GRAPH-STRUCTURED NONLINEAR PROGRAMMING: PROPERTIES AND ALGORITHMS

by

Sungho Shin

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
(Chemical Engineering)

at the
UNIVERSITY OF WISCONSIN–MADISON
2021

Date of final oral examination: May 21, 2021

The dissertation is approved by the following members of the Final Oral Committee:

Victor M. Zavala, Associate Professor, Chemical and Biological Engineering
Reid Van Lehn, Assistant Professor, Chemical and Biological Engineering
Ross Swaney, Associate Professor, Chemical and Biological Engineering
Xiangru Xu, Assistant Professor, Mechanical Engineering

To my dear wife Jungeun.

TABLE OF CONTENTS

Appendix

Page

LIST OF FIGURES

Appendix
Figure

Appendix
Figure

NOMENCLATURE

EDS	Exponential Decay of Sensitivity
OSM	Overalpping Schwarz Method
NLP	Nonlinear Program
gsNLP	Graph-Structured Nonlinear Program
QP	Quadratic Program
gsQP	Graph-Structured Quadratic Program
LS	Linear System
gsLS	Graph-Structured Linear System
BLH	Bounded Lagrangian Hessian
uBLH	Uniformly Bounded Lagrangian Hessian
gBLH	Globally Bounded Lagrangian Hessian
LICQ	Linear Independence Constraint Qualifications
uLICQ	Uniform Linear Independence Constraint Qualifications
gLICQ	Global Linear Independence Constraint Qualifications
(S)SOSC	(Strong) Second Order Sufficiency Conditions
uSSOSC	Uniform Strong Second Order Sufficiency Conditions
gSSOSC	Global Strong Second Order Sufficiency Conditions
SC	Strict Complementarity
KKT	Karush-Kuhn-Tucker
GE	Generalized Equations

ODE	Ordinary Differential Equations
DAE	Differential nad Algebraic Equations
PDE	Partial Differential Equations
MPC	Model Predictive Control
MHE	Moving Horizon Estimation
LQR	Linear Quadratic Regulator
RAS	Restricted Additive Schwarz
GMRES	Generalized Minimal Residual
DC	Direct Current
AC	Alternating Current
OPF	Optimal Power Flow
PSSE	Power System State Estimation
ALM	Augmented Lagrangian Method
SQP	Sequential Quadratic Programming
IPM	Interior Point Method
ADMM	Alternating Direction Method of Multipliers
ALADIN	Augmented Lagrangian-based Alternating Direction Inexact Newton method

LIST OF SYMBOLS

$\mathbb{R}$	Set of real numbers
$\mathbb{R}_{\geq 0}$	Set of non-negative real numbers
$\mathbb{R}_{>0}$	Set of positive real numbers
$\mathbb{I}$	Set of integers
$\mathbb{I}_{\geq 0}$	Set of non-negative integers
$\mathbb{I}_{>0}$	Set of positive integers
$\mathbb{I}_{[a,b]}$	$\mathbb{I} \cap [a, b]$
$\boldsymbol{I}$	Identity matrix
$\mathbf{0}$	Zero matrix or vector
$[M_1; \cdots ; M_n]$	$[M_1^\top \ \cdots \ M_n^\top]^\top$
$\{M_i\}_{i \in \mathcal{I}}$	$[M_{i_1}; \cdots ; M_{i_m}]$, where $\mathcal{I} := \{i_1 < \cdots < i_m\}$
$\{M_{i,j}\}_{i \in \mathcal{I}, j \in \mathcal{J}}$	$\{\{M_{i,j}^\top\}_{j \in \mathcal{J}}^\top\}_{i \in \mathcal{I}}$, where $\mathcal{I} = \{i_1 < \cdots < i_m\}$ and $\mathcal{J} = \{j_1 < \cdots < j_n\}$
$v[i]$	i-th component of v
$M[i, j]$	(i, j)-th component of M
$v[\mathcal{I}]$	$\{v[i]\}_{i \in \mathcal{I}}$
$M[\mathcal{I}, \mathcal{J}]$	$\{M[i, j]\}_{i \in \mathcal{I}, j \in \mathcal{J}}$
$\|\cdot\|$	vector 2-norms or induced 2-norm of matrix
$A \succ (\succeq) B$	$A - B$ is positive (semi)-definite
$A > (\geq) B$	each component of $A - B$ is greater than (or equal to) zero
$A + x$	$\{x' + x : x' \in A\}$

$f(X)$	$\{f(x) \in Y : x \in X\}$
$\nabla_w \varphi(x)$	$\{\frac{\partial}{\partial w[j]}\varphi(x)[i]\}_{i \in \mathbb{I}_{[1,m]}, j \in \mathbb{I}_{[1,s]}}$, where $w \in \mathbb{R}^s$ and $\varphi : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{yz} \phi(x)$	$\{\frac{\partial^2}{\partial y[i] \partial z[j]}\phi(x)\}_{i \in \mathbb{I}_{[1,p]}, j \in \mathbb{I}_{[1,q]}}$, where $y \in \mathbb{R}^p$, $z \in \mathbb{R}^q$, and $\phi : \mathbb{R}^n \to \mathbb{R}$

ABSTRACT

This dissertation studies graph-structured nonlinear programs, nonlinear optimization problems whose algebraic structures are induced by graphs. A graph-structured nonlinear program is a generalized abstraction for dynamic optimization, stochastic optimization, optimization with embedded partial differential equations, and network system optimization. This generalized abstraction allows studying properties these problems share in common and creating tailored solution algorithms.

In the first part, we study the fundamental property of solutions of graph-structured nonlinear programs. Specifically, we study how strongly a nodal parametric perturbation influences the nodal solution of another node. Building upon existing nonlinear programming sensitivity theory, we prove that solution sensitivity decays exponentially with the distance from the perturbation point on the graph. Remarkably, this result, which we call the exponential decay of sensitivity, holds under fairly standard assumptions used in classical nonlinear programming sensitivity theory: strong second-order sufficiency conditions and linear independence constraint qualifications. In addition, we establish uniform regularity conditions, the sufficient conditions under which the sensitivity decay rate remains uniformly bounded (independent of the size of the problem). Such uniformity allows studying sensitivity behaviors of problems with arbitrarily large graphs.

The exponential decay of sensitivity enables the creation of a specialized decomposition method for graph-structured nonlinear programs, the overlapping Schwarz method. In the second part, we study the convergence properties and implementation of this method. The overlapping Schwarz method has been traditionally used for the solution of sparse linear systems, which typically arise in discretized partial differential equations. In this work, we generalize this algorithm for graph-structured nonlinear programs. The proposed overlapping Schwarz algorithm partitions the original graph domain into a set of overlapping subdomains, which yields a set of coupled subproblems, and solves the subproblems in parallel and iteratively with the exchange of solution information at the boundary. The overlap is a crucial element in this method. Based on the exponential decay of sensitivity, we show that the algorithm locally converges if the overlap is sufficiently large and that the convergence rate improves exponentially with the size of overlap under certain conditions. We discuss several different variations of implementing the overlapping Schwarz method (problem level, subproblem level, and linear algebra level). Finally, with diverse numerical examples, we demonstrate the effectiveness of the proposed method.

ACKNOWLEDGMENTS

First and foremost, I'd like to express my sincere gratitude to my advisor Prof. Victor Zavala for his tireless support throughout my Ph.D. studies. He always encouraged me to come up with my own ideas, listened to my thoughts, and be excited when I make research progress (sometimes even more than I was). This not only gave me the confidence in doing research but also made doing research more enjoyable. In retrospect, I believe this positive reinforcement provided me with the opportunities to grow as a better researcher.

I am also grateful for my defense committee: Prof. Reid Van Lehn, Prof. Ross Swaney, and Prof. Xiangru Xu, and my preliminary exam committee: Prof. Jim Rawlings, Prof. James Dumesic, and Prof. Daniel Klingenberg. Especially, I was very fortunate to attend Prof. Rawlings' advanced process control class. Even after many years, what I learned from his lectures is inspiring my research.

I'd also like to thank my colleagues. The Zavala group has had an exceptionally warm and welcoming group culture, and I believe three of the first batch of students–Jordan Jalving, Ranjeet Kumar, and Apoorva Sampat–take full credit for building such a wonderful environment. Thanks to their warm and caring personalities, everyone in the group could feel welcomed, and all the group members, myself especially, were beneficiaries of such a great group culture. Moreover, I want to thank Joshua Pulsipher and Weiqi Zhang for carefully proofreading this dissertation. Also, I'd like to thank Yankai Cao, Qiugang Lu, Philip Tominac, Yicheng Hu, Alex Smith, and all the other group members for being awesome colleagues. One of the things I missed the most during the COVID-19 pandemic was exploring the great restaurants in every Friday lunch with them.

I'd like to acknowledge my first internship supervisor, and soon-to-be my postdoc mentor, Prof. Mihai Anitescu. His great insight has led us to the topic of overlapping decomposition, and it has become one of the central themes of my research. He always had brilliant research ideas and stimulated my intellectual curiosity. My graduate research has been tremendously benefited from the continued collaboration with him. I'd also like to thank my second internship supervisors, Dr. Carleton Coffrin and Dr. Kaarthik Sundar for their great help in developing my first Julia package `MadNLP.jl` and examining its capabilities with different energy infrastructure models. I'd also like to thank all the other collaborators, including Prof. Ophelia Venturelli, Prof. Timm Faulwasser, and Prof. Mario Zanon, and Sen Na. Each collaborative project gave me valuable lessons.

I thank Prof. Jong Min Lee for guiding me to pursue graduate studies. He was the one who made me excited about process control for the first time (with his inspiring undergraduate control course) and decide to come to Wisconsin to study it further. He has been a great mentor throughout my time in Wisconsin, even when he was 6000 miles away.

I also thank many open-source software developers around the world. Many of the numerical studies and software development projects would not have been possible without the open-source community. Especially, my research has been greatly benefited from using Julia Language. Thanks to its simplicity (but surprisingly without performance compromise), I was able to pick up the language quickly and generate cool results. My research has also benefited from many excellent open-source packages within Julia community like `JuMP.jl`, `Plots.jl`, `LightGraphs.jl`, and `PowerModels.jl`.

I'd also like to thank my great friends Alex Chew, Ho-jae Lee, Joonjae Ryu, and Hochan Chang for making my time in Madison much more memorable.

Finally, and most importantly, I want to thank my wife Jungeun for her unwavering love and caring. Her presence was absolutely the strongest emotional support and the motivation for moving forward.

Sungho Shin
Milwaukee, WI
May, 2021

Chapter 1

Introduction

Many decision-making problems in science and engineering are formulated as *structured optimization problems* whose algebraic structures are induced by graphs. Examples include dynamic optimization (the graph is a time horizon; e.g., optimal control, long-term planning, and state estimation) [18, 20], multi-stage stochastic programs (the graph is a scenario tree) [126, 143], optimization problems constrained by discretized partial differential equations (the graph is a discretization mesh) [23], and network optimization (the graph is a physical network; e.g., energy networks, supply chains) [38, 46, 97, 174]. The graphs associated with these problems are depicted in Figure 1.1. This observation motivates us to create a *unifying abstraction* for structured optimization based on graph theory. This graph-based, unifying abstraction gives rise to the main topic of the current dissertation, *graph-structured nonlinear programming* (gsNLP). This abstraction allows studying several seemingly different problem classes under a unifying perspective and enables discovering fundamental properties that these problems share in common. Furthermore, the study of such properties allows the creation of specialized solution algorithms for gsNLPs. This dissertation aims to study such properties and algorithms for gsNLPs.

In Part I, we study the sensitivity of gsNLPs; specifically, we establish a fundamental property of gsNLPs that we call the exponential decay of sensitivity (EDS). Our result implies that if a gsNLP satisfies the regularity conditions given by strong second-order sufficiency (SSOSC) and linear independence constraint qualifications (LICQ), the impact of a nodal data perturbation on the nodal solution on another node decays exponentially with respect

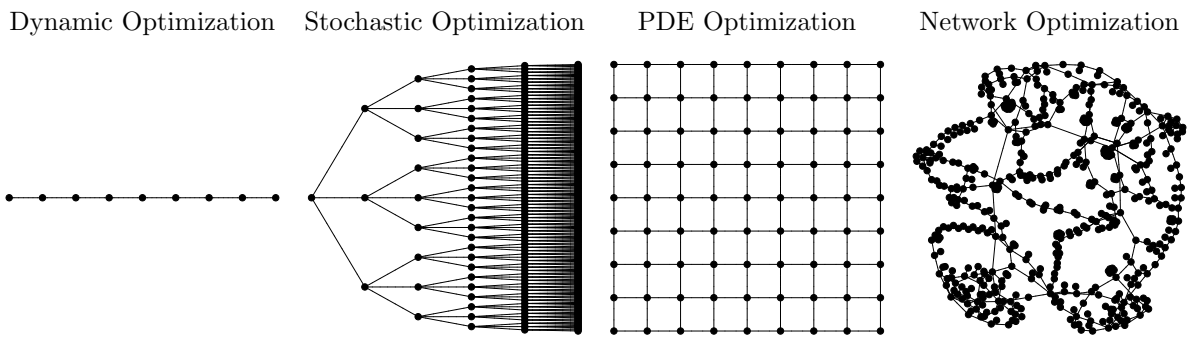

Figure 1.1 Illustration of graphs associated with various gsNLPs: dynamic optimization, stochastic optimization, optimization with embedded PDE, and network optimization.

to the distance from the perturbation point. This property holds for any gsNLP regardless of the topology of the graph. Furthermore, under the *uniform* regularity conditions, which are obtained by strengthening the SSOSC and LICQ, the decay rate can be set independently of the problem size. This allows us to establish EDS for gsNLPs whose size may be indefinitely large.

The EDS result has a number of practical implications. First, it helps understand and characterize the behavior of the solution with respect to variations in the data. For instance, in the context of dynamic optimization, EDS implies that the data in the far future (e.g., changes in the set-point or the disturbance forecast) time stages do not have a strong impact on the decisions in the current time step. This observation enables the creation of efficient discretization/approximation schemes for model predictive control (MPC) [84, 151, 166] and helps explain the empirical observation that the closed-loop performance loss of finite-horizon receding horizon control decreases with respect to the prediction horizon length [171]. In the context of energy network optimization, EDS indicates that the remote disturbance (e.g., changes in the energy demand or the generator capacity) does not have a strong impact on the decisions for the close generators; this observation leads to the creation of efficient decentralized control and estimation schemes, which in turn enables modular design and operation [153]. Moreover, EDS enables the creation of on-line optimization algorithms [120] as well as decomposition algorithms (the main topic of the second part) [121, 145, 147, 153].

In Part II, we study decomposition algorithms for gsNLPs. Many real-world examples of gsNLPs are formulated as large-scale optimization problems which may defy the scope of centralized, off-the-shelf optimization solvers. To address such a challenge, diverse decomposition schemes for large-scale NLPs have been proposed in the literature; the existing methods include Lagrangian decomposition [96, 111, 128, 137, 147], the alternating direction method of multipliers (ADMM) [27, 148], the augmented Lagrangian based alternating direction inexact Newton method (ALADIN) [56], parallel Newton methods [36, 136], and Jacobi/Gauss-Seidel methods [150]. The basic tenet behind such algorithms is to decompose the original intractable problem into smaller tractable subproblems and to coordinate subproblem solutions by exchanging the primal-dual solution information. However, a disadvantage of these schemes is that convergence can be rather slow; in most cases the convergence is linear and the convergence rate may be arbitrarily close to one. Thus, obtaining a high-precision solution can be challenging in practice (e.g., see [107] for a benchmark of different decomposition techniques).

The *Overlapping Schwarz* method (OSM) was originally developed for the solution of large, sparse linear systems, which typically arise in discretized partial differential equations (PDEs) [69]. We have recently generalized this method for graph-structured optimization and demonstrated that it can effectively solve large graph-structured optimization instances [121, 145, 147, 153]. As its name suggests, the OSM decomposes the full problem into a set of smaller *subproblems* that are defined over *overlapping subdomains*. Then the subproblems are solved in parallel and iteratively, and the convergence is promoted by exchanging the primal-dual solution information at the boundary. The OSM is designed to exploit the EDS using overlap, and the overlap plays a crucial role in the convergence of the algorithm. In particular, we show that for general gsNLPs satisfying the regularity conditions and the polynomial growth condition on the underlying graph topology, the local convergence of the OSM is guaranteed if the size of overlap is sufficiently large, and the convergence rate improves exponentially with the size of overlap. This result generalizes the convergence result of our previous work, which were under quadratic programming [145, 153] and nonlinear

optimal control settings [121, 144]. With diverse numerical examples, we demonstrate that this decomposition method is highly effective in solving large-scale gsNLPs.

In the remainder of the current chapter, we formally introduce the formulation of gsNLP. Then, the relationship with special cases (dynamic, stochastic, PDE, and network optimization) are discussed. The chapter concludes with an outline of the dissertation.

