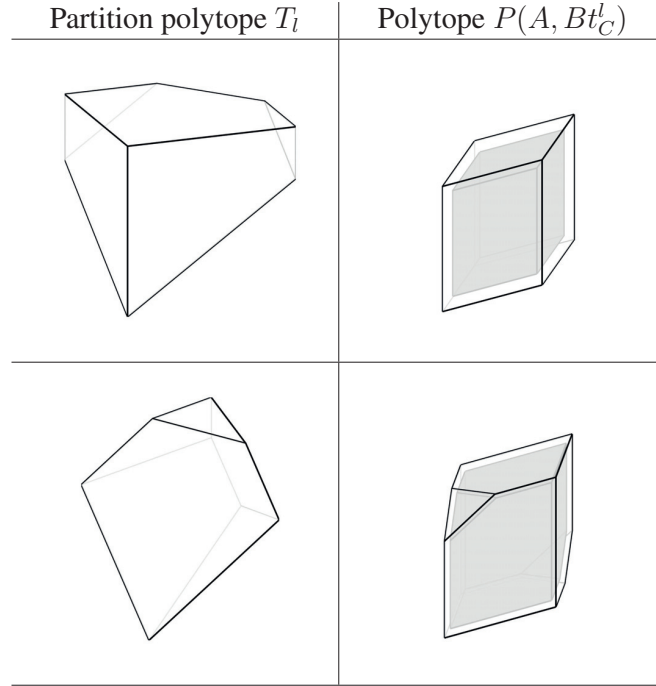


Figure C.6: Examples of partition polytopes for Acetaminophen and associated parametric polytopes. The shape parameter t_C^l is taken as the Chebychev center of the partition polytope for the grey polytopes, and a random vector around the Chebychev center for the transparent ones.

erated images are shown in Fig. C.8. It appears that in most cases the algorithm provides a very accurate estimate of the shape parameter, since the estimation error is generally very low (less than 1%). Low estimation errors are coupled with low re-projection errors, showing that the proposed nonlinear least-squares program is efficient for estimating the crystal shape parameter. On the contrary, on real crystal images, the quality of fit can only be compared in terms of the re-projection error, since the true shapes of crystals in suspension cannot be accurately evaluated in another way, the real shape parameters \tilde{t} are unknown. Statistics in Fig. C.9 show low re-projection errors (less than 5% in the case of Acetaminophen, Ibuprofen and L-glutamic acid β), yet higher than for generated images. This can be explained by the fact that real data contains a lot of outliers. Thus the algorithm has difficulties extracting relevant data vertices and the contour fitting approach does not perform as good as in the case of generated images. Yet visually good fits are obtained, as shown in Fig. C.7.

C.8.3 Computational aspects of the shape parameter estimation procedure

The crystal reconstruction procedure has been implemented in MATLAB using the toolbox MPT 3.0 ([80]) for polyhedral operations and an IPOPT MEX interface built on the parallel linear solver