1.1 Graph-Structured Nonlinear Programming

1.1.1 Formulation

Graph-structured nonlinear programs are formulated as follows:

$$\min_{\{x_i\}_{i\in\mathcal{V}}} \sum_{i\in\mathcal{V}} f_i(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]};\{p_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) \tag{1.1a}$$

$$\text{s.t. } c_i^E(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]};\{p_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) = 0,\ i\in\mathcal{V},\ (y_i^E) \tag{1.1b}$$

$$c_i^I(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]};\{p_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) \geq 0,\ i\in\mathcal{V},\ (y_i^I). \tag{1.1c}$$

Here, $\mathcal{G}=(\mathcal{V},\mathcal{E})$ is an undirected graph with the nonempty, strictly ordered node set $\mathcal{V}$ and the edge set $\mathcal{E}\subseteq\{\{i,j\}\subseteq\mathcal{V}:i\neq j\}$; $\mathcal{N}_\mathcal{G}[i]:=\{j\in\mathcal{V}:\{i,j\}\in\mathcal{E}\}\cup\{i\}$ denotes the closed neighborhood of $i\in\mathcal{V}$ on $\mathcal{G}$; for each node $i\in\mathcal{V}$, $x_i\in\mathbb{R}^{n_{x_i}}$ is the primal variable vector; $p_i\in\mathbb{R}^{n_{p_i}}$ is the data vector; $f_i:\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{x_j}}\times\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{p_j}}\to\mathbb{R}$ is the objective function; $c_i^E:\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{x_j}}\times\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{p_j}}\to\mathbb{R}^{n_{y_i^E}}$ is the equality constraint vector function; $c_i^I:\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{x_j}}\times\prod_{j\in\mathcal{N}_\mathcal{G}[i]}\mathbb{R}^{n_{p_j}}\to\mathbb{R}^{n_{y_i^I}}$ is the inequality constraint vector function; and $y_i^E\in\mathbb{R}^{n_{y_i^E}}$ and $y_i^I\in\mathbb{R}^{n_{y_i^I}}$ are the dual variable vectors associated with (1.1b) and (1.1c), respectively. To enable compact notation, we define: $c_i(\cdot):=[c_i^E(\cdot);c_i^I(\cdot)]$; $y_i:=[y_i^E;y_i^I]$; $z_i:=[x_i;y_i]$; $n_{y_i}=n_{y_i^E}+n_{y_i^I}$; and $n_{z_i}=n_{x_i}+n_{y_i}$. Furthermore, we define $\boldsymbol{x}:=\{x_i\}_{i\in\mathcal{V}}$; $\boldsymbol{y}^E:=\{y_i^E\}_{i\in\mathcal{V}}$; $\boldsymbol{y}^I:=\{y_i^I\}_{i\in\mathcal{V}}$; $\boldsymbol{y}:=\{y_i\}_{i\in\mathcal{V}}$, $\boldsymbol{z}:=\{z_i\}_{i\in\mathcal{V}}$; and $\boldsymbol{p}:=\{p_i\}_{i\in\mathcal{V}}$.

The gsNLP is composed of a collection of nodes $i\in\mathcal{V}$ wherein each node i has its own nodal variables x_i, nodal data p_i, nodal objective function $f_i(\cdot)$, and nodal constraint functions $c_i^E(\cdot), c_i^I(\cdot)$. However, the problem is not separable for each node because there exists inter-node coupling through the objective and constraint functions. The nodal objective and constraint functions at node i depend on the variables $\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}$ and data $\{p_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}$ in the

neighboring nodes $\mathcal{N}_{\mathcal{G}}[i]$. In other words, node i is coupled algebraically to its neighbors, and the topology of such connectivity is dictated by the graph $\mathcal{G}$. Throughout the dissertation, we assume that the variables in (1.1) are continuous variables (as opposed to discrete variables in the mixed integer programming context), and the functions in (1.1) are sufficiently smooth (formal assumption will be introduced later), but potentially nonconvex. Thus, the problem is cast as a nonconvex, equality and inequality-constraint nonlinear program (NLP). Throughout the paper, when we refer to something as a solution, it means a local solution in the sense of [124, Chapter 12].

Remark 1.1 (Generality of Problem (1.1))**.** *In some applications (e.g., energy networks), we might encounter variables, data, objectives, and constraints defined over edges (not explicitly expressed in Problem (1.1)). Such information can be captured within "super-nodes" that encapsulate edges (this is possible because the formulation allows for nodes with different numbers of variables and constraints). Alternatively, one may treat edges in the graph as nodes and rewrite the problem with a newly defined "lifted graph"* $\widetilde{\mathcal{G}} = (\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}})$*, where* $\widetilde{\mathcal{V}} := \mathcal{V} \cup \mathcal{E}$ *and* $\widetilde{\mathcal{E}} := \{\{i, e\} : i \in \mathcal{V},\ i \in e,\ e \in \mathcal{E}\}$ *(an order needs to be assigned for* $\widetilde{\mathcal{V}}$*). In this way, one can reformulate the problem so that it fits into (1.1) form. Thus, the problem under study (1.1) is adequately general to capture diverse settings in different applications.*

Remark 1.2. *The formulation in (1.1) assumes that the modeler is aware of the graph structure. However, such intuition may not be readily available to the modeler. In such a case, one may still model the problem as a graph-structured problem by viewing each primal and dual variable as a node. One can construct the set of edges later by inspecting the connectivity pattern in the objective and constraints.*

Related Work: The exploitation of graph structure within optimization has received increasing attention in the literature. The study of structured optimization problems traces back to the early works of Dembo [7, 46]; therein, nonlinear network optimization models,

which generalize hydroelectric power system scheduling, financial planning, matrix estimation, air traffic control problems, were studied. The authors also provide numerical experiments demonstrating that a structure-exploiting solver can be significantly faster than a general-purpose one. A similar type of network-constrained abstraction is utilized in a number of other works [8]. The algebraic structure in the optimization problem is often exploited in the context of decomposition. Rantzer and coworkers studied the use of the dual decomposition method for solving optimization problems in a distributed manner, where the problem is decomposed for each subsystem [128]. The stability properties of such distributed controllers have been studied in subsequent works [74–76]. A similar (but often more performant) method ADMM is also widely used for exploiting the block structure within the optimization problems [59, 131, 136, 148]. Houska, Diehl, and coworkers have developed an augmented Lagrangian-based alternating direction inexact Newton method, which solves a structured optimization problem by exploiting structure in the subproblem solution [91]. Although the authors did not explicitly make the connection with the graphs, the separability in the objective and the inequality constraints implicitly assume that the algebraic structure of the problem is induced by the sparsity of linear constraint mapping. Such a method is applied for the distributed solution of structured optimization problems, such as optimal power flow and optimal control [57, 106]. In subsequent work, nonlinear optimization problems over tree graphs were studied and a specialized algorithm has been developed [100]. Not only these, there exist myriads of works in the literature where the problem's algebraic structure is exploited to enable more efficient and scalable computations [36, 40, 58, 79, 94, 102, 137]. Tang, Allman, Daoutidis, and coworkers have studied the community detection method that identifies the optimal decomposition of structured nonlinear optimization problems [9, 44, 159]. The modeling aspect of graph-structured optimization problems is studied in [97–99]; therein, the authors have posed an optimization problem of which algebraic structure is induced by a graph. Then, the practical advantages of modeling the problem in such a form are discussed; specifically, it facilitates modular construction of the model and the use of decomposition strategies.

1.1.2 Special Cases

The gsNLP formulation (1.1) is a unifying abstraction for dynamic, stochastic, PDE, and network optimization. We now discuss each of these examples and explain how they can be abstracted as gsNLPs.

Dynamic optimization problems are the problems in which a series of decisions are made over a specified time horizon [130]. Typically, the summation (integration in the case of a continuous-time problem) of stage cost functions over a specified time horizon is minimized while the dynamics of the system are enforced as constraints for the variables in the adjacent time stages. Examples of dynamic optimization problems include model predictive control [18, 20, 130], state estimation [129], multi-period optimal power flow [73, 105], transient gas network operations [174, 175]. The algebraic structure of dynamic optimization problems can be represented as a chain of temporal dependencies (see Figure 1.1); here, the time indices represent the nodes and the dynamic coupling produces the edges. The state and input variables in each stage constitute the nodal decision variables, dynamic and state/input constraints constitute the nodal constraints, the stage cost functions constitute the nodal objective function terms, and the set-points and disturbance forecasts constitute the nodal data. Dynamic optimization problems are often formulated as a semi-infinite continuous-domain optimization problem constrained by ordinary differential equations (ODEs) or differential and algebraic equations (DAEs). Those problems can be converted into finite-dimensional NLPs by applying discretization techniques [22]. For such cases, the temporal dependency pattern makes discretized problems gsNLPs with linear graphs. We discuss dynamic optimization problems further in Chapter 4 and study their sensitivity properties. The readers are referred to Appendix B.1, B.2, B.8, B.9 for practical examples of dynamic optimization.

Decision-making problems under uncertainty over multiple time stages can be formulated as multi-stage stochastic programs [93, 104, 114, 126]. Multi-stage stochastic programming framework has been successfully adopted in energy system planning [126], capacity and production planning [93, 104], water resources management [114], battery system control [108]. These problems seek to minimize the expected cost over the specified temporal horizon while

seeking to satisfy the dynamic coupling constraints and state/input constraints. In particular, they make decisions over an uncertainty scenario tree, wherein each node represents a particular realization of the uncertain parameters up to a certain time stage. The dynamic and state/input constraints are enforced for each node, and the objective function is expressed by a weighted sum of the nodal objectives (the weight is the probability of the realization of the node). Such a multi-stage stochastic program can be considered as a gsNLP whose graph is a scenario tree. The decision variables associated with each node in the scenario tree can be considered as nodal decision variables, and the uncertain parameters in each node can be considered as the nodal data, and so on. Oftentimes, the uncertainty space is continuous (e.g., the uncertain parameter is normally distributed), and a certain discretization method needs to be applied to construct a scenario tree. The readers are referred to Appendix B.3 for a practical example of stochastic optimization.

Control and optimization problems for distributed systems are often cast as optimization problems constrained by partial differential equations and boundary conditions. These are often called PDE-based, or PDE-constrained optimization [22, 23, 45, 88]. Typical PDE-constrained optimization problems seek to minimize the integration of certain performance metric functions over the domain of interest by making the best use of its flexibility (usually by manipulating the controlled variables) while satisfying the PDEs and the boundary conditions. Since the original problem is an infinite-dimensional continuous-domain optimization problem, in most cases, analytically solving the problem is not possible, and an approximate solution is often sought. Such an approximate solution can be obtained by solving a discretized PDE-constrained optimization problem [87]. These problems are formulated as large-scale NLPs that directly embed the discretization of the PDEs over the domain of interest. These problems can be considered as gsNLPs induced by its discretization mesh, wherein the discretization points in the mesh become the node and the discretized differential expressions produce the couplings between the nodes. The readers are referred to Appendix B.4 for a practical example of PDE-based optimization.

Many optimization problems for network systems fall into the category of graph-structured problems. For example, optimal power flow problems seek to determine the best-operating levels for the power generators in the network that minimally use the operation cost while meeting the given electricity demands throughout the network [39,67,95,156]. For this problem, the power network itself can be regarded as the graph for the problem. The decision variables for each node (power generation and voltages) can be regarded as the nodal variables, the physical limits and power flow equations can be regarded as nodal constraints, the nodal operation costs can be regarded as the nodal objective functions, and the electricity demand at the nodes can be regarded as nodal data. Similarly, problems arising in power system state estimation (PSSE) [6,118,141], natural gas network operations [174,175], and supply chains [79] can also be formulated as gsNLPs. Oftentimes, the problems are cast as dynamic optimization problems over networks possibly with embedded partial differential equations; one of such examples is the transient gas network operation/estimation problem [73,105,174,175]. The readers are referred to Appendix B.6, B.5 for practical examples of network optimization.

Finally, we discuss what type of problems are *not* gsNLPs of interest. Trivially, any NLP can be modeled as a single-node gsNLP in which all the variables, constraints, objective terms, and data are located within a single node. However, such an abstraction makes our subsequent theoretical analysis trivial. For example, our EDS results characterize the interplay between multiple nodes and use the distance between the nodes to characterize the decay in the sensitivity; in the single-node graph case, there are no inter-node interactions, and the distance is always zero. Thus, in order for our analysis to be non-trivial, the graph needs to be sparse and substantially large in its diameter (the longest distance between two nodes on the graph). Alternatively, one can model the problem as a complete graph (as in Figure 1.2, first panel), but this also makes the diameter of the graph one. There exists a number of large-scale nonlinear programming instances that do not admit such sparse and large graphs. First, any type of dense optimization problem does not admit such a graph. Those dense models often arise from statistical model learning problems. This is because,

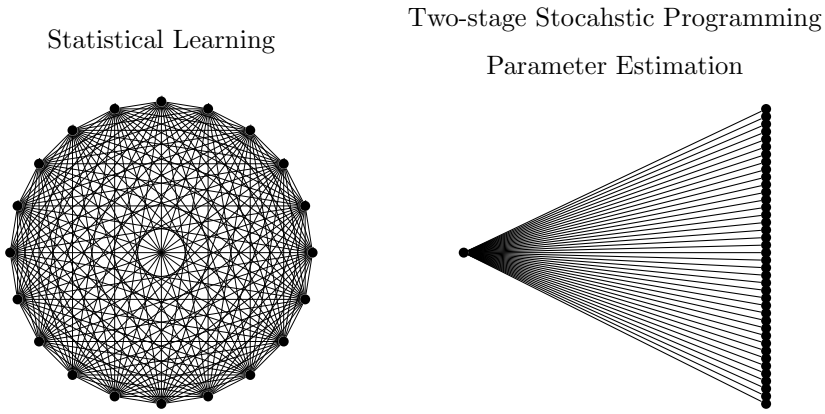

Figure 1.2 Illustration of graphs associated with non-gsNLPs: statistical learning, two-stage stochastic programming and parameter estimation.

for typical statistical models, the structure of the model is dense; i.e., all the variables are coupled with other variables [78, 162]. Also, even if the problem itself is sparse, there may not exist a graph that has a sufficiently long diameter; two-stage stochastic optimization problems [13, 25] and parameter estimation problems [149, 172] are such examples. For those problems, there exists a set of variables that are coupled with all the other variables (those could be first-stage variables in the two-stage stochastic program or the global parameters in parameter estimation problems). Due to the existence of such variables, the distance between any pair of nodes is not greater than 2; consequently, this makes our results trivial. The graphs associated with these problems are depicted in Figure 1.2.

1.2 Outline

The rest of the dissertation is divided into two parts, followed by the conclusions. In Part I, the results on the EDS are presented, and in Part II, the results on the OSM are presented. The detailed outlines of each chapter are as follows.

Chapter 2: This chapter presents the main theoretical results on EDS. First, we study the properties of graph-induced matrices. Our result states that the i, j-block of a graph-structured matrix decays exponentially with the distance between i and j on the graph, where i, j are the nodes in the graph. Then, this result is applied to classical NLP sensitivity

theory, yielding the main theorem: under the regularity conditions (SSOSC and LICQ), the sensitivity of the nodal solution decays exponentially with respect to the distance from the perturbation point.

Chapter 3: This chapter presents the uniform regularity conditions for gsNLPs. These conditions are strengthened versions of the standard regularity conditions, used in Chapter 2. They enable uniformly bounding the sensitivity decay rates derived in Chapter 2 by constants that are independent of the problem size. Thus, under uniform regularity, the sensitivity decay rate remains unchanged even if the size of the problem is extended indefinitely. However, analytically checking the satisfaction of uniform regularity for indefinitely growing problems can be challenging. To address this issue, we also provide composability principles for uniform regularity; with this, one can check the uniform regularity for the smaller blocks and then compose these conditions to obtain the uniform regularity for the full problem.

Chapter 4: This chapter presents the application of the results in Chapter 2, 3 to dynamic optimization problems. We apply the composability principles in Chapter 3 to show the uniform regularity of the dynamic optimization problems for controllable and observable dynamical systems. This showcases the capability of the composability principles presented in Chapter 3. Our result reveals the relationship between controllability and LICQ and that between observability and SSOSC.

Chapter 5: This chapter presents numerical demonstrations of the EDS using diverse examples of gsNLPs, including dynamic, stochastic, PDE, and network optimization. Furthermore, we illustrate how the satisfaction of regularity conditions affects the decay rate. This concludes Part I.

Chapter 6: This chapter presents the OSM for solving gsNLPs and analyzes its local convergence based on the EDS. The subproblem for the OSM is constructed by collecting the variables, objective terms, and constraints over the associated subdomain and incorporating the coupled constraints as augmented Lagrangian. The coupled variables are fixed to the current guess of the solution. We show that this subproblem formulation satisfies the consistency (the full solution meets the first-order condition for the subproblems) and

inheritance of regularity (subproblems satisfy uniform regularity if the full problem satisfies uniform regularity). This allows us to apply the results in Chapter 2, 3 to analyze the sensitivity of the subproblems, and show that the impact of misspecification of the boundary solutions decays exponentially. Using such a sensitivity result, we analyze the convergence of the algorithm; specifically, we show the exponential improvement of the local convergence rate with respect to the size of overlap.

Chapter 7: This chapter presents the specializations of the results in Chapter 6 for graph-structured convex quadratic programs (gsQP). The QP setting allows establishing a stronger convergence result. In particular, under the strengthened uniform regularity conditions, which we call global regularity, we show the global EDS. With this global sensitivity result, we show the global convergence of the OSM.

Chapter 8: This chapter presents the specializations of the results in Chapter 6 for graph-structured linear systems (gsLS). We target indefinite and potentially non-symmetric matrix, as the systems with definite matrices can be considered as QPs. Although any gsLS is a gsQP whose objective is constant, we discuss a simpler algorithm than the direct application of the result in Chapter 7; this method facilitates a flexible and efficient implementation. We establish the global convergence of the proposed method, but this result should be caveated, as one cannot guarantee the bounded conditioning of the subproblems. Here, we also establish a formal connection with the OSM for sparse linear systems (commonly referred to as restricted additive Schwarz).

Chapter 9: This chapter discusses the implementation of the OSM for solving gsNLPs. Here, we discuss not only the implementation of solution algorithms but also the implementation of algebraic and graph-based modeling tools. This is because, for the implementation of OSM, a flexible algebraic and graph-based modeling tool is required; thus, the modeling interface is not a separate issue. We show that our implementation of algebraic and graph-based modeling interface allows such flexible manipulation. For the solution algorithm, we discuss applying overlapping Schwarz at three different levels: problem-level, subproblem-level, and linear algebra-level. Problem-level decomposition applies overlapping Schwarz

directly to the problem, while subproblem and linear algebra-level decomposition apply the OSM within a certain NLP solution algorithm (e.g., augmented Lagrangian method, sequential quadratic programming, interior point method).

Chapter 10: This chapter showcases the capability of overlapping Schwarz decomposition (in particular, problem-level decomposition and linear algebra-level decomposition) for solving large-scale gsNLPs. We compare the performance with centralized solver Ipopt and standard decomposition method ADMM. The results indicate that the OSM is certainly much faster than ADMM, and also faster than the centralized solver for sufficiently large instances. This chapter concludes Part II.

Chapter 11: This chapter concludes the dissertation with a summary of the contributions and a brief overview of the future research plan.

How to Read: The dissertation can be read sequentially. However, if the reader is mainly concerned about the theoretical development, the reader can only read Chapter 2, 3, 6, and skip the rest of the chapters, as the main theoretical contributions are concentrated in those chapters. Chapter 4, 7, 8 present the specialization of the main theoretical results to more specialized settings; they do not simply present the corollaries, but they present further insights one can obtain from a more specialized context. As such, if any of the particular settings is of interest to the reader, it is recommended to read those chapters too. If the reader is mainly concerned about the convergence result for linear systems, reading Section 2.1 and Chapter 8 would be sufficient. If the reader is mainly interested in the implementation of the OSM, the reader can directly read Chapter 9; however, to understand the algorithm more deeply, it is recommended to also read Chapter 2, 3, 6, 7, 8.

Relationship with the Published Works: The results in Chapter 2, 3, 5 are recently submitted for publication in [146], and the results in Chapter 4 is submitted for publication in [152]. The results in Chapter 6, 7, 8 are loosely based on [121, 145, 147, 153], but they are significantly modified from the published results, and many new results are added. A number of numerical studies in Chapter 10 are presented in the published works [145, 147, 153], but many of them are first presented in this dissertation. The newly presented results in

Chapter 6, 7, 8, 9, 10 will be submitted for a journal publication after depositing the current dissertation.

Part I

Properties

Chapter 2

Exponential Decay of Sensitivity

This chapter seeks to answer the following question:

$$\begin{gathered} \textit{How does the primal-dual solution at node } i \in \mathcal{V} \textit{ change} \\ \textit{when the data at node } j \in \mathcal{V} \textit{ is perturbed?} \end{gathered} \tag{Q1}$$

We provide an answer to this question by identifying conditions under which we can find *nodal* sensitivity coefficients $\{C_{ij} \in \mathbb{R}_{\geq 0}\}_{i,j\in\mathcal{V}}$ satisfying:

$$\|z_i^\dagger(\boldsymbol{p}) - z_i^\dagger(\boldsymbol{p}')\| \leq \sum_{j\in\mathcal{V}} C_{ij}\|p_j - p_j'\|, \quad i \in \mathcal{V}, \tag{2.1}$$

where $\boldsymbol{z}^\dagger(\boldsymbol{p})$ and $\boldsymbol{z}^\dagger(\boldsymbol{p}')$ are solutions of NLP (1.1) for data $\boldsymbol{p}$ and $\boldsymbol{p}'$, respectively. Moreover, $\boldsymbol{p}$ and $\boldsymbol{p}'$ are perturbations of the base data $\boldsymbol{p}^\star$ (associated with base local primal-dual solution $\boldsymbol{z}^\star$).

Our main result in this chapter (Theorem 2.2) shows that if: (i) the strong second order sufficiency condition (SSOSC) and the linear independence constraint qualification (LICQ) hold at the base solution $\boldsymbol{z}^\star$, and (ii) $\boldsymbol{p}$ and $\boldsymbol{p}'$ are sufficiently close to $\boldsymbol{p}^\star$, then (2.1) holds with $C_{ij} = \Upsilon\rho^{\lceil d_{\mathcal{G}}(i,j)/4-1\rceil_+}$. Here, $\Upsilon > 0$ and $\rho \in (0,1)$ are constants, $d_{\mathcal{G}}(i,j)$ is the graph distance between nodes i and j on $\mathcal{G}$, and $\lceil\cdot\rceil_+$ denotes the smallest non-negative integer that is greater than or equal to the argument. In other words, *solution sensitivity decays exponentially* with respect to the distance to the perturbation point. We call this property *exponential decay of sensitivity* (EDS). This result is a specialization of classical sensitivity results for general NLPs [26,51,63,132,134] to a graph-structured setting. Specifically, classical results establish an *overall* sensitivity coefficient $C \in \mathbb{R}_{\geq 0}$ satisfying $\|\boldsymbol{z}^\dagger(\boldsymbol{p}) - \boldsymbol{z}^\dagger(\boldsymbol{p}')\| \leq C\|\boldsymbol{p} - \boldsymbol{p}'\|$, while here

we establish *nodal* sensitivity coefficients $\{C_{ij}\}_{i,j\in\mathcal{V}}$ satisfying (2.1). Our results thus provide intuition into how perturbations propagate through the structure of the NLP.

Related Work: Nonlinear programming sensitivity has been extensively studied in the literature; early works include [63, 132, 176]. Those works study the sensitivity of nonlinear programs under the twice continuous differentiability, SOSC, LICQ, and strict complementarity (SC) assumptions. The SC allows locally reducing the Karush-Kuhn-Tucker (KKT) conditions to a set of nonlinear equations. Thus, the local sensitivity of the solutions of the KKT conditions can be studied based on the standard implicit function theorem. Those works show the Lipschitz continuity of the local solution mapping. Subsequently, the KKT conditions for the nonlinear programs have been generalized into so-called generalized equations (GEs) [133, 135]. In [134], the sensitivity results of the solutions of GE have been established based on the strong regularity assumptions, which are the regularity conditions adopted in our analysis. The analysis based on GEs allows relaxing the SC assumption but requires assuming strong second sufficiency (but certainly SSOSC+LICQ is weaker than SSOSC+LICQ+SC). Subsequently, these results have been published as a part of a review paper [64] and books [26, 51]. Question (Q1) has been recently addressed in specific settings such as nonlinear dynamic optimization [119, 121, 144, 165] and graph-structured quadratic programs [145, 153]. Our work generalizes such results for nonlinear settings.

Addressing (Q1) is crucial for understanding solution stability of a wide range of problem classes, for designing approximation schemes (often cast as parametric perturbations) [21, 49, 120, 151, 165], and for designing solution algorithms [121, 144, 145]. For instance, it has been recently shown that EDS plays a central role in assessing the impact of coarsening schemes [84, 151] for dynamic optimization and in establishing convergence of OSM for graph-structured problems [121, 144, 145]. From an application standpoint, our results provide new insights on how perturbations propagate through graphs and on how the problem formulation influences such propagation. Specifically, we provide empirical evidence that positive objective curvature and constraint flexibility tend to dampen propagation (promote sensitivity decay). Such insights can be used, for instance, to design systems that dampen

(or magnify) perturbations or to identify system elements that are sensitive (or insensitive) to perturbations.

2.1 Graph-Structured Matrix Properties

2.1.1 Distance and Bandwidth

This section derives basic properties of *graph-structured matrices*. The results in this section will be crucial in deriving the sensitivity results of interest. In particular, the study of nodal sensitivity eventually boils down to the analysis of graph-structured matrices. This is because the sensitivity of the solution of nonlinear programs can locally be evaluated by that of quadratic programs, and the solution sensitivity of quadratic programs can be expressed by a solution of linear systems. Thus, the results established in this section will serve as an analytical tool for the subsequent analysis.

Properties of graph-structured *positive definite* matrices are reported in [47, 153]; here, we establish properties for general (non-symmetric and indefinite) matrices. We begin by introducing the notion of distance on graphs and establish basic properties for such distance.

Definition 2.1 (Graph Distance and Diameter)**.** *The distance $d_{\mathcal{G}}(i,j)$ between nodes $i,j \in \mathcal{V}$ on a finite, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the number of edges in the shortest path connecting them. If $i = j$, $d_{\mathcal{G}}(i,j) = 0$. If there does not exist a path that connects i, j, $d_{\mathcal{G}}(i,j) = \infty$. Furthermore, the diameter $D_{\mathcal{G}}$ of $\mathcal{G}$ is the largest distance between any pair of nodes in $\mathcal{V}$.*

Proposition 2.1. *The distance $d_{\mathcal{G}} : \mathcal{V} \times \mathcal{V} \to \mathbb{I}_{\geq 0} \cup \{\infty\}$ is a metric on $\mathcal{V}$; that is,*

(a) $d_{\mathcal{G}}(i,j) \geq 0$ for any $i,j \in \mathcal{V}$;

(b) $d_{\mathcal{G}}(i,j) = 0$ if and only if $i = j$;

(c) $d_{\mathcal{G}}(i,j) = d_{\mathcal{G}}(j,i)$ for any $i,j \in \mathcal{V}$;

(d) $d_{\mathcal{G}}(i,j) \leq d_{\mathcal{G}}(i,k) + d_{\mathcal{G}}(k,j)$ for any $i,j,k \in \mathcal{V}$.

The proof of this result is straightforward and is thus omitted. We now introduce the concept of graph-structured *matrix bandwidth*.

Definition 2.2 (Graph-Structured Matrix Bandwidth)**.** *Consider a matrix $X \in \mathbb{R}^{m\times n}$, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and index sets $\mathcal{I} = \{I_i\}_{i\in\mathcal{V}}$, $\mathcal{J} = \{J_i\}_{i\in\mathcal{V}}$ that partition*[1] *$\mathbb{I}_{[1,m]}$ and $\mathbb{I}_{[1,n]}$, respectively. Matrix X is said to have bandwidth B induced by an ordered triple $(\mathcal{G}, \mathcal{I}, \mathcal{J})$, if B is the smallest non-negative integer such that $X_{[i][j]} = \mathbf{0}$ for any $i, j \in \mathcal{V}$ with $d_{\mathcal{G}}(i, j) > B$, where $X_{[i][j]} := X[I_i, J_j]$.*

We refer to $X_{[i][j]}$ as the $[i][j]$-block of matrix X. The bandwidth defined above is a generalization of the standard notion of matrix bandwidth [77, Section 1.2.1]. If the matrix $X \in \mathbb{R}^{n\times n}$ is square, $\mathcal{V} = \mathbb{I}_{[1,n]}$, $\mathcal{E} = \{\{i, i+1\}\}_{i=1}^{n-1}$, and $\mathcal{I} = \mathcal{J} = \{\{i\}\}_{i=1}^{n}$, then the graph-structured matrix bandwidth reduces to the standard definition of matrix bandwidth.

Definition 2.2 enables a formal definition of graph-structured matrices; specifically, a graph-structured matrix is a matrix X that has a triple $(\mathcal{G}, \mathcal{I}, \mathcal{J})$ such that the bandwidth B of X, induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$, is much smaller than the diameter $D_{\mathcal{G}}$ of $\mathcal{G}$ (i.e., $B \ll D_{\mathcal{G}}$). This corresponds to the notion of a *block-banded matrix*. Block-diagonal matrices whose blocks are defined by $\mathcal{I}, \mathcal{J}$ (including identity matrices and zero matrices) always have bandwidth of zero (by Proposition 2.1(b)). We now state basic properties of the matrix bandwidth.

Lemma 2.1. *Consider $X \in \mathbb{R}^{m\times n}$ with bandwidth no greater than B_X induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$; we have that:*

(a) $X^\top$ has bandwidth not greater than B_X induced by $(\mathcal{G}, \mathcal{J}, \mathcal{I})$;

(b) if $Y \in \mathbb{R}^{m\times n}$ has bandwidth not greater than B_Y induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$, then $X + Y$ has bandwidth not greater than $\max(B_X, B_Y)$ induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$;

(c) if $W \in \mathbb{R}^{n\times k}$ has bandwidth not greater than B_W induced by $(\mathcal{G}, \mathcal{J}, \mathcal{K})$, then XW has bandwidth not greater than $B_X + B_W$ induced by $(\mathcal{G}, \mathcal{I}, \mathcal{K})$.

Proof of (a). We have that $(X^\top)_{[i][j]} = (X^\top)[J_i][I_j] = (X[I_j][J_i])^\top = (X_{[j][i]})^\top$. From the assumption that X has bandwidth not greater than B_X and Proposition 2.1(c), $(X^\top)_{[i][j]} =$

[1] In this paper, we call a family $\{X_1, \cdots, X_k\}$ of subsets of X to be a partition if $\bigcup_{k=1}^{K} X_k = X$ and $X_1, \cdots, X_k$ are disjoint; here, *we allow X_k to be empty sets*. Note that this differs from the standard definition of a partition, where the nonemptiness of X_k is enforced. This modification allows us to handle nodes with empty variables, constraints, or data.

$\mathbf{0}$ if $d_{\mathcal{G}}(i,j) > B_X$; therefore, X has bandwidth not greater than B_X, and induced by $(\mathcal{G}, \mathcal{J}, \mathcal{I})$. □

Proof of (b). We have that $X_{[i][j]} = \mathbf{0}$ and $Y_{[i][j]} = \mathbf{0}$ if $d_{\mathcal{G}}(i,j) > \max(B_X, B_Y)$, which yields $(X+Y)_{[i][j]} = \mathbf{0}$ if $d_{\mathcal{G}}(i,j) > \max(B_X, B_Y)$. Thus, $X+Y$ has bandwidth not greater than $\max(B_X, B_Y)$, and induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$. □

Proof of (c). If $d_{\mathcal{G}}(i,j) > B_X + B_W$, from Proposition 2.1(d) we have that for any $k \in \mathcal{V}$, $d_{\mathcal{G}}(i,k) > B_X$ or $d_{\mathcal{G}}(j,k) > B_W$. Thus, if $d_{\mathcal{G}}(i,j) > B_X + B_W$, we have $(XW)_{[i][j]} = \sum_{k \in \mathcal{V}} X_{[i][k]} W_{[k][j]} = \mathbf{0}$ (where the first equality comes from the block matrix multiplication law and the second equality comes from observing that $d_{\mathcal{G}}(i,k) > B_X$ or $d_{\mathcal{G}}(j,k) > B_W$). Therefore, XW has bandwidth not greater than $B_X + B_W$, and induced by $(\mathcal{G}, \mathcal{I}, \mathcal{K})$. □

Lemma 2.1 implies that graph-structured properties of a matrix are preserved under transposition, addition, and multiplication (as long as the associated index sets are compatible). Using Lemma 2.1, we can establish properties for the inverse of a graph-structured matrix (this is the main result of this section).

2.1.2 Inverse of Graph-Structured Matrix

Using the basic result in Section 2.1.1, we show the main result of this section. In particular, we analyze the norm of the block $[i][j]$ component of the inverse of a graph-structured matrix (more precisely, a multiplication of the inverse a graph-structured matrix with other graph-structured matrices). We show that such block $[i][j]$ component decays exponentially with respect to $d_{\mathcal{G}}(i,j)$, and the decay becomes faster as the conditioning of the matrix improves.

Theorem 2.1. *Consider a nonsingular matrix $X \in \mathbb{R}^{n \times n}$ with bandwidth not greater than $B_X \geq 1$ induced by $(\mathcal{G}, \mathcal{K}, \mathcal{P})$, $Y \in \mathbb{R}^{n \times m}$ with bandwidth not greater than B_Y induced by $(\mathcal{G}, \mathcal{K}, \mathcal{J})$, and $W \in \mathbb{R}^{\ell \times n}$ with bandwidth not greater than B_W induced by $(\mathcal{G}, \mathcal{I}, \mathcal{P})$; for constants $\overline{\sigma}_X \geq \overline{\sigma}(X)$, $\overline{\sigma}_Y \geq \overline{\sigma}(Y)$, $\overline{\sigma}_W \geq \overline{\sigma}(W)$, and $0 < \underline{\sigma}_X \leq \underline{\sigma}(X)$ (where $\overline{\sigma}(\cdot)$ and $\underline{\sigma}(\cdot)$*

denote the largest and smallest non-trivial singular values of the argument),[2] *the following holds:*

$$\|(WX^{-1}Y)_{[i][j]}\| \leq \frac{\overline{\sigma}_X\overline{\sigma}_Y\overline{\sigma}_W}{\underline{\sigma}_X^2}\left(\frac{\overline{\sigma}_X^2-\underline{\sigma}_X^2}{\overline{\sigma}_X^2+\underline{\sigma}_X^2}\right)^{\left\lceil\frac{d_{\mathcal{G}}(i,j)-B_X-B_Y-B_W}{2B_X}\right\rceil_+}, \quad i,j\in\mathcal{V}, \tag{2.2}$$

where $(WX^{-1}Y)_{[i][j]} := (WX^{-1}Y)[I_i, J_j]$ *and* $\lceil\cdot\rceil_+$ *is the smallest non-negative integer that is greater than or equal to the argument.*

Proof. By definition of singular values and the assumptions on $\overline{\sigma}_X$, $\overline{\sigma}_Y$, $\overline{\sigma}_W$, and $\underline{\sigma}_X$, we have

$$\underline{\sigma}_X^2\boldsymbol{I} \preceq \underline{\sigma}(X)^2\boldsymbol{I} \preceq XX^\top \preceq \overline{\sigma}(X)^2\boldsymbol{I} \preceq \overline{\sigma}_X^2\boldsymbol{I}.$$

From this, one can obtain:

$$\frac{\underline{\sigma}_X^2-\overline{\sigma}_X^2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}\boldsymbol{I} \preceq \boldsymbol{I}-\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top \preceq \frac{-\underline{\sigma}_X^2+\overline{\sigma}_X^2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}\boldsymbol{I}. \tag{2.3}$$

From the nonsingularity of X, $XX^\top$ is nonsingular. This observation implies that:

$$\begin{aligned}
WX^{-1}Y &= \frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}WX^\top\left(\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top\right)^{-1}Y && \text{(2.4a)}\\
&= \frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}WX^\top\left(\boldsymbol{I}-(\boldsymbol{I}-\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top)\right)^{-1}Y && \text{(2.4b)}\\
&= \frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}WX^\top\left(\sum_{q=0}^{\infty}\left(\boldsymbol{I}-\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top\right)^q\right)Y, && \text{(2.4c)}\\
&= \frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}\sum_{q=0}^{\infty}WX^\top\left(\boldsymbol{I}-\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top\right)^qY. && \text{(2.4d)}
\end{aligned}$$

The second equality follows from a simple algebraic manipulation, the third equality follows from [90, Corollary 5.6.16] and (2.3), and the last equality follows from the fact that the series in (2.4d) converges (due to (2.3)). By Lemma 2.1(c), $XX^\top$ has bandwidth not greater than $2B_X$ induced by $(\mathcal{G},\mathcal{K},\mathcal{K})$. In addition, from Lemma 2.1, we see that

$$WX^\top\left(\boldsymbol{I}-\frac{2}{\underline{\sigma}_X^2+\overline{\sigma}_X^2}XX^\top\right)^qY$$

[2]Non-trivial in the sense that we exclude the singular values that are trivially zero due to the nonsquare size of the matrix.

has bandwidth not greater than $(2q+1)B_X + B_Y + B_W$ and induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$. By extracting submatrices defined by the row index I_i and the column index J_j from (2.4), one can obtain:

$$(WX^{-1}Y)_{[i][j]} = \frac{2}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} \sum_{q=q_0(i,j)}^{\infty} \left(WX^\top \left(\boldsymbol{I} - \frac{2XX^\top}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} \right)^q Y \right)_{[i][j]}$$

where

$$q_0(i,j) := \left\lceil \frac{d_{\mathcal{G}}(i,j) - B_X - B_Y - B_W}{2B_X} \right\rceil_+ ;$$

the summation over $q = 0, \cdots, q_0(i,j) - 1$ is zero; because such q satisfy

$$(2q+1)B_X + B_Y + B_W < d_{\mathcal{G}}(i,j).$$

Thus,

$$\left(WX^\top \left(\boldsymbol{I} - \frac{2}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} XX^\top \right)^q Y \right)_{[i][j]} = \boldsymbol{0}.$$

Using the triangle inequality and the fact that the matrix norm of a submatrix is smaller than that of the original matrix,

$$\begin{aligned} \|(WX^{-1}Y)_{[i][j]}\| &\leq \frac{2}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} \sum_{q=q_0(i,j)}^{\infty} \left\| WX^\top \left(\boldsymbol{I} - \frac{2XX^\top}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} \right)^q Y \right\| \\ &\leq \frac{2}{\underline{\sigma}_X^2 + \overline{\sigma}_X^2} \sum_{q=q_0(i,j)}^{\infty} \overline{\sigma}_W \overline{\sigma}_X \left(\frac{\overline{\sigma}_X^2 - \underline{\sigma}_X^2}{\overline{\sigma}_X^2 + \underline{\sigma}_X^2} \right)^q \overline{\sigma}_Y \\ &\leq \frac{\overline{\sigma}_X \overline{\sigma}_Y \overline{\sigma}_W}{\underline{\sigma}_X^2} \left(\frac{\overline{\sigma}_X^2 - \underline{\sigma}_X^2}{\overline{\sigma}_X^2 + \underline{\sigma}_X^2} \right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j) - B_X - B_Y - B_W}{2B_X} \right\rceil_+}. \end{aligned} \tag{2.5}$$

The second inequality follows from the submultiplicativity of the matrix norm and (2.3); and the last inequality follows from the summation of the geometric series. Therefore, (2.2) is obtained. □

The result indicates that the norm of the $[i][j]$-block of $WX^{-1}Y$ *decays exponentially* with respect to the distance between nodes i and j on $\mathcal{G}$; the decay rate becomes faster (smaller) as the condition number $\overline{\sigma}(X)/\underline{\sigma}(X)$ of X decreases; and the decay rate becomes faster as the bandwidths B_X decrease. This property will be key in establishing EDS for the

gsNLP (1.1) and hints at the fact that EDS arises from connectivity induced by the graph (at the linear algebra level).