APPENDIX C. PARAMETRIC POLYTOPE RECONSTRUCTION, AN APPLICATION TO CRYSTAL SHAPE ESTIMATION

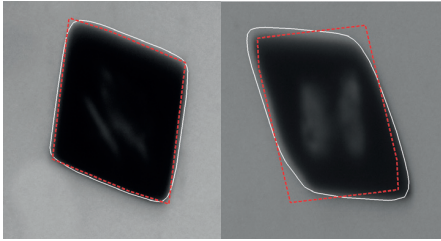
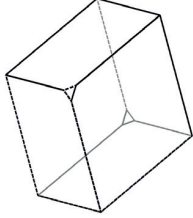
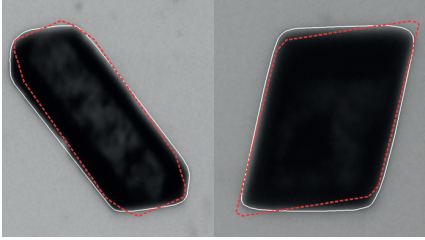
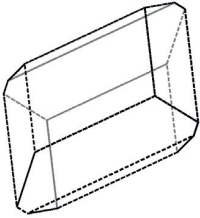
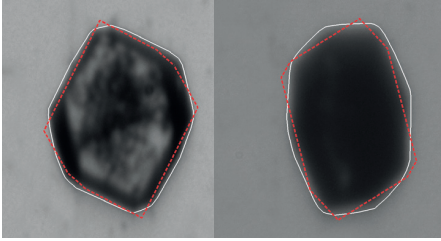
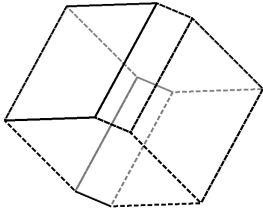
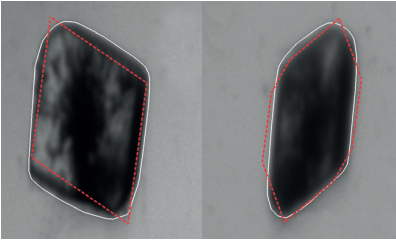
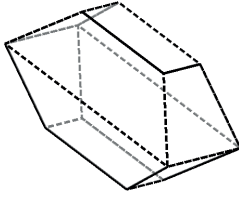
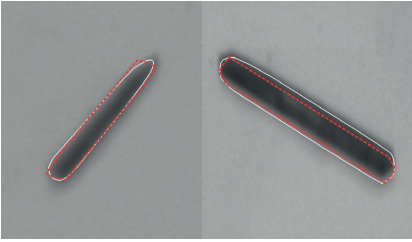
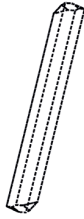
	(a) Projections	(b) 3-polytope	(c) $s\hat{t}$	(d) ϵ_{Π}
Acetaminophen			295 159 288	0.04
Ascorbic acid			60 90 167	0.06
Ibuprofen			296 193 190	0.08
L-glutamic acid α			240 208	0.10
L-glutamic acid β			607 42 485	0.10

Figure C.7: Fitting of different organic crystals. (a) Photographs with extracted contours (white) and fitted projections (dashed, red). (b) Reconstructed 3D polytope. (c) calculated scaling vector \hat{t} . The scalar s is a multiplying constant. (d) re-projection error.