Related Work: Theorem 2.1 is a generalization of [153, Theorem 1] (which assumes positive definiteness of X and $Y = W = \boldsymbol{I}$). We also note that [47] has studied exponential decay of the components of the inverse of banded matrices. Specifically, in [47, Theorem 2.4], exponential decay for indefinite banded matrices (with the standard definition of bandwidth) is established. Furthermore, in [47, Proposition 5.1], a less general form of Theorem 2.1 is presented; however, graph-structured matrices are not formally introduced, and only positive definite matrices are considered. Theorem 2.1 generalizes these results by introducing the notion of graph-structured matrices and by allowing non-symmetric and indefinite matrices.

2.2 Nonlinear Programming Sensitivity

We now aim to provide an answer to question (Q1). The sketch of our analysis is as follows: we invoke classical results of NLP sensitivity theory [51, 134] to obtain an explicit representation for the one-sided directional derivative of the primal-dual solution mapping with respect to the data; the representation involves the inverse of a graph-structured matrix. The results from the previous section are then applied to this representation to establish bounds on the nodal sensitivity coefficients. Finally, the one-sided directional derivative is integrated over the line segment between a pair of data points that are within the neighborhood of the base data to obtain the result in the form of (2.1). This yields explicit expressions for (Υ, ρ).

To enable compact notation, we introduce the following definitions:

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x};\boldsymbol{p}) &:= \sum_{i\in\mathcal{V}} f_i(\{x_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]};\{p_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]}) \\
\boldsymbol{c}^E(\boldsymbol{x};\boldsymbol{p}) &:= \{c_i^E(\{x_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]};\{p_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]})\}_{i\in\mathcal{V}} \\
\boldsymbol{c}^I(\boldsymbol{x};\boldsymbol{p}) &:= \{c_i^I(\{x_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]};\{p_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]})\}_{i\in\mathcal{V}} \\
\boldsymbol{c}(\boldsymbol{x};\boldsymbol{p}) &:= \{c_i(\{x_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]};\{p_j\}_{j\in\mathcal{N}_{\mathcal{G}}[i]})\}_{i\in\mathcal{V}},
\end{aligned}
$$

$\boldsymbol{n_x} := \sum_{i\in\mathcal{V}} n_{x_i}$, $\boldsymbol{n_{y^E}} := \sum_{i\in\mathcal{V}} n_{y_i^E}$, $\boldsymbol{n_{y^I}} := \sum_{i\in\mathcal{V}} n_{y_i^I}$, $\boldsymbol{n_y} := \sum_{i\in\mathcal{V}} n_{y_i}$, $\boldsymbol{n_z} := \sum_{i\in\mathcal{V}} n_{z_i}$, and $\boldsymbol{n_p} := \sum_{i\in\mathcal{V}} n_{p_i}$. Boldface symbols are used whenever a variable or a function is associated with more than one node. Using these definitions, (1.1) can be expressed as a general parametric NLP of the form:

$$\begin{aligned}\mathcal{P}(\boldsymbol{p}):\ \min_{\boldsymbol{x}}\ & \boldsymbol{f}(\boldsymbol{x};\boldsymbol{p})\\ \text{s.t. }& \boldsymbol{c}^E(\boldsymbol{x};\boldsymbol{p})=\boldsymbol{0},\ (\boldsymbol{y}^E)\\ & \boldsymbol{c}^I(\boldsymbol{x};\boldsymbol{p})\geq\boldsymbol{0},\ \ (\boldsymbol{y}^I).\end{aligned}$$

We denote this problem as $\mathcal{P}(\boldsymbol{p})$; its Lagrange function $\mathcal{L}:\mathbb{R}^{\boldsymbol{n_z}}\times\mathbb{R}^{\boldsymbol{n_p}}\to\mathbb{R}$ is given by

$$\mathcal{L}(\boldsymbol{z};\boldsymbol{p}) := \boldsymbol{f}(\boldsymbol{x};\boldsymbol{p}) - \boldsymbol{y}^\top \boldsymbol{c}(\boldsymbol{x};\boldsymbol{p}).$$

We use $\boldsymbol{z}^\star\in\mathbb{R}^{\boldsymbol{n_z}}$ to denote the primal-dual base solution of $\mathcal{P}(\boldsymbol{p}^\star)$. We denote the primal and dual components of $\boldsymbol{z}^\star=\{z_i^\star\}_{i\in\mathcal{V}}=\{[x_i^\star;y_i^\star]\}_{i\in\mathcal{V}}$ as $\boldsymbol{x}^\star := \{x_i^\star\}_{i\in\mathcal{V}}$ and $\boldsymbol{y}^\star=\{y_i^\star\}_{i\in\mathcal{V}}$, respectively. We say that $\boldsymbol{z}^\star$ is a primal-dual solution if $\boldsymbol{x}^\star$ satisfies the first-order optimality conditions with Lagrange multiplier $\boldsymbol{y}^\star$ (see [124]).

We now make key assumptions that are necessary to establish our main sensitivity result.

Assumption 2.1 (Twice Continuous Differentiability of Functions)**.** *Given the base solution $\boldsymbol{z}^\star$ and data $\boldsymbol{p}^\star$, the functions $\boldsymbol{f}:\mathbb{R}^{\boldsymbol{n_x}}\times\mathbb{R}^{\boldsymbol{n_p}}\to\mathbb{R}$ and $\boldsymbol{c}:\mathbb{R}^{\boldsymbol{n_x}}\times\mathbb{R}^{\boldsymbol{n_p}}\to\mathbb{R}^{\boldsymbol{n_y}}$ are twice continuously differentiable in the neighborhood of $[\boldsymbol{x}^\star;\boldsymbol{p}^\star]$.*

Assumption 2.2 (Strong Second Order Sufficiency)**.** *Given the base solution $\boldsymbol{z}^\star$ and data $\boldsymbol{p}^\star$, the SSOSC for $\mathcal{P}(\boldsymbol{p}^\star)$ are satisfied at $\boldsymbol{z}^\star$.*

Assumption 2.3 (Linear Independence Constraint Qualifications)**.** *Given the base solution $\boldsymbol{z}^\star$ and data $\boldsymbol{p}^\star$, the LICQ for $\mathcal{P}(\boldsymbol{p}^\star)$ are satisfied at $\boldsymbol{z}^\star$.*

We recall that SSOSC requires positive definiteness of the reduced Hessian of the Lagrangian at $\boldsymbol{z}^\star$. The reduced Hessian is the Hessian projected on the null space defined by equality constraints and active inequality constraints with nonzero duals. LICQ requires

that the constraint Jacobian defined by equality and active inequality constraints are linearly independent at $\boldsymbol{z}^\star$. These requirements are stated formally as:

$$ReH(\nabla^2_{\boldsymbol{xx}}\mathcal{L}(\boldsymbol{z}^\star;\boldsymbol{p}^\star), \nabla_{\boldsymbol{x}}\boldsymbol{c}(\boldsymbol{x}^\star;\boldsymbol{p}^\star)[\mathcal{A}^0(\boldsymbol{p}^\star),:]) \succ \mathbf{0} \tag{SSOSC}$$

$$\underline{\sigma}(\nabla_{\boldsymbol{x}}\boldsymbol{c}(\boldsymbol{x}^\star;\boldsymbol{p}^\star)[\mathcal{A}^1(\boldsymbol{p}^\star),:]) > 0. \tag{LICQ}$$

Here, $ReH(H, J) := Z^\top HZ$ is the reduced Hessian, where Z is a null-space matrix of J, and

$$\mathcal{A}^0(\boldsymbol{p}^\star) := \mathcal{A}^E \cup \{i \in \mathcal{A}^I : \boldsymbol{c}(\boldsymbol{x}^\star)[i] = 0,\ \boldsymbol{y}^\star[i] \neq 0\} \tag{2.6}$$

$$\mathcal{A}^1(\boldsymbol{p}^\star) := \mathcal{A}^E \cup \{i \in \mathcal{A}^I : \boldsymbol{c}(\boldsymbol{x}^\star)[i] = 0\}, \tag{2.7}$$

where $\mathcal{A}^E$ and $\mathcal{A}^I$ are the set of equality and inequality constraint indices within $\mathbb{I}_{[1,\boldsymbol{n_y}]}$, respectively.

SSOSC and LICQ are standard assumptions used in NLP sensitivity theory. For instance, Assumption 2.2, 2.3 guarantees strong regularity of the GE representation of the first-order optimality conditions of (1.1) at $\boldsymbol{z}^\star$ [134]. Strong regularity is then used to establish properties for the solution mapping for the parametric NLP $\mathcal{P}(\boldsymbol{p})$. In what follows, we refer to $\boldsymbol{p}^\star$ as the base data and $\boldsymbol{z}^\star$ as the base solution.

Lemma 2.2. *Under Assumption 2.1, 2.2, 2.3, there exist neighborhoods $\mathbb{P} \subseteq \mathbb{R}^{\boldsymbol{n_p}}$ of $\boldsymbol{p}^\star$ and $\mathbb{Z} \subseteq \mathbb{R}^{\boldsymbol{n_z}}$ of $\boldsymbol{z}^\star$ and a continuous function $\boldsymbol{z}^\dagger : \mathbb{P} \to \mathbb{Z}$ such that $\boldsymbol{z}^\dagger(\boldsymbol{p})$ is a primal-dual solution of $\mathcal{P}(\boldsymbol{p})$ that satisfies SSOSC and LICQ. Moreover, for any $\boldsymbol{q} := \{q_i\}_{i\in\mathcal{V}} \in \mathbb{R}^{\boldsymbol{n_p}}$, the one-sided directional derivative of $\boldsymbol{z}^\dagger(\cdot)$ is given by:*

$$D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p}) := \lim_{h \searrow 0} \frac{\boldsymbol{z}^\dagger(\boldsymbol{p} + \boldsymbol{q}h) - \boldsymbol{z}^\dagger(\boldsymbol{p})}{h};$$

with $D_{\boldsymbol{q}}\boldsymbol{x}^\dagger(\boldsymbol{p}) = \boldsymbol{\xi}^\dagger_i(\boldsymbol{p}, \boldsymbol{q})$, $D_{\boldsymbol{q}}\boldsymbol{y}^\dagger(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{\eta}^\dagger(\boldsymbol{p}, \boldsymbol{q})$. We also have that $\boldsymbol{\xi}^\dagger(\boldsymbol{p}, \boldsymbol{q})$ and $\boldsymbol{\eta}^\dagger(\boldsymbol{p}, \boldsymbol{q})[\mathcal{A}^1(\boldsymbol{p})]$ is a primal-dual solution of the quadratic program:

$$\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q}) : \quad \min_{\boldsymbol{\xi}} \frac{1}{2}\boldsymbol{\xi}^\top \boldsymbol{G}(\boldsymbol{p})\boldsymbol{\xi} + \boldsymbol{\xi}^\top \boldsymbol{E}(\boldsymbol{p})\boldsymbol{q}, \tag{2.8a}$$

$$\text{s.t. } \boldsymbol{J}(\boldsymbol{p})[i,:]^\top \boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q} = 0,\ i \in \mathcal{A}^0(\boldsymbol{p}),\ (\boldsymbol{\eta}[i]) \tag{2.8b}$$

$$\boldsymbol{J}(\boldsymbol{p})[i,:]^\top \boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q} \geq 0,\ i \in \mathcal{A}^1(\boldsymbol{p}) \setminus \mathcal{A}^0(\boldsymbol{p}),\ (\boldsymbol{\eta}[i]) \tag{2.8c}$$

and $\boldsymbol{\eta}^\dagger(\boldsymbol{p},\boldsymbol{q})[\mathbb{I}_{[1,\boldsymbol{n_y}]} \setminus \mathcal{A}^1(\boldsymbol{p})] = \boldsymbol{0}$ *(i.e., the free dual variables are fixed to zero), where:*

$$\mathcal{A}^0(\boldsymbol{p}) := \mathcal{A}^E \cup \{i \in \mathcal{A}^I : \boldsymbol{c}(\boldsymbol{x}^\dagger(\boldsymbol{p}))[i] = 0,\ \boldsymbol{y}^\dagger(\boldsymbol{p})[i] \neq 0\} \tag{2.9a}$$

$$\mathcal{A}^1(\boldsymbol{p}) := \mathcal{A}^E \cup \{i \in \mathcal{A}^I : \boldsymbol{c}(\boldsymbol{x}^\dagger(\boldsymbol{p}))[i] = 0\} \tag{2.9b}$$

$$\boldsymbol{G}(\boldsymbol{p}) := \nabla_{\boldsymbol{xx}}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}) \tag{2.9c}$$

$$\boldsymbol{E}(\boldsymbol{p}) := \nabla_{\boldsymbol{xp}}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}) \tag{2.9d}$$

$$\boldsymbol{J}(\boldsymbol{p}) := \nabla_{\boldsymbol{x}}\boldsymbol{c}(\boldsymbol{x}^\dagger(\boldsymbol{p});\boldsymbol{p}) \tag{2.9e}$$

$$\boldsymbol{K}(\boldsymbol{p}) := \nabla_{\boldsymbol{p}}\boldsymbol{c}(\boldsymbol{x}^\dagger(\boldsymbol{p});\boldsymbol{p}). \tag{2.9f}$$

Moreover, a unique primal-dual solution of $\mathcal{QP}(\boldsymbol{p},\boldsymbol{q})$ *exists for any* $\boldsymbol{p} \in \mathbb{P}$ *and* $\boldsymbol{q} \in \mathbb{R}^{\boldsymbol{n_p}}$*; thus,* $\boldsymbol{\xi}^\dagger(\cdot,\cdot)$ *and* $\boldsymbol{\eta}^\dagger(\cdot,\cdot)$ *are well-defined.*

Note that the results in Lemma 2.2 are well-known. We provide the proof below to point the readers to the relevant results in the literature.

Proof. The results in [51, Theorem 2G.8] ensure semidifferentiability (which guarantees continuity) of the solution of the first-order optimality conditions for $\mathcal{P}(\boldsymbol{p})$ over a certain neighborhood of $\boldsymbol{p}^\star$. This result is established by using the GE representation of the first-order conditions. Furthermore, [51, Theorem 2G.9] establishes that over a certain neighborhood $\mathbb{P}$, the solution mapping of the GE satisfies SSOSC and LICQ; that is, within $\mathbb{P}$, the solution mapping for the GE is the solution mapping for $\mathcal{P}(\boldsymbol{p})$ at which SSOSC and LICQ are satisfied. Moreover, by [51, Theorem 2G.8], the one-sided directional derivative of the solution mapping for the GE (which exists for any direction $\boldsymbol{q} \in \mathbb{R}^{\boldsymbol{n_p}}$ by semidifferentiability [51, Theorem 2D.1]) can be evaluated by using the linearized GE. The linear GE are the first-order optimality conditions for $\mathcal{QP}(\boldsymbol{p},\boldsymbol{q})$ (see [51, Equation (35)]); here, the first-order conditions are necessary and sufficient conditions for the optimality due to the convexity $\mathcal{QP}(\boldsymbol{p},\boldsymbol{q})$ (guaranteed by SSOSC and LICQ of $\boldsymbol{z}^\star$ for the original problem). As such, the one-sided directional derivative of the solution mapping for $\mathcal{P}(\boldsymbol{p})$ can be evaluated by solving $\mathcal{QP}(\boldsymbol{p},\boldsymbol{q})$. Finally, the strong regularity of the GE at $\boldsymbol{z}^\star$ (obtained under SSOSC and

LICQ) guarantees that there exists a unique solution of the linearized GE [134], which in turn guarantees the existence of a unique solution of $\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q})$. □

Under Lemma 2.2, the rate of change $D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})$ of the primal-dual solution of $\mathcal{P}(\boldsymbol{p})$ (for a given direction $\boldsymbol{q}$) can be quantified by using the solution of $\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q})$. For given $\boldsymbol{p}$ and $\boldsymbol{q}$, the parameters in $\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q})$ can be evaluated explicitly and thus its solution can be calculated; as such, Lemma 2.2 provides a computational procedure to evaluate primal-dual solution sensitivity.