APPENDIX C. PARAMETRIC POLYTOPE RECONSTRUCTION, AN APPLICATION TO CRYSTAL SHAPE ESTIMATION

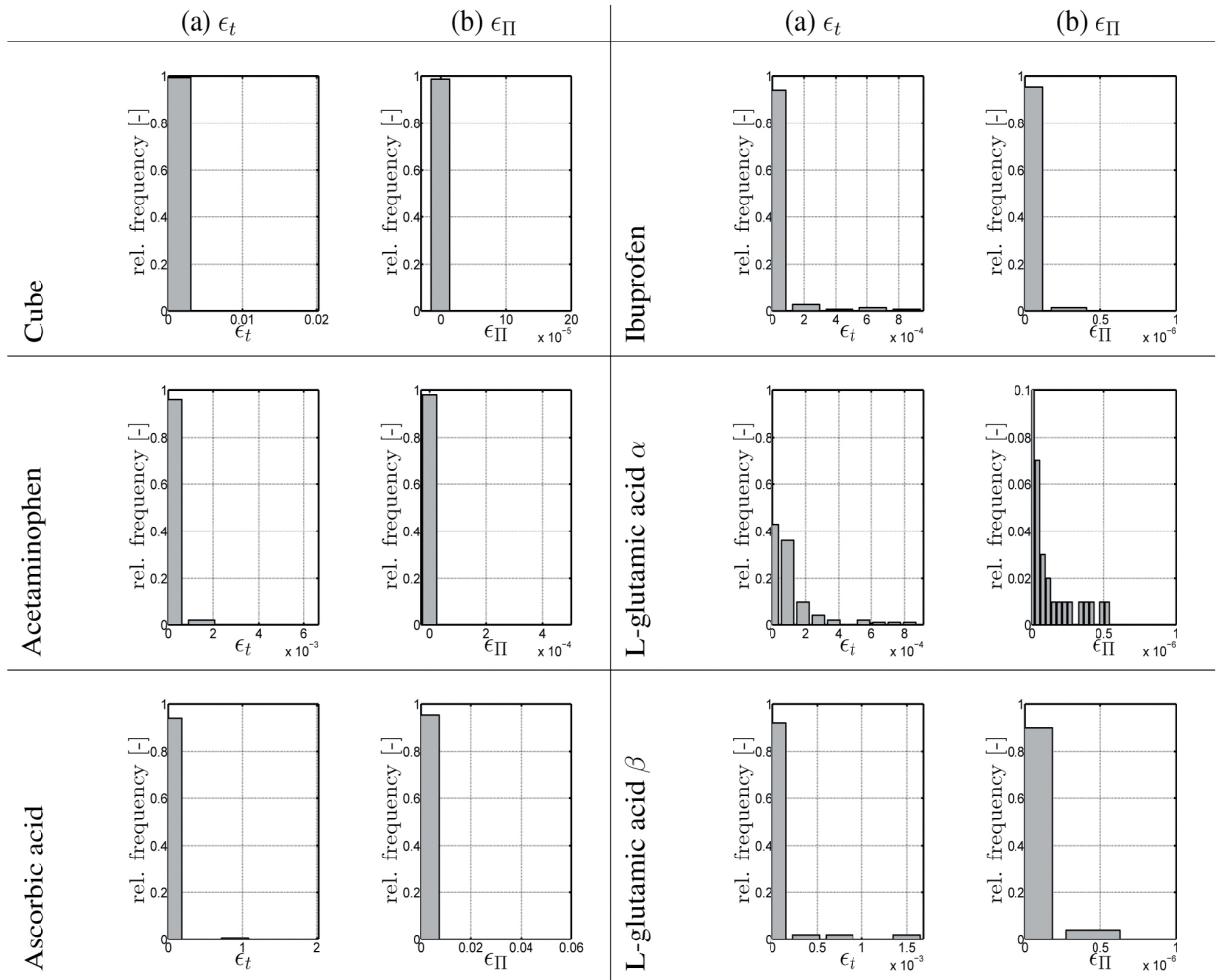


Figure C.8: Statistics for all simulated particles. (a) scaling error ϵ_t and (b) re-projection error ϵ_{Π} . MUMPS [6] for solving the nonlinear least-squares problem (C.6). For solving a single warm-started nonlinear optimization problem, the IPOPT CPU time is approximately 1.5 secs and the CPU time in function evaluations is approximately 5 seconds. Efficiency can be further improved by a C implementation of the objective, constraints function, jacobian of constraints and hessian of the Lagrangian. Furthermore, the reconstruction procedure is slowed by the fact that the nonlinear least-squares problem is to be solved to global optimality, which requires a grid of initial conditions. Subsequently, for estimating one shape parameter from a given pair of images, the entire procedure currently takes 20 to 200 secs.

C.8.4 Comparison to existing in-situ stereological methods

The two existing approaches, which can be compared to our approach, are the M-SHARC algorithm proposed in [107] and the ‘Projective Stereological Size and Shape Estimator’ of [124], as these