The quadratic program $\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q})$ plays a central role in our analysis and we thus examine its properties in more detail. The first-order conditions of this problem are:

$$\boldsymbol{G}(\boldsymbol{p})\boldsymbol{\xi} + \boldsymbol{E}(\boldsymbol{p})\boldsymbol{q} + \boldsymbol{J}^\top\boldsymbol{\eta} = \boldsymbol{0} \tag{2.10a}$$

$$\boldsymbol{J}(\boldsymbol{p})[i,:]\boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q} = 0,\ i \in \mathcal{A}^0(\boldsymbol{p}) \tag{2.10b}$$

$$\boldsymbol{J}(\boldsymbol{p})[i,:]\boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q} \geq 0,\ i \in \mathcal{A}^1(\boldsymbol{p}) \setminus \mathcal{A}^0(\boldsymbol{p}) \tag{2.10c}$$

$$\boldsymbol{\eta}[i] \geq 0,\ i \in \mathcal{A}^1(\boldsymbol{p}) \setminus \mathcal{A}^0(\boldsymbol{p}) \tag{2.10d}$$

$$(\boldsymbol{J}(\boldsymbol{p})[i,:]\boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q})\,\boldsymbol{\eta}[i] = 0,\ i \in \mathcal{A}^1(\boldsymbol{p}) \setminus \mathcal{A}^0(\boldsymbol{p}). \tag{2.10e}$$

Under SSOSC and LICQ for $\mathcal{P}(\boldsymbol{p}^\star)$ at $\boldsymbol{z}^\star$, these conditions are necessary and sufficient for any solution of $\mathcal{QP}(\boldsymbol{p}, \boldsymbol{q})$. From the complementarity condition (2.10e), one can observe that there exists $\mathcal{A}^0(\boldsymbol{p}) \subseteq \mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}) \subseteq \mathcal{A}^1(\boldsymbol{p})$ such that:

$$\boldsymbol{J}(\boldsymbol{p})[i,:]\boldsymbol{\xi} + \boldsymbol{K}(\boldsymbol{p})[i,:]\boldsymbol{q} = 0,\ i \in \mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}) \tag{2.11a}$$

$$\boldsymbol{\eta}[i] = 0,\ i \in \mathcal{A}^1(\boldsymbol{p}) \setminus \mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}). \tag{2.11b}$$

are satisfied at $\boldsymbol{\xi}^\dagger(\boldsymbol{p}, \boldsymbol{q})$, $\boldsymbol{\eta}^\dagger(\boldsymbol{p}, \boldsymbol{q})$. Thus, from (A.8), (2.11), and $\boldsymbol{\eta}[i] = 0$ for $i \in \mathbb{I}_{[1,n_y]} \setminus \mathcal{A}^1(\boldsymbol{p})$ (from Lemma 2.2), we have:

$$\begin{bmatrix} \boldsymbol{G}(\boldsymbol{p}) & \boldsymbol{J}(\boldsymbol{p})[\mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}), :]^\top \\ \boldsymbol{J}(\boldsymbol{p})[\mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}), :] & \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta}[\mathcal{A}'(\boldsymbol{p}, \boldsymbol{q})] \end{bmatrix} = - \begin{bmatrix} \boldsymbol{E}(\boldsymbol{p}) \\ \boldsymbol{K}(\boldsymbol{p})[\mathcal{A}'(\boldsymbol{p}, \boldsymbol{q}), :] \end{bmatrix} \boldsymbol{q} \tag{2.12}$$

$$\boldsymbol{\eta}[\mathbb{I}_{[1,n_y]} \setminus \mathcal{A}'(\boldsymbol{p}, \boldsymbol{q})] = \boldsymbol{0}.$$

The linear equation (2.12) provides a relationship between $[\boldsymbol{\xi}^\dagger(\boldsymbol{p},\boldsymbol{q});\boldsymbol{\eta}^\dagger(\boldsymbol{p},\boldsymbol{q})]$ and $\boldsymbol{q}$; however, it does not provide an explicit relationship because $\mathcal{A}'(\boldsymbol{p},\boldsymbol{q})$ depends on $\boldsymbol{q}$.

By rearranging $[\boldsymbol{\xi}^\dagger(\boldsymbol{p},\boldsymbol{q});\boldsymbol{\eta}^\dagger(\boldsymbol{p},\boldsymbol{q})]$ one can obtain $\boldsymbol{\zeta}^\dagger(\boldsymbol{p}) = \{[\xi_i^\dagger(\boldsymbol{p},\boldsymbol{q});\eta_i^\dagger(\boldsymbol{p},\boldsymbol{q})]\}_{i\in\mathcal{V}}$, where $\boldsymbol{\xi}^\dagger(\boldsymbol{p},\boldsymbol{q}) = \{\xi_i^\dagger(\boldsymbol{p},\boldsymbol{q})\}_{i\in\mathcal{V}}$, $\boldsymbol{\eta}^\dagger(\boldsymbol{p},\boldsymbol{q}) = \{\eta_i^\dagger(\boldsymbol{p},\boldsymbol{q})\}_{i\in\mathcal{V}}$. To perform such rearrangement, we consider a permutation $\phi : \mathbb{I}_{[1,\boldsymbol{n}]} \to \mathbb{I}_{[1,\boldsymbol{n}]}$ that achieves $\boldsymbol{z}[\phi(i)] = [\boldsymbol{\xi};\boldsymbol{\eta}][i]$. This permutation enables the following definition:

$$\mathcal{B}^0(\boldsymbol{p}) := \phi(\mathbb{I}_{[1,\boldsymbol{n}_{\boldsymbol{x}}]} \cup (\mathcal{A}^0(\boldsymbol{p}) + \boldsymbol{n}_{\boldsymbol{x}})) \tag{2.13a}$$

$$\mathcal{B}^1(\boldsymbol{p}) := \phi(\mathbb{I}_{[1,\boldsymbol{n}_{\boldsymbol{x}}]} \cup (\mathcal{A}^1(\boldsymbol{p}) + \boldsymbol{n}_{\boldsymbol{x}})) \tag{2.13b}$$

$$\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}) := \phi(\mathbb{I}_{[1,\boldsymbol{n}_{\boldsymbol{x}}]} \cup (\mathcal{A}'(\boldsymbol{p},\boldsymbol{q}) + \boldsymbol{n}_{\boldsymbol{x}})). \tag{2.13c}$$

Finally, $[\boldsymbol{\xi};\boldsymbol{\eta}]$ can be rearranged in such a way that the relationship $D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p}) = \boldsymbol{\zeta}^\dagger(\boldsymbol{p})$ (from Lemma 2.2) can be used; this yields:

$$D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})] = -\left(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]\right)^{-1}\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),:]\boldsymbol{q}, \tag{2.14a}$$

$$D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[\mathbb{I}_{[1,\boldsymbol{n}]} \setminus \mathcal{B}'(\boldsymbol{p},\boldsymbol{q})] = \boldsymbol{0}, \tag{2.14b}$$

where:

$$\boldsymbol{H}(\boldsymbol{p}) := \nabla_{\boldsymbol{z}\boldsymbol{z}}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}) \tag{2.15a}$$

$$\boldsymbol{F}(\boldsymbol{p}) := \nabla_{\boldsymbol{z}\boldsymbol{p}}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}). \tag{2.15b}$$

Here, the nonsingularity of $\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]$ is guaranteed by the fact that $\boldsymbol{G}(\boldsymbol{p})$ is positive definite on the null-space of $\boldsymbol{J}(\boldsymbol{p})[\mathcal{A}^0(\boldsymbol{p}),:]$. This follows from the satisfaction of LICQ and SSOSC (from Lemma 2.2) and [124, Lemma 16.1].

2.3 Nodal Sensitivity Result

We now are ready to establish the nodal sensitivity result.

Related Work: The nodal solution sensitivity was studied in a number of previous works in the literature. The nodal sensitivity of dynamic programs has been studied in [119,165,166]. In particular, the work of [165] first established the decay of stage-wide solution sensitivity

under inequality-constrained linear-quadratic control setting, and [166] subsequently generalized the inequality constraint setting. The study of sensitivity in the nonlinear dynamic program was established in [119]. The sensitivity of graph-structured quadratic programs has been studied in [145]. Finally, the nodal sensitivity result for the gsNLPs is established in [146] (the basis of this chapter).

First, we observe that $\boldsymbol{H}(\boldsymbol{p})$ and $\boldsymbol{F}(\boldsymbol{p})$ are graph-structured matrices; these have bandwidth not greater than two, induced by $(\mathcal{G},\mathcal{I},\mathcal{I})$ and $(\mathcal{G},\mathcal{I},\mathcal{K})$, and where for $i \in \mathcal{V}$,

$$I_i := \mathbb{I}_{\sum_{j\in\mathcal{V},j<i} n_{p_j}+[1,n_{z_i}]} \tag{2.16}$$

$$K_i := \mathbb{I}_{\sum_{j\in\mathcal{V},j<i} n_{p_j}+[1,n_{p_i}]}. \tag{2.17}$$

Note that $\mathcal{I} := \{I_i\}_{i\in\mathcal{V}}$ and $\mathcal{K} := \{K_i\}_{i\in\mathcal{V}}$ partition $\mathbb{I}_{[1,\boldsymbol{n_z}]}$ and $\mathbb{I}_{[1,\boldsymbol{n_p}]}$, respectively. We now observe that:

$$H_{ij}(\boldsymbol{p}) := \nabla^2_{z_i z_j}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}) = (\boldsymbol{H}(\boldsymbol{p}))_{[i][j]} \tag{2.18a}$$

$$F_{ij}(\boldsymbol{p}) := \nabla^2_{z_i p_j}\mathcal{L}(\boldsymbol{z}^\dagger(\boldsymbol{p});\boldsymbol{p}) = (\boldsymbol{F}(\boldsymbol{p}))_{[i][j]}. \tag{2.18b}$$

From this we can see see that, if $d_{\mathcal{G}}(i,j) > 2$ holds, then $(\boldsymbol{H}(\boldsymbol{p}))_{[i][j]} = \boldsymbol{0}$ and $(\boldsymbol{F}(\boldsymbol{p}))_{[i][j]} = \boldsymbol{0}$ hold. The reason that the bandwidth may be greater than one is as follows. Suppose that $\mathcal{V} = \{1,2,3\}$ and $\mathcal{E} = \{\{1,2\},\{2,3\}\}$; then the constraints at node 2 can be coupled with variables/data at node 1 and 3. This produces potential non-zero coupling between variables/data at node 1 and node 3 (i.e., $H_{13} \neq \boldsymbol{0}$); here, $d_{\mathcal{G}}(i,j) = 2$. Likewise, the variables that are within distance 2 can have potential coupling.

The submatrices of $\boldsymbol{H}(\boldsymbol{p})$ and $\boldsymbol{F}(\boldsymbol{p})$ are also graph-structured (induced by properly chosen index sets). In particular, $\boldsymbol{H}(\boldsymbol{p})[\mathcal{B},\mathcal{B}]$ and $\boldsymbol{F}[\mathcal{B},:]$ with $\mathcal{B} \subseteq \mathbb{I}_{[1,\boldsymbol{n}]}$ have bandwidth not greater than two induced by $(\mathcal{G},\mathcal{I}^{\mathcal{B}},\mathcal{I}^{\mathcal{B}})$ and $(\mathcal{G},\mathcal{I}^{\mathcal{B}},\mathcal{K})$, where $\mathcal{I}^{\mathcal{B}} := \{I_i \cap \mathcal{B}\}_{i\in\mathcal{V}}$.

We thus see that (2.12) involves the inverse of a graph-structured matrix; as such, Theorem 2.1 can be used for establishing the desired sensitivity bounds. By combining Theorem 2.1 and Lemma 2.2, one can establish the following result.

Lemma 2.3. *Suppose Assumption 2.1, 2.2, 2.3 hold, and suppose that, for given* $\boldsymbol{p} \in \mathbb{P}$ *(defined in Lemma 2.2) and* $\boldsymbol{q} := \{q_i\}_{i\in\mathcal{V}} \in \mathbb{R}^{\boldsymbol{n_p}}$*, we have* $\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})$*,* $\overline{\sigma}_{\boldsymbol{F}}(\boldsymbol{p},\boldsymbol{q})$*, and* $\underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})$ *such that:*

$$\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q}) \geq \overline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]) \tag{2.19a}$$

$$\overline{\sigma}_{\boldsymbol{F}}(\boldsymbol{p},\boldsymbol{q}) \geq \overline{\sigma}(\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),:]) \tag{2.19b}$$

$$0 < \underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q}) \leq \underline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}),\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]), \tag{2.19c}$$

where $\mathcal{B}'$ *is defined in (2.13c); then the following holds for any* $i \in \mathcal{V}$*:*

$$\|D_{\boldsymbol{q}} z_i^\dagger(\boldsymbol{p},\boldsymbol{q})\| \leq \sum_{j\in\mathcal{V}} \frac{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})\overline{\sigma}_{\boldsymbol{F}}(\boldsymbol{p},\boldsymbol{q})}{\underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2} \left(\frac{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2 - \underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2}{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2 + \underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2}\right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1 \right\rceil_+} \|q_j\|. \tag{2.20}$$

Proof. For simplicity, we denote $\mathcal{B}'(\boldsymbol{p},\boldsymbol{q})$ as $\mathcal{B}'$. From the fact that $\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}',\mathcal{B}']$ is always nonsingular (as discussed after (2.15)) we have that $\underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})$ satisfying (2.19c) always exists. By inspecting the block structure of (2.14) we can see that:

$$D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[I_i \cap \mathcal{B}'] = \sum_{j\in\mathcal{V}} -((\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}',\mathcal{B}'])^{-1}\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}',:])_{[i][j]} q_j, \tag{2.21}$$

where $\mathcal{I}^{\mathcal{B}'} := \{I_i \cap \mathcal{B}'\}_{i\in\mathcal{V}}$ and $\mathcal{K} := \{K_i\}_{i\in\mathcal{V}}$ (defined in (2.16)) are used for index sets. We have already established that $\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}',\mathcal{B}']$ has bandwidth not greater than two, induced by $(\mathcal{G},\mathcal{I}^{\mathcal{B}'},\mathcal{I}^{\mathcal{B}'})$ and that $\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}',:]$ has bandwidth not greater than two, induced by $(\mathcal{G},\mathcal{I}^{\mathcal{B}'},\mathcal{K})$. By applying Theorem 2.1, we obtain:

$$\|((\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}',\mathcal{B}'])^{-1}\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}',:])_{[i][j]}\| \leq \frac{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})\overline{\sigma}_{\boldsymbol{F}}(\boldsymbol{p},\boldsymbol{q})}{\underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2} \left(\frac{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2 - \underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2}{\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2 + \underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p},\boldsymbol{q})^2}\right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1 \right\rceil_+}. \tag{2.22}$$

Now note that $\|D_{\boldsymbol{q}} z_i^\dagger(\boldsymbol{p})\| \leq \|D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[I_i \cap \mathcal{B}']\| + \|D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[I_i \setminus \mathcal{B}']\|$, and recall from (2.14b) that $D_{\boldsymbol{q}}\boldsymbol{z}^\dagger(\boldsymbol{p})[I_i \setminus \mathcal{B}'] = \boldsymbol{0}$. Hence, by applying (2.22) to (2.21), and applying triangle inequality, we obtain (2.20). □

Lemma 2.3 establishes that the dependence of $\|D_{\boldsymbol{q}} z_i^\dagger(\boldsymbol{p})\|$ on perturbation p_j decays with $d_{\mathcal{G}}(i,j)$. However, the right-hand side of (2.22) still depends on $\boldsymbol{p},\boldsymbol{q}$ and this complicates

the use of Lemma 2.3 (needed to quantify sensitivity behavior). To express this result as in (2.1), wherein the expressions on the right-hand side are independent of $\boldsymbol{p}, \boldsymbol{q}$, we exploit the continuity of singular values [77, Corollary 8.6.2]. This gives us the main result of this chapter.

Theorem 2.2 (Exponential Decay of Sensitivity (EDS))**.** *Under Assumption 2.1, 2.2, 2.3, for given* $\epsilon > 0$, $\overline{\sigma}_{\boldsymbol{H}} \geq \overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)$, $\overline{\sigma}_{\boldsymbol{F}} \geq \overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p}^\star)$, *and* $0 < \underline{\sigma}_{\boldsymbol{H}} \leq \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)$, *where*

$$\begin{aligned}
\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}) &:= \max\{\overline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B},\mathcal{B}]) : \mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)\} \\
\overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p}) &:= \max\{\overline{\sigma}(\boldsymbol{F}(\boldsymbol{p})[\mathcal{B},:]) : \mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)\} \\
\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}) &:= \min\{\underline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B},\mathcal{B}]) : \mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)\},
\end{aligned}$$

there exists a neighborhood $\mathbb{P}_\epsilon$ *of* $\boldsymbol{p}^\star$ *such that the following holds for any* $\boldsymbol{p}, \boldsymbol{p}' \in \mathbb{P}_\epsilon$*:*

$$\|z_i^\dagger(\boldsymbol{p}) - z_i^\dagger(\boldsymbol{p}')\| \leq \sum_{j \in \mathcal{V}} \Upsilon \rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1 \right\rceil_+} \|p_j - p_j'\|, \quad i \in \mathcal{V}, \tag{2.23}$$

with $\Upsilon := \dfrac{\overline{\sigma}_{\boldsymbol{H}}\overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}_{\boldsymbol{H}}^2} + \epsilon$ *and* $\rho := \dfrac{\overline{\sigma}_{\boldsymbol{H}}^2 - \underline{\sigma}_{\boldsymbol{H}}^2}{\overline{\sigma}_{\boldsymbol{H}}^2 + \underline{\sigma}_{\boldsymbol{H}}^2} + \epsilon$.

Proof. From the continuity of $\boldsymbol{z}^\dagger(\cdot)$ in the neighborhood of $\boldsymbol{p}^\star$ and the continuity of $\boldsymbol{c}(\cdot,\cdot)$ in the neighborhood of $[\boldsymbol{z}^\star; \boldsymbol{p}^\star]$, there exists a neighborhood $\widetilde{\mathbb{P}} \subseteq \mathbb{P}$ of $\boldsymbol{p}^\star$ such that, for $\boldsymbol{p} \in \widetilde{\mathbb{P}}$ and $i \in \mathbb{I}_{[1,\boldsymbol{n_y}]}$,

$$\begin{aligned}
\boldsymbol{c}(\boldsymbol{x}^\star)[i] > 0 \;&\Rightarrow\; \boldsymbol{c}(\boldsymbol{x}^\dagger(\boldsymbol{p}))[i] > 0, \\
\boldsymbol{y}^\star[i] \neq 0 \;&\Rightarrow\; \boldsymbol{y}^\dagger(\boldsymbol{p})[i] \neq 0.
\end{aligned}$$

These conditions and complementarity slackness imply that for $\boldsymbol{p} \in \widetilde{\mathbb{P}}$, we have $\mathcal{A}^0(\boldsymbol{p}^\star) \subseteq \mathcal{A}^0(\boldsymbol{p})$ and $\mathcal{A}^1(\boldsymbol{p}) \subseteq \mathcal{A}^1(\boldsymbol{p}^\star)$; that is, $\mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B}^0(\boldsymbol{p})$ and $\mathcal{B}^1(\boldsymbol{p}) \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)$. From this result and the fact that $\mathcal{B}^0(\boldsymbol{p}) \subseteq \mathcal{B}'(\boldsymbol{p},\boldsymbol{q}) \subseteq \mathcal{B}^1(\boldsymbol{p})$, we have that:

$$\overline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}), \mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]) \leq \overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}) \tag{2.24a}$$

$$\overline{\sigma}(\boldsymbol{F}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}), :]) \leq \overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p}) \tag{2.24b}$$

$$\underline{\sigma}(\boldsymbol{H}(\boldsymbol{p})[\mathcal{B}'(\boldsymbol{p},\boldsymbol{q}), \mathcal{B}'(\boldsymbol{p},\boldsymbol{q})]) \geq \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}). \tag{2.24c}$$

By the twice-continuous differentiability of $\mathcal{L}(\cdot,\cdot)$ and the continuity of $\boldsymbol{z}^\dagger(\cdot)$, we have that $\boldsymbol{H}(\cdot)$ is continuous. The same holds true for its submatrices: $\boldsymbol{H}(\cdot)[\mathcal{B},\mathcal{B}]$ with $\mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)$. From the continuity of singular values with respect to its entries [77, Corollary 8.6.2], we have that $\overline{\sigma}(\boldsymbol{H}(\cdot)[\mathcal{B},\mathcal{B}])$ and $\underline{\sigma}(\boldsymbol{H}(\cdot)[\mathcal{B},\mathcal{B}])$ are continuous for any $\mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)$; accordingly, since a maximum and a minimum of a fixed and finite number of continuous functions is continuous, we have that $\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})$, $\overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p})$, $\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})$ are continuous with respect to $\boldsymbol{p}$ in $\widetilde{\mathbb{P}}$. Thus, there exists a convex neighborhood $\mathbb{P}_\epsilon \subseteq \widetilde{\mathbb{P}}$ of $\boldsymbol{p}^\star$ wherein the following are satisfied:

$$\frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})\overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p})}{\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})^2} \leq \frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)\overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p}^\star)}{\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2} + \epsilon \tag{2.25a}$$

$$\frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})^2 - \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})^2}{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})^2 + \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p})^2} \leq \frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2 - \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2}{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2 + \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2} + \epsilon. \tag{2.25b}$$

Here, note that $\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star) > 0$ because $\underline{\sigma}(\boldsymbol{H}[\mathcal{B},\mathcal{B}]) > 0$ holds for any $\mathcal{B}^0(\boldsymbol{p}^\star) \subseteq \mathcal{B} \subseteq \mathcal{B}^1(\boldsymbol{p}^\star)$ (as discussed in the proof of Lemma 2.3). By applying (2.24) and (2.25) to Lemma 2.3, we obtain:

$$\begin{aligned}
\|D_{\boldsymbol{q}} z_i^\dagger(\boldsymbol{p})\| &\leq \sum_{j\in\mathcal{V}} \left(\frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)\overline{\overline{\sigma}}_{\boldsymbol{F}}(\boldsymbol{p}^\star)}{\underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2} + \epsilon\right)\left(\frac{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2 - \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2}{\overline{\overline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2 + \underline{\underline{\sigma}}_{\boldsymbol{H}}(\boldsymbol{p}^\star)^2} + \epsilon\right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1\right\rceil_+} \|q_j\| \\
&\leq \sum_{j\in\mathcal{V}} \left(\frac{\overline{\sigma}_{\boldsymbol{H}}\overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}_{\boldsymbol{H}}^2} + \epsilon\right)\left(\frac{\overline{\sigma}_{\boldsymbol{H}}^2 - \underline{\sigma}_{\boldsymbol{H}}^2}{\overline{\sigma}_{\boldsymbol{H}}^2 + \underline{\sigma}_{\boldsymbol{H}}^2} + \epsilon\right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1\right\rceil_+} \|q_j\| \\
&\leq \sum_{j\in\mathcal{V}} \Upsilon \rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1\right\rceil_+} \|q_j\|,
\end{aligned} \tag{2.26}$$

for any $\boldsymbol{p} \in \mathbb{P}_\epsilon$ and $\boldsymbol{q} \in \mathbb{R}^{\boldsymbol{n_p}}$. Finally, since we have chosen $\mathbb{P}_\epsilon$ to be convex, for any $\boldsymbol{p}, \boldsymbol{p}' \in \mathbb{P}_\epsilon$, the line segment between $\boldsymbol{p}, \boldsymbol{p}'$ is within $\mathbb{P}_\epsilon$. Thus, we have:

$$\begin{aligned}
\|z_i^\dagger(\boldsymbol{p}) - z_i^\dagger(\boldsymbol{p}')\| &\leq \left\| \int_0^1 D_{\boldsymbol{p}'-\boldsymbol{p}} z_i^\dagger((1-t)\boldsymbol{p} + t\boldsymbol{p}')dt \right\| \\
&\leq \int_0^1 \|D_{\boldsymbol{p}'-\boldsymbol{p}} z_i^\dagger((1-t)\boldsymbol{p} + t\boldsymbol{p}')\| dt \\
&\leq \sum_{j\in\mathcal{V}} \Upsilon \rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1\right\rceil_+} \|p_j - p_j'\|,
\end{aligned}$$

where the first inequality is from Newton-Leibniz, the second inequality follows from triangle inequality for integrals, and the last inequality follows from (2.26). □

Theorem 2.2 establishes the sensitivity bounds $\{C_{ij} = \Upsilon\rho^{\lceil d_{\mathcal{G}}(i,j)/4-1\rceil_+}\}_{i,j\in\mathcal{V}}$ that appeared in Question (Q1). One can observe that $\Upsilon > 0$ and $\rho \in (0,1)$ hold; consequently, we have that the upper bound of the nodal sensitivity decays exponentially as $d_{\mathcal{G}}(i,j)$ increases. We can also see that (Υ, ρ) depend on the singular values of the submatrices of $\boldsymbol{H}(\boldsymbol{p}^\star)$, $\boldsymbol{F}(\boldsymbol{p}^\star)$, which are submatrices of the full Hessian matrix $\nabla^2\mathcal{L}(\boldsymbol{z}^\star;\boldsymbol{p}^\star)$. Therefore, the singular values of the submatrices of $\boldsymbol{H}(\boldsymbol{p}^\star)$ and $\boldsymbol{F}(\boldsymbol{p}^\star)$ play important roles in sensitivity behavior.

Remark 2.3. *One can establish EDS for a more general version of gsNLPs, in which coupling is allowed within the expanded neighborhood:*

$$\mathcal{N}_{\mathcal{G}}^{B}[i] := \{j \in \mathcal{V} : d_{\mathcal{G}}(i,j) \le B\}, \tag{2.27}$$

with $B > 1$. Such an NLP arises when algebraic coupling between nodes extends beyond immediate neighbors. In such a case, the matrices $\boldsymbol{H}(\boldsymbol{p})$ and $\boldsymbol{F}(\boldsymbol{p})$ (and their submatrices) have bandwidths not greater than $2B$. For this more general setting, the corresponding results for Theorem 2.2 can be established; in particular, if the rest of the assumptions in Theorem 2.2 remain the same, the following holds:

$$\|z_i^\dagger(\boldsymbol{p}) - z_i^\dagger(\boldsymbol{p}')\| \le \sum_{j\in\mathcal{V}} \Upsilon\rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4B}-1\right\rceil_+}\|p_j - p_j'\|. \tag{2.28}$$

We can observe that the exponential decay rate increases (the decay becomes slower) as the constant B increases. This implies that we require a small coupling radius B in order to have fast decay of sensitivity (which makes intuitive sense).

Chapter 3

Uniform Regularity Conditions

An interesting class of gsNLPs is that in which the underlying graph is a subgraph of an infinite-dimensional graph. Examples include time-dependent problems (in which we might want to extend the horizon) and discretized PDE optimization (in which we might want to expand the domain). To analyze this setting, we consider a family of problems $\{\mathcal{P}_{(k)}(\cdot)\}_{k\in\mathcal{K}}$ with a potentially infinite problem index set $\mathcal{K}$. The associated quantities are introduced accordingly; $\{\mathcal{G}_{(k)} = (\mathcal{V}_{(k)}, \mathcal{E}_{(k)})\}_{k\in\mathcal{K}}$, $\{\boldsymbol{f}_{(k)}(\cdot)\}_{k\in\mathcal{K}}$, $\{\boldsymbol{c}_{(k)}(\cdot)\}_{k\in\mathcal{K}}$. Also, a set of data $\{\boldsymbol{p}^\star_{(k)}\}_{k\in\mathcal{K}}$ and the associated base solutions $\{\boldsymbol{z}^\star_{(k)}\}_{k\in\mathcal{K}}$ are considered. The submatrices $\{\boldsymbol{H}_{(k)}(\boldsymbol{p}^\star)\}_{k\in\mathcal{K}}$, $\{\boldsymbol{F}_{(k)}(\boldsymbol{p}^\star)\}_{k\in\mathcal{K}}$ of the full Hessian matrix can be defined as in (2.15) for each k.

This chapter aims to establish sufficient conditions for:

$$\sup_{k\in\mathcal{K}} \overline{\overline{\sigma}}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)}) < +\infty; \tag{3.1a}$$

$$\sup_{k\in\mathcal{K}} \overline{\overline{\sigma}}_{\boldsymbol{F},(k)}(\boldsymbol{p}^\star_{(k)}) < +\infty; \tag{3.1b}$$

$$\inf_{k\in\mathcal{K}} \underline{\sigma}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)}) > 0, \tag{3.1c}$$

where $\overline{\overline{\sigma}}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)})$, $\overline{\overline{\sigma}}_{\boldsymbol{F},(k)}(\boldsymbol{p}^\star_{(k)})$, and $\underline{\sigma}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)})$ are defined in Theorem 2.2, but the problem index k is added. One can observe that, if (3.1) is violated, $\Upsilon_{(k)}$ may become indefinitely large and $\rho_{(k)}$ may approach one (thus making the bounds derived in Theorem 2.2 not particularly useful). Hence, ensuring (3.1) is crucial for guaranteeing a moderately bounded base sensitivity magnitude $\Upsilon_{(k)}$ and a fast sensitivity decay rate $\rho_{(k)}$.

We call (3.1) *uniform* boundedness conditions; in addition, we call a quantity to be *uniform* in k if the quantity is independent of the index k. Note that (3.1) holds trivially if

$\mathcal{K}$ is finite and Assumption 2.1, 2.2, 2.3 hold for each $k \in \mathcal{K}$. However, even if $\mathcal{K}$ is finite, it is necessary for Theorem 2.2 to be practically useful that $\inf_{k\in\mathcal{K}} \underline{\underline{\sigma}}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)})$ is sufficiently bounded away from zero and that $\sup_{k\in\mathcal{K}} \overline{\overline{\sigma}}_{\boldsymbol{H},(k)}(\boldsymbol{p}^\star_{(k)})$ and $\sup_{k\in\mathcal{K}} \overline{\overline{\sigma}}_{\boldsymbol{F},(k)}(\boldsymbol{p}^\star_{(k)})$ are bounded above by a moderately large number. As such, the results in this section provide useful information even if $\mathcal{K}$ is finite (and even if $\mathcal{K}$ is a singleton). Hereafter, we will drop the notation for the dependency on k as well as $\boldsymbol{p}$ (e.g., $\boldsymbol{H} \leftarrow \boldsymbol{H}_{(k)}(\boldsymbol{p}^\star)$). This is because (i) even if we assume that there might be multiple problem of interest, we study the condition that applies uniformly to each problem; thus, we do not need to specify the problem index k each time; and (ii) $\boldsymbol{p}$ is fixed to $\boldsymbol{p}^\star$ for the rest of the discussion in this section.

3.1 Uniform Regularity Conditions

We now state key *uniform regularity assumptions* that enable uniform boundedness (3.1). These assumptions provide basic uniform parameters from which we can establish explicit bounds for the quantities in (3.1).

Assumption 3.4 (L-Uniformly Bounded Lagrangian Hessian)**.** *There exists $L \geq 0$ such that the following holds:*

$$\|\nabla^2_{\boldsymbol{zz}}\mathcal{L}(\boldsymbol{z}^\star;\boldsymbol{p}^\star)\|, \|\nabla^2_{\boldsymbol{zp}}\mathcal{L}(\boldsymbol{z}^\star;\boldsymbol{p}^\star)\| \leq L. \tag{3.2}$$

Assumption 3.5 (γ-Uniform SSOSC)**.** *There exists $\gamma > 0$ such that*

$$ReH(\boldsymbol{G}, \boldsymbol{J}^0) \succeq \gamma \boldsymbol{I}, \tag{3.3}$$

where $\boldsymbol{J}^0 := \boldsymbol{J}[\mathcal{A}^0, :]$; here, $\mathcal{A}^0$, $\boldsymbol{G}$, and $\boldsymbol{J}$ are defined in (2.9).

Assumption 3.6 (β-Uniform LICQ)**.** *There exists $\beta > 0$ such that*

$$\boldsymbol{J}^1(\boldsymbol{J}^1)^\top \succeq \beta \boldsymbol{I}, \tag{3.4}$$

where $\boldsymbol{J}^1 := \boldsymbol{J}[\mathcal{A}^1, :]$; here, $\mathcal{A}^1$ and $\boldsymbol{J}$ are defined in (2.9).

Assumption 3.4 is an extension of Assumption 2.1; in particular, it assumes that the second order derivative not only exists, but also is uniformly bounded. Assumption 3.5,

3.6 are extensions of Assumption 2.2, 2.3, respectively; in particular, the assumptions are strengthened by introducing additional uniform paramters, $\gamma, \beta > 0$. With parameters L, γ, β, we can establish the uniform bounds in (3.1). First, the upper bounds (3.1a)-(3.1b) are trivially obtained.

Lemma 3.4. *Under Assumption 3.4,*

$$\overline{\overline{\sigma}}(\boldsymbol{H}), \overline{\overline{\sigma}}(\boldsymbol{F}) \leq L.$$

Proof. Follows from that $\boldsymbol{H}$, $\boldsymbol{L}$ are the submatrices of the full Lagrangian Hessian $\nabla_{\boldsymbol{zz}}\mathcal{L}(\boldsymbol{z}^\star; \boldsymbol{p}^\star)$ and $\nabla_{\boldsymbol{zp}}\mathcal{L}(\boldsymbol{z}^\star; \boldsymbol{p}^\star)$, respectively. □

To establish a lower bound for $\underline{\underline{\sigma}}_{\boldsymbol{H}}$, we prove the following lemmas.

Lemma 3.5. *Under Assumption 3.4, 3.5, 3.6, we have that:*

$$\underline{\underline{\sigma}}_{\boldsymbol{H}} \geq \left(\frac{2}{\gamma} + \frac{8\overline{\mu}L^2}{\gamma^3\beta} + \frac{4L}{\gamma^2\beta}\right)^{-1} (1 + \overline{\mu}L)^{-1}, \tag{3.5}$$

where $\overline{\mu} := (2L^2/\gamma + \gamma + L)/\beta$.

We first prove that $\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0$ is uniformly positive definite (recall that SSOSC does not necessarily guarantee positive definiteness of $\boldsymbol{G}$).