APPENDIX C. PARAMETRIC POLYTOPE RECONSTRUCTION, AN APPLICATION TO CRYSTAL SHAPE ESTIMATION

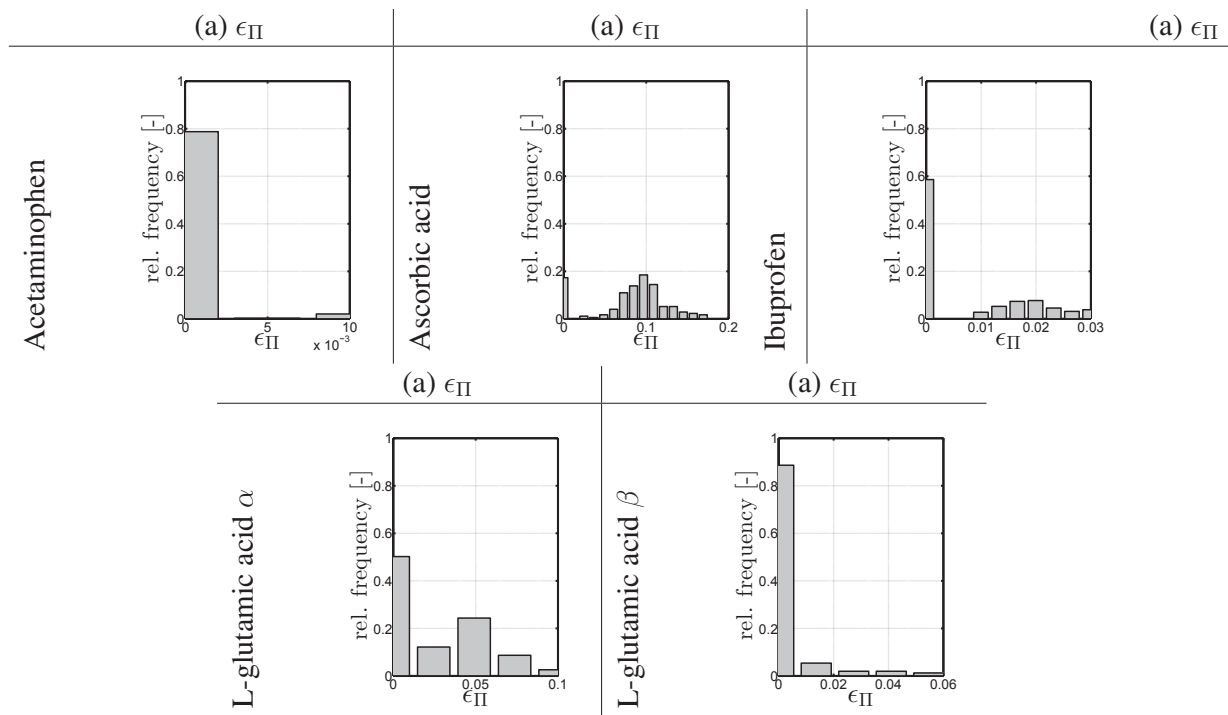


Figure C.9: Statistics for the re-projection error ϵ_{Π} for all photographed particles.

two procedures are both applied in an in-situ context, where a set of 2D views is available along with a crystal model.

First, the M-SHARC algorithm is based on finding correspondences between data and model primitives, which are made of linear features. Extracting linear features and finding correspondences between data and model can be costly in practice. It clearly appears in Table 2 of [107] that this step is actually the most costly part of the whole algorithm. On the contrary, our approach avoids the correspondence finding step by an appropriate choice of the re-projection error. Moreover, the M-SHARC algorithm relies on some assumptions on the crystal orientation, which is likely to hamper the accuracy of the method. In our approach, the orientation is estimated along with the shape parameter. Yet, our approach is still slower than the M-SHARC algorithm due to the costly optimisation step, in which a low objective needs to be found in order to ensure an accurate shape estimate.

The procedure proposed by [124] is applicable to any 3D convex body. Yet the shape model is quite restrictive, as only five types of objects are considered and the shape is modelled by two parameters, the length and elongation. Our approach is specifically targeted at any polytopic convex body, which seems to be more adapted to the crystal shape estimation problem.

Curriculum Vitae

Jean-Hubert Hours

born April 16th, 1987 in Lyon, France.

2011-2015 Doctorate at the Automatic Control Laboratory, EPF Lausanne

2009-2011 Master of Science in Electrical Engineering, ETH Zürich

2007-2009 Bachelor of Science in Electrical Engineering, Supélec, Paris

2005-2007 Preparatory classes, Mathematics-Physics, Lycée du Parc, Lyon