Lemma 3.6. *Under Assumption 3.4, 3.5, 3.6,*

$$\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0 \succeq (\gamma/2)\boldsymbol{I}.$$

Proof. From Lemma 3.7, $\|\boldsymbol{H}\| \leq L$; this implies that its submatrices $\boldsymbol{G}, \boldsymbol{J}^0$ satisfy $\overline{\sigma}(\boldsymbol{G}), \overline{\sigma}(\boldsymbol{J}^0) \leq L$. The smallest eigenvalue of $\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0$ is obtained from:

$$\min_{w} \; w^\top (\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0) w \tag{3.6a}$$

$$\text{s.t. } \|w\| = 1. \tag{3.6b}$$

By fundamental theorem of linear algebra, any $w \in \mathbb{R}^{\boldsymbol{n_x}}$ can be expressed as $w = Zw_Z + Yw_Y$, where the columns of Z form an orthonormal basis for the null space of $\boldsymbol{J}^0$ and the columns of Y form an orthonormal basis for the row space of $\boldsymbol{J}^0$. We have that

$$\|w_Z\|^2 + \|w_Y\|^2 = 1 \tag{3.7}$$

$$\|Zw_Z\| = \|w_Z\| \tag{3.8}$$

$$\|Yw_Y\| = \|w_Y\|, \tag{3.9}$$

which follows from (3.6b) and orthogonality of Z and Y. The objective (3.6a) satisfies:

$$\begin{aligned}
&w^\top(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0)w \qquad (3.10)\\
&= w_Z^\top Z^\top \boldsymbol{G} Z w_Z + 2w_Y^\top Y^\top \boldsymbol{G} Z w_Z + w_Y^\top Y^\top \boldsymbol{G} Y w_Y + \overline{\mu} w_Y^\top Y^\top (\boldsymbol{J}^0)^\top \boldsymbol{J}^0 Y w_Y\\
&\geq \gamma\|w_Z\|^2 - 2\|\boldsymbol{G}\|\|Zw_Z\|\|Yw_Y\| - \|\boldsymbol{G}\|\|Yw_Y\|^2 + \overline{\mu}\underline{\lambda}(Y^\top(\boldsymbol{J}^0)^\top \boldsymbol{J}^0 Y)\|w_Y\|^2\\
&\geq \gamma(1 - \|w_Y\|^2) - 2\overline{\sigma}(\boldsymbol{G})\|w_Z\|\|w_Y\| - \overline{\sigma}(\boldsymbol{G})\|w_Y\|^2 + \overline{\mu}\underline{\lambda}(\boldsymbol{J}^0 Y Y^\top (\boldsymbol{J}^0)^\top)\|w_Y\|^2\\
&\geq \gamma(1 - \|w_Y\|^2) - 2L\|w_Y\| - L\|w_Y\|^2 + \overline{\mu}\underline{\lambda}(\boldsymbol{J}^0(\boldsymbol{J}^0)^\top)\|w_Y\|^2\\
&\geq \gamma - 2L\|w_Y\| + (\overline{\mu}\beta - \gamma - L)\|w_Y\|^2,
\end{aligned}$$

where $\underline{\lambda}(\cdot)$ denotes the smallest eigenvalue of the symmetric matrix argument. The equality follows from $\boldsymbol{J}^0 Z = \boldsymbol{0}$; the first inequality follows from (i) Assumption 3.5, (ii) submultiplicativity of matrix norms, and (iii) the fact that $w^\top M w \geq \underline{\lambda}(M)\|w\|^2$ for positive definite M; the second inequality comes from (i) Equation (3.7), (ii) the fact that the induced 2-norm is equal to the largest singular value, and (iii) the equality $\underline{\lambda}(MM^\top) = \underline{\lambda}(M^\top M)$ for square M; the third inequality follows from (i) Lemma 3.7 and (ii) $\boldsymbol{J}^0 Y Y^\top = \boldsymbol{J}^0$ since Y is an orthogonal matrix whose columns span the row space of $\boldsymbol{J}^0$; and the last inequality follows from Assumption 3.6.

Since $\mathcal{V}$ is nonempty, we have that $D > 0$; furthermore, we have that $M \neq 0$ from SSOSC and MICQ and thus $L \neq 0$ holds. This implies that $\overline{\mu}\beta - \gamma - L = 2L^2/\gamma > 0$. Accordingly, the quadratic expression on the right-hand side of the last inequality of (3.10) is lower-bounded by:

$$w^\top(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0)w \geq \gamma - \frac{L^2}{\overline{\mu}\beta - \gamma - L} = \frac{\gamma}{2}.$$

□

Proof of Lemma 3.5. It suffices to show that $\underline{\sigma}(\boldsymbol{H}[\mathcal{B},\mathcal{B}])$ for any $\mathcal{B}^0 \subseteq \mathcal{B} \subseteq \mathcal{B}^1$ is lower bounded by the right-hand-side of (3.5). Moreover, we know that $\boldsymbol{H}[\mathcal{B},\mathcal{B}]$ is a permutation of:

$$\begin{bmatrix} \boldsymbol{G} & (\boldsymbol{J}')^\top \\ \boldsymbol{J}', & \end{bmatrix} \tag{3.11}$$

where $\boldsymbol{J}' := \boldsymbol{J}[\mathcal{A},:]$, and $\mathcal{A} := (\phi^{-1}(\mathcal{B}) \setminus \mathbb{I}_{[1,\boldsymbol{n_x}]}) - \boldsymbol{n_x}$; here, $\phi : \mathbb{I}_{[1,\boldsymbol{n}]} \to \mathbb{I}_{[1,\boldsymbol{n}]}$ is a permutation that achieves $\boldsymbol{z}[\phi(i)] = [\boldsymbol{\xi};\boldsymbol{\eta}][i]$. It thus suffices to show that the lowest singular value of the matrix in (3.11) with $\mathcal{A}^0 \subseteq \mathcal{A} \subseteq \mathcal{A}^1$ is lower bounded by the right-hand side of (3.5).

We now make the following observation:

$$(\boldsymbol{J}')^\top \boldsymbol{J}' \succeq (\boldsymbol{J}^0)^\top \boldsymbol{J}^0; \tag{3.12a}$$

$$\underline{\lambda}(\boldsymbol{J}'(\boldsymbol{J}')^\top) \geq \underline{\lambda}(\boldsymbol{J}^1(\boldsymbol{J}^1)^\top); \tag{3.12b}$$

here, the first inequality results from

$$(\boldsymbol{J}')^\top \boldsymbol{J}' - (\boldsymbol{J}^0)^\top \boldsymbol{J}^0 = \boldsymbol{J}[\mathcal{A} \setminus \mathcal{A}^0,:]^\top \boldsymbol{J}[\mathcal{A} \setminus \mathcal{A}^0,:] \succeq \boldsymbol{0}$$

. To establish the second inequality, we consider a unit vector $w \in \mathbb{R}^{\boldsymbol{m}}$ such that $w[\mathcal{A}]$ is the eigenvector of $\boldsymbol{J}'(\boldsymbol{J}')^\top$ associated with the smallest eigenvalue and $w[\mathbb{I}_{[1,\boldsymbol{m}]} \setminus \mathcal{A}] = \boldsymbol{0}$. We can see that:

$$\underline{\lambda}(\boldsymbol{J}^1(\boldsymbol{J}^1)^\top) \leq w[\mathcal{A}^1]^\top \boldsymbol{J}^1(\boldsymbol{J}^1)^\top w[\mathcal{A}^1] = \underline{\lambda}(\boldsymbol{J}'(\boldsymbol{J}')^\top); \tag{3.13}$$

here, the first inequality follows from the fact that $\lambda(\boldsymbol{J}^1(\boldsymbol{J}^1)^\top)$ is the smallest eigenvalue, and the equality follows from the fact that $w[\mathcal{A}^1 \setminus \mathcal{A}] = \boldsymbol{0}$. This establishes the second inequality in (3.12).

We now study the inverse of the matrix in (3.11); note that $\underline{\sigma}(\boldsymbol{H}) = \|\boldsymbol{H}^{-1}\|$ and

$$\begin{bmatrix} \boldsymbol{G} & (\boldsymbol{J}')^\top \\ \boldsymbol{J}' & \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}' & (\boldsymbol{J}')^\top \\ \boldsymbol{J}' & \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{I} & \overline{\mu}(\boldsymbol{J}')^\top \\ & \boldsymbol{I} \end{bmatrix} \tag{3.14a}$$

$$= \begin{bmatrix} T + \overline{\mu} T (\boldsymbol{J}')^\top S \boldsymbol{J}' T & T(\boldsymbol{J}')^\top S \\ S \boldsymbol{J}' T & \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \overline{\mu} (\boldsymbol{J}')^\top \\ & \boldsymbol{I} \end{bmatrix}, \tag{3.14b}$$

where $T := (\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}')^{-1}$, and $S := (\boldsymbol{J}'(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}')(\boldsymbol{J}')^\top)^{-1}$; here, the first equality can be easily verified; and the second equality follows from [15, Proposition 2.8.7]. Now observe that:

$$\underline{\lambda}(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}') \geq \underline{\lambda}(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}^0)^\top \boldsymbol{J}^0) \tag{3.15a}$$

$$\geq \gamma/2; \tag{3.15b}$$

here, the first inequality follows from (3.12) and the second inequality follows from Lemma 3.6. Furthermore,

$$\begin{aligned} \underline{\lambda}(\boldsymbol{J}'(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}')(\boldsymbol{J}')^\top) &\geq \underline{\lambda}(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}') \underline{\lambda}(\boldsymbol{J}'(\boldsymbol{J}')^\top) \\ &\geq \gamma\beta/2; \end{aligned}$$

here, the first inequality follows from

$$\begin{aligned} \min_{\|w\| \leq 1} w^\top (\boldsymbol{J}'(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}')(\boldsymbol{J}')^\top) w &\geq \min_{\|w\| \leq 1} \underline{\lambda}(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}') \|(\boldsymbol{J}')^\top w\|^2 \\ &\geq \underline{\lambda}(\boldsymbol{G} + \overline{\mu}(\boldsymbol{J}')^\top \boldsymbol{J}') \underline{\lambda}(\boldsymbol{J}'(\boldsymbol{J}')^\top), \end{aligned}$$

and the second inequality follows from (3.15) and (3.12). Thus, $\|T\| \leq 2/\gamma$ and $\|S\| \leq 2/\gamma\beta$. By using Lemma 3.8 (will be introduced later), the triangle inequality, the submultiplicativity of matrix norms, and the fact that $\boldsymbol{G}$, $\boldsymbol{J}'$ are submatrices of $\boldsymbol{H}$, we have:

$$\left\| \begin{bmatrix} T + \overline{\mu} T (\boldsymbol{J}')^\top S \boldsymbol{J}' T & T(\boldsymbol{J}')^\top S \\ S \boldsymbol{J}' T & \end{bmatrix} \right\| \leq \frac{2}{\gamma} + \frac{8\overline{\mu} L^2}{\gamma^3 \beta} + \frac{4L}{\gamma^2 \beta} \tag{3.16a}$$

$$\left\| \begin{bmatrix} \boldsymbol{I} & \overline{\mu} (\boldsymbol{J}')^\top \\ & \boldsymbol{I} \end{bmatrix} \right\| \leq 1 + \overline{\mu} L. \tag{3.16b}$$

Therefore, from (3.14) and (3.16), we obtain:

$$\left\| \begin{bmatrix} \boldsymbol{G} & (\boldsymbol{J}')^\top \\ \boldsymbol{J}' & \end{bmatrix}^{-1} \right\| \leq \left(\frac{2}{\gamma} + \frac{8\overline{\mu} L^2}{\gamma^3 \beta} + \frac{4L}{\gamma^2 \beta} \right) (1 + \overline{\mu} L). \tag{3.17}$$

Because (3.17) holds for any $\mathcal{A}^0 \subset \mathcal{A} \subset \mathcal{A}^1$, the desired condition is obtained; the proof is complete. □

We have established in Lemma 3.5 that Assumption 3.4, 3.5, 3.6 guarantee the uniform regularity conditions (3.1). The result is summarized as follows.

Theorem 3.3. *Under Assumption 3.4, 3.5, 3.6, we have that:*

$$\overline{\overline{\sigma}}_{\boldsymbol{H}} \leq L \tag{3.18a}$$

$$\overline{\overline{\sigma}}_{\boldsymbol{F}} \leq L \tag{3.18b}$$

$$\underline{\sigma}_{\boldsymbol{H}} \geq \left(\frac{2}{\gamma} + \frac{8\overline{\mu}L^2}{\gamma^3\beta} + \frac{4L}{\gamma^2\beta}\right)^{-1} (1+\overline{\mu}L)^{-1}, \tag{3.18c}$$

where $\overline{\mu}$ is defined in Lemma 3.5 and $\overline{\overline{\sigma}}_{\boldsymbol{H}}, \overline{\overline{\sigma}}_{\boldsymbol{F}}, \underline{\sigma}_{\boldsymbol{H}}$ are defined in (2.2).

Proof. The result follows directly from Lemma 3.4, 3.5. □

Now that we have the uniform upper and lower bounds of $\overline{\overline{\sigma}}_{\boldsymbol{H}}, \overline{\overline{\sigma}}_{\boldsymbol{F}}, \underline{\sigma}_{\boldsymbol{H}}$, all the quantities in Theorem 2.2 can be expressed using uniform parameters: L, γ, and β. We additionally have the $\epsilon > 0$, but that can be chosen arbitrarily; e.g., one can choose it as $\rho/10$. Therefore, we can uniformly bound the exponential decay parameters (Υ, ρ) using Theorem 3.3.

3.2 Sufficient Conditions for Uniformly Bounded Lagrangian Hessian

We now show that the uBLH (3.2) can be established from the composable conditions. In particular, we assume the boundedness conditions for each node and the boundedness in the graph degree to establish the uniform boundedness condition for the full problem. The key assumptions are stated as follows.

Assumption 3.7 (Uniformly Bounded Degree of Graphs)**.** *There exists a uniform upper bound $D \in \mathbb{I}_{>0}$ of the degrees of nodes in $\mathcal{G}$. That is, for any $i \in \mathcal{V}$,*

$$|\mathcal{N}_{\mathcal{G}}[i]| \leq D.$$

Assumption 3.8 (Uniformly Bounded Second Derivatives)**.** *There exists $C \geq 0$ such that*

$$\|H_{ij}\|, \|F_{ij}\| \leq C$$

for any $i, j \in \mathcal{V}$, where H_{ij} and F_{ij} are defined in (2.18).

Lemma 3.7. *Under Assumption 3.7, 3.8, we have that*

$$\overline{\sigma}_{\boldsymbol{H}}, \overline{\sigma}_{\boldsymbol{F}} \leq CD^2.$$

In order to prove this lemma, we first need to establish a general inequality for matrix norms. The following lemma is a generalization of inequality $\|M\| \leq (\|M\|_1 \|M\|_\infty)^{1/2}$ [77, Corollary 2.3.2].

Lemma 3.8. *Consider $M \in \mathbb{R}^{m \times n}$ with index set families $\mathcal{I}$ and $\mathcal{J}$ that partition $\mathbb{I}_{[1,m]}$ and $\mathbb{I}_{[1,n]}$, respectively. The following holds:*

$$\overline{\sigma}(M) \leq \Big(\Big(\max_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} \|M_{[i][j]}\| \Big) \Big(\max_{j \in \mathcal{V}} \sum_{i \in \mathcal{V}} \|M_{[i][j]}\| \Big) \Big)^{1/2}, \tag{3.19}$$

where M is a graph-structured matrix induced by $(\mathcal{G}, \mathcal{I}, \mathcal{J})$.

Proof. The inequality holds trivially if $M = \mathbf{0}$; we thus assume $M \neq \mathbf{0}$. Consider the left singular vector $v \in \mathbb{R}^n$ of M with singular value $\overline{\sigma}(M)$. We have that $\overline{\sigma}(M)^2 v = MM^\top v$. We let $u = M^\top v$, which yields $\overline{\sigma}(M)^2 v = Mu$; accordingly,

$$\begin{aligned}
\overline{\sigma}(M)^2 \sum_{i \in \mathcal{V}} \|v_{[i]}\| &= \sum_{i \in \mathcal{V}} \Big\| \sum_{j \in \mathcal{V}} M_{[i][j]} u_{[j]} \Big\| && \text{(3.20a)} \\
&\leq \sum_{j \in \mathcal{V}} \Big(\sum_{i \in \mathcal{V}} \|M_{[i][j]}\| \Big) \|u_{[j]}\| && \text{(3.20b)} \\
&\leq \Big(\max_{j \in \mathcal{V}} \sum_{i \in \mathcal{V}} \|M_{[i][j]}\| \Big) \Big(\sum_{j \in \mathcal{V}} \|u_{[j]}\| \Big), && \text{(3.20c)}
\end{aligned}$$

where the first inequality is obtained by applying the triangle inequality and the submultiplicativity of the matrix norm, and by switching the order of summation; the second inequality is obtained from

$$\sum_{i \in \mathcal{V}} \|M_{[i][j]}\| \leq \max_{j \in \mathcal{V}} \sum_{i \in \mathcal{V}} \|M_{[i][j]}\|.$$

Using the same logic, we obtain:

$$\sum_{j\in\mathcal{V}} \|u_{[j]}\| \leq \left(\max_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}} \|M_{[i][j]}\|\right)\left(\sum_{i\in\mathcal{V}} \|v_{[i]}\|\right).$$

From these results, (3.20), and the fact that $v \neq \mathbf{0}$ (by $M \neq \mathbf{0}$), we obtain (3.19). □

Proof of Lemma 3.7. Since $\mathcal{B}^1 \subseteq \mathbb{I}_{[1,\boldsymbol{n}]}$, $\overline{\sigma}(\boldsymbol{H}) \geq \overline{\overline{\sigma}}_{\boldsymbol{H}}$ and $\overline{\sigma}(\boldsymbol{F}) \geq \overline{\overline{\sigma}}_{\boldsymbol{F}}$. Thus, it suffices to show that $\overline{\sigma}(\boldsymbol{H}), \overline{\sigma}(\boldsymbol{F}) \leq CD^2$. As observed in Section 2.3, $\boldsymbol{H}$ and $\boldsymbol{F}$ have bandwidth not greater than two since H_{ij} and F_{ij} equal zero if $d_{\mathcal{G}}(i,j) > 2$. Hence, the number of nonzero blocks on one-block rows or on one-block columns of $\boldsymbol{H}$ and $\boldsymbol{F}$ is at most D^2, since

$$|\mathcal{N}^2_{\mathcal{G}}[i]| \leq \underbrace{1}_{i \text{ itself}} + \underbrace{(D-1)}_{\text{nodes with distance 1}} + \underbrace{(D-1)^2}_{\text{nodes with distance 2}} \leq D^2$$

for any $i \in \mathcal{V}$ (i.e., for any node, there exist at most D^2 nodes within distance two); here, the second inequality follows from $D \geq 1$ (the graph is nonempty). As such, we have:

$$\max_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}} \|H_{ij}\| \vee \max_{j\in\mathcal{V}} \sum_{i\in\mathcal{V}} \|H_{ij}\| \leq CD^2;$$
$$\max_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}} \|F_{ij}\| \vee \max_{j\in\mathcal{V}} \sum_{i\in\mathcal{V}} \|F_{ij}\| \leq CD^2.$$

By Lemma 3.8, $\overline{\sigma}(\boldsymbol{H}) \leq CD^2$ and $\overline{\sigma}(\boldsymbol{F}) \leq CD^2$. □

3.3 Sufficient Conditions for Uniform SSOSC and LICQ

Verifying Assumption 3.5, 3.6 can be challenging if the size of $\boldsymbol{G}$ and $\boldsymbol{J}$ grows indefinitely. Thus, in this section, we provide a composability principles for uSSOSC and uLICQ; in particular, we establish the sufficient conditions for uSSOSC and uLICQ that do not require checking singular values of indefinitely large matrices. The problems of interest can have arbitrarily large graphs (e.g., dynamic optimization with infinite horizons and PDE optimization with an unbounded domains). A key characteristic of such problems is that there exists a recurrent structure (as depicted in Figure 1.1). As such, we can construct sufficient conditions based on uSSOSC and uLICQ over *blocks* of the Hessian and Jacobian

matrices, defined by a partition $\mathcal{J} := \{J_{(m)}\}_{m\in\mathcal{M}}$ of the primal variable index set $\mathbb{I}_{[1,\boldsymbol{n_x}]}$. To state these assumptions, we define the following submatrices of $\boldsymbol{G}$ and $\boldsymbol{J}$ for each $m \in \mathcal{M}$:

$$\begin{aligned}
\boldsymbol{G}_{(m)} &:= \boldsymbol{G}[J_{(m)}, J_{(m)}] \\
\boldsymbol{G}_{(-m)} &:= \boldsymbol{G}[J_{(m)}, \mathbb{I}_{[1,\boldsymbol{n_x}]} \setminus J_{(m)}]; \\
\boldsymbol{J}^{-}_{(m)} &:= \boldsymbol{J}[\mathcal{A}^{-}_{(m)}, J_{(m)}] \\
\boldsymbol{J}^{+}_{(m)} &:= \boldsymbol{J}[\mathcal{A}^{+}_{(m)}, J_{(m)}],
\end{aligned}$$

where:

$$\begin{aligned}
\mathcal{A}^{-}_{(m)} &:= \{i \in \mathcal{A}^0 : \boldsymbol{J}[i, \mathbb{I}_{[1,\boldsymbol{n_x}]} \setminus J_{(m)}] = \boldsymbol{0}\}; \\
\mathcal{A}^{+}_{(m)} &:= \{i \in \mathcal{A}^1 : \boldsymbol{J}[i, J_{(m)}] \neq \boldsymbol{0}\}.
\end{aligned}$$

In words, $\mathcal{A}^{-}_{(m)}$ denotes the set of constraint indices that are exclusively coupled with the variables in $J_{(m)}$, and $\mathcal{A}^{+}_{(m)}$ denotes the set of constraint indices that have nonempty coupling with the variables in $J_{(m)}$. Based on these, we now present the key assumptions.

Assumption 3.9 (Block Diagonal $\boldsymbol{G}$)**.** $\boldsymbol{G}_{(-m)} = \boldsymbol{0}$ *for* $m \in \mathcal{M}$.

Assumption 3.10 (Nonzero Rows of $\boldsymbol{J}$)**.** $\boldsymbol{J}[i,:] \neq \boldsymbol{0}$ *for* $i \in \mathcal{A}^1$.

Assumption 3.11 (Block SSOSC)**.** *There exists* $\gamma > 0$ *such that for any* $m \in \mathcal{M}$*:*

$$ReH(\boldsymbol{G}_{(m)}, \boldsymbol{J}^{-}_{(m)}) \succeq \gamma \boldsymbol{I}.$$

Assumption 3.12 (Block LICQ)**.** *There exists* $\beta > 0$ *such that for any* $m \in \mathcal{M}$*:*

$$\boldsymbol{J}^{+}_{(m)}(\boldsymbol{J}^{+}_{(m)})^{\top} \succeq \beta \boldsymbol{I}.$$

We emphasize that Assumption 3.9 does not assume separability of the problem; a block-diagonal structure in $\boldsymbol{G}$ is obtained when coupling across blocks exists only via linear constraints. This is not a restrictive assumption since any problem of the form (1.1) can be reformulated into a form with linear coupling by introducing auxiliary variables (i.e., via a *lifting* procedure). Assumption 3.10 is not difficult to satisfy.

In the following lemmas, we show that the above assumptions guarantee uSSOSC and uLICQ for the original NLP (1.1).

Lemma 3.9. *Under Assumption 3.9, 3.11 we have*

$$ReH(\boldsymbol{G}, \boldsymbol{J}^0) \succeq \gamma \boldsymbol{I}.$$

Proof. From the block diagonal structure of $\boldsymbol{G}$ (Assumption 3.9),

$$\boldsymbol{x}^\top \boldsymbol{G} \boldsymbol{x} = \sum_{m \in \mathcal{M}} \boldsymbol{x}[J_{(m)}]^\top \boldsymbol{G}_{(m)} \boldsymbol{x}[J_{(m)}].$$

If $\boldsymbol{J}^0 \boldsymbol{x} = \boldsymbol{0}$, $\boldsymbol{J}^-_{(m)} \boldsymbol{x}[J_{(m)}] = \boldsymbol{0}$ holds for $m \in \mathcal{M}$; therefore, by Assumption 3.11, the following can be obtained: if $\boldsymbol{J}^0 \boldsymbol{x} = \boldsymbol{0}$,

$$\sum_{m \in \mathcal{M}} \boldsymbol{x}[J_{(m)}]^\top \boldsymbol{G}_{(m)} \boldsymbol{x}[J_{(m)}] \geq \sum_{m \in \mathcal{M}} \gamma \|\boldsymbol{x}[J_{(m)}]\|^2 = \gamma \|\boldsymbol{x}\|^2. \tag{3.21}$$

Here, the last equality follows from the fact that $\{J_{(m)}\}_{m \in \mathcal{M}}$ partitions $\mathbb{I}_{[1, \boldsymbol{n_x}]}$. From (3.21), we obtain the result. □

Lemma 3.10. *Under Assumption 3.10, 3.12 we have*

$$\boldsymbol{J}^1 (\boldsymbol{J}^1)^\top \succeq \beta \boldsymbol{I}.$$

Proof. We have that for any $\boldsymbol{y} \in \mathbb{I}_{[1, \boldsymbol{m}]}$,

$$\begin{aligned} \boldsymbol{y}[\mathcal{A}^1]^\top \boldsymbol{J}^1 (\boldsymbol{J}^1)^\top \boldsymbol{y}[\mathcal{A}^1] &= \boldsymbol{y}[\mathcal{A}^1]^\top \Big(\sum_{m \in \mathcal{M}} \boldsymbol{J}[\mathcal{A}^1, J_{(m)}] (\boldsymbol{J}[\mathcal{A}^1, J_{(m)}])^\top \Big) \boldsymbol{y}[\mathcal{A}^1] \\ &= \sum_{m \in \mathcal{M}} (\boldsymbol{y}[\mathcal{A}^+_{(m)}])^\top \boldsymbol{J}^+_{(m)} (\boldsymbol{J}^+_{(m)})^\top \boldsymbol{y}[\mathcal{A}^+_{(m)}]. \end{aligned}$$

Here the first equality follows from block multiplication formula and the second equality follows from the fact that $\boldsymbol{J}[\mathcal{A}^1 \setminus \mathcal{A}^+_{(m)}, J_{(m)}] = \boldsymbol{0}$. By Assumption 3.12,

$$\boldsymbol{y}[\mathcal{A}^1]^\top \boldsymbol{J}^1 (\boldsymbol{J}^1)^\top \boldsymbol{y}[\mathcal{A}^1] \geq \sum_{m \in \mathcal{M}} \beta \|\boldsymbol{y}[\mathcal{A}^+_{(m)}]\|^2 \tag{3.22a}$$

$$\geq \beta \|\boldsymbol{y}[\mathcal{A}^1]\|^2. \tag{3.22b}$$

where the second inequality follows from the fact that $\bigcup_{m \in \mathcal{M}} \mathcal{A}^+_{(m)} = \mathcal{A}^1$, which follows from Assumption 3.10. Inequality (3.22) implies the desired result. □

We now summarize the developments in Section 3.2, 3.3 in the following theorem.

Theorem 3.4. *Under Assumption 3.7, 3.8, 3.9, 3.10, 3.11, 3.12, we have*

$$\overline{\overline{\sigma}}_{\boldsymbol{H}} \leq CD^2; \tag{3.23a}$$

$$\overline{\overline{\sigma}}_{\boldsymbol{R}} \leq CD^2; \tag{3.23b}$$

$$\underline{\underline{\sigma}}_{\boldsymbol{H}} \geq \left(\frac{2}{\gamma} + \frac{8\overline{\mu}C^2D^4}{\gamma^3\beta} + \frac{4CD^2}{\gamma^2\beta}\right)^{-1} \left(1 + \overline{\mu}CD^2\right)^{-1}, \tag{3.23c}$$

where $\overline{\mu}$ is defined in Lemma 3.5, and $\overline{\overline{\sigma}}_{\boldsymbol{H}}, \overline{\overline{\sigma}}_{\boldsymbol{F}}, \underline{\underline{\sigma}}_{\boldsymbol{H}}$ are defined in (2.2).

Proof. The result follows from Theorem 3.3 and Lemma 3.7, 3.9, 3.10. □

The results in Section 3.1-3.3 are useful for different problems of interest but might not be applicable to certain problem classes. For instance, it is difficult to derive uniform regularity conditions for multi-stage stochastic programs with a fixed number of children per node because the probability of a given stage decays asymptotically over time (this prevents Assumption 3.11 to hold). This indicates that these types of problems might exhibit parasitic behavior that might manifest as extreme sensitivity (associated with non-uniqueness of the solution). We will leave specialized treatment for those problems as a topic of future work. Also, we have not discussed how the sensitivity behavior changes when the discretization resolution changes; such behavior can be used to understand the sensitivity behavior of the continuous-time (infinite-dimensional) optimization problems studied in [82–84]. This is also left as a topic of future work.

The above results also provide *qualitative* conditions under which quantities in (3.1) are likely to be moderately bounded (and thus the problem exhibits EDS). The first is having *sufficient positive curvature* in the objective function (related to Assumption 3.5, 3.11), and the second is having a *sufficient flexibility* in the constraints (related to Assumption 3.6, 3.12). Indeed, in the absence of nonlinear constraints, the SSOSC implies the strong convexity of the objective function on the null space. In addition, in many practical domains, flexibility is defined as the ability to endure and adjust to the variations in conditions [37, 80, 110, 125, 158, 158]. In the context of sensitivity analysis, this can be interpreted as the ability to

remain feasible without changing the solution too aggressively when the system is subject to data perturbations. The justification is that the big jump in the solution may force the system to violate the constraints. Thus, this notion of flexibility is related to the smallest non-trivial singular value of the active constraint Jacobian. Intuitively, the first qualitative condition helps the decay of sensitivity because positive curvature produces a direction to which the solution tends and the second qualitative condition helps the decay of sensitivity as it enables the solutions to dampen the impact of perturbations. These conditions can be related to specific properties of particular problem classes; for example, for the dynamic optimization problems analyzed in [119, 152], it can be seen that uniform LICQ is related to uniform controllability; similarly, the observability is directly related to SSOSC for state and parameter estimation problems [152, 169]; we will formalize this in the next chapter.

Chapter 4

Dynamic Optimization

In this chapter, we discuss the specialization of the results in Chapter 2-3 for dynamic optimization problems. In particular, we showcase how the composability principles established in Section 3.3 can be applied in the context of dynamic optimization. This allows us to make a formal connection between the system theoretic properties (controllability and observability) with the uniform regularity conditions. Eventually, this allows us to show the EDS with uniformly bounded parameters from the assumptions on the system theoretic properties.

Related Work: Recently, EDS has been established for MPC problems under the inequality-constrained linear-quadratic control setting with the convex objective and controllability assumptions [165, 166] and under nonlinear setting with uSSOSC and uniform controllability assumptions [119]; however, their proof technique is different from ours (they are based on a Riccati recursion). EDS for continuous-time, linear-quadratic MPC problems has been reported in [82–84]. Recently, it has also been shown that EDS is an important property in that that it can be used to construct efficient time-coarsening or discretization schemes [84, 151] and to establish convergence of decomposition algorithms [120, 121]. Furthermore, as revealed in [84] there could be connections between the asymptotic stability of MPC and turnpike properties with EDS and establishing closed-loop stability of MPC [60, 61, 81]. Lastly, there has been recent interest in analyzing the closed-loop regret (performance loss) of finite-horizon MPC policy [112, 113, 173]. We expect that the study of sensitivity may allow a similar regret analysis for nonlinear MPC settings.

We formally define the dynamic optimization (DO) problems of interest:

$$\mathcal{P}_{0:N}(p_{-1:N}): \quad \min_{\substack{s_{0:N} \\ u_{0:N-1}}} \sum_{i=0}^{N-1} h_i(s_i, u_i; p_i) + h_N(s_N; p_N) \tag{4.1a}$$

$$\text{s.t. } Ts_0 = p_{-1} \quad (y_{-1}) \tag{4.1b}$$

$$s_{i+1} = g_i(s_i, u_i; p_i), i \in \mathbb{I}_{[0,N-1]} \quad (y_i). \tag{4.1c}$$

Here, $N \in \mathbb{I}_{>0}$ is the horizon length; for each stage (time) i, $s_i \in \mathbb{R}^{n_x}$ are the states, $u_i \in \mathbb{R}^{n_u}$ are the controls, $p_i \in \mathbb{R}^{n_p}$ are the data (parameters), $y_i \in \mathbb{R}^{n_x}$ are the dual variables, $h_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}$ are the stage cost functions, $g_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ are the dynamic mapping functions, $h_N : \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}$ is the final cost function. The initial state constraint (4.1b) is enforced with the initial state mapping $T \in \mathbb{R}^{n_{y_{-1}} \times n_x}$ and parameter $p_{-1} \in \mathbb{R}^{n_{p_{-1}}}$. We let s_{-1}, u_{-1}, u_N, y_N be empty vectors (for convenience), and define $x_i := [s_i, u_i]$, $z_i := [x_i; y_i]$ for $i \in \mathbb{I}_{[-1,N]}$, and we use the syntax $v_{a:b} := [v_a; v_{a+1}; \cdots; v_b]$ for $v = s, u, y, d, x, z$. Problem (4.1) is a parametric nonlinear program that we denote as $\mathcal{P}_{0:N}(p_{-1:N})$. We assume that all functions are twice continuously differentiable and potentially nonconvex. Typical MPC problems are formulated with $T = I$ and typical moving horizon estimation (MHE) problems are formulated with an empty matrix $T \in \mathbb{R}^{0 \times n_x}$ (i.e., initial constraint is not enforced). State-output mappings encountered in such problem formulations are assumed to be embedded within the stage costs.

The main result of this chapter is a specialization of the EDS result in Chapter 2, 3 to DO problems. Specifically, one can easily see that the DO problem in (4.1) is a gsNLP. First, we state the corollary of Theorem 2.2 for the DO setting in (4.1). Then, we establish the formal connection between the uniform regularity conditions (uBLH, uSSOSC, and uLICQ) and the system theoretic properties (controllability and observability).

4.1 Exponential Decay of Sensitivity

Consider a base data $p^\star_{-1:N}$ and the associated base solution $z^\star_{-1:N}$. We can see from Lemma 2.2 that there exists a well-defined solution mapping $z^\dagger_{-1:N}(\cdot)$ around the neighborhood of $p^\star_{-1:N}$. We now study *stage-wise* solution sensitivity by characterizing the dependence of $z^\dagger_i(\cdot)$ on the data $p_{-1:N}$. First, we define the Lagrangian function for $\mathcal{P}_{0:N}(p_{-1:N})$ as follows:

$$\mathcal{L}_{0:N}(z_{-1:N}; d_{-1:N}) := \sum_{i=0}^{N} \mathcal{L}_i(x_i, y_{i-1:i}; d_i), \tag{4.2}$$

where:

$$\mathcal{L}_i(x_i, y_{i-1:i}; d_i) := \ell_i(x_i; d_i) - y_{i-1}^\top s_i + y_i^\top g_i(x_i; d_i) \tag{4.3a}$$

$$\mathcal{L}_N(s_N, y_{N-1}; d_N) := \ell_N(s_N; d_N) - y_{N-1}^\top s_N. \tag{4.3b}$$

The following is a corollary of Theorem 2.2 applied for the dynamic optimization problem in (4.1).

Corollary 4.1 (EDS in Dynamic Optimization)**.** *Under Assumption 2.1, 3.4, 3.5, 3.6, neighborhoods $\mathbb{P}_{-1:N}$ of $p^\star_{-1:N}$ and $\mathbb{Z}_{-1:N}$ of $z^\star_{-1:N}$ and a continuous $z^\dagger_{-1:N} : \mathbb{P}_{-1:N} \to \mathbb{Z}_{-1:N}$ such that for any $p_{-1:N} \in \mathbb{P}_{-1:N}$, $z^\dagger_{-1:N}(p_{-1:N})$ is a local solution of $\mathcal{P}_{0:N}(p_{-1:N})$, and*

$$\|z_i^\dagger(p_{-1:N}) - z_i^\dagger(p'_{-1:N})\| \leq \sum_{j \in V} \Upsilon \rho^{|i-j|} \|p_j - p'_j\| \tag{4.4}$$

holds for any $p_{-1:N}, p'_{-1:N} \in \mathbb{P}_{-1:N}$ and $i \in \mathbb{I}_{[-1,N]}$, where, Υ and ρ are defined in Theorem 2.2, $\overline{\sigma}_{\boldsymbol{H}}$, $\overline{\sigma}_{\boldsymbol{F}}$, $\underline{\sigma}_{\boldsymbol{H}}$ are defined in Theorem 3.3 (in particular, in the right hand sides of (3.18)).

Proof. The existence of the neighborhood and the solution mapping follows from Lemma 2.2. We observe that $\mathcal{P}_{0:N}(\cdot)$ is graph-structured (induced by $\mathcal{G}_N = (\mathcal{V}_N, \mathcal{E}_N)$, where $\mathcal{V}_N = \{-1, 0, \cdots, N\}$ and $\mathcal{E}_N = \{\{-1, 0\}, \{0, 1\}, \cdots, \{N-1, N\}\}$). From uBLH, uLICQ, and uSSOSC, one can see that assumptions in Theorem 3.3 are satisfied. This implies that the singular values of $\nabla^2_{z_{-1:N} z_{-1:N}} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N})$ are uniformly upper and lower bounded and those of $\nabla^2_{z_{-1:N} p_{-1:N}} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N})$ are uniformly upper bounded (uniform constants

given by functions of L, β, γ; see (3.18)). We then apply Theorem 2.2 to obtain $\Upsilon > 0$ and $\rho \in (0, 1)$ as functions of the upper and lower bounds of the singular values (here, we may choose $\epsilon > 0$ to be sufficiently small). This allows expressing Υ, ρ as functions of L, β, γ as above. □

Corollary 4.1 establishes EDS under the regularity conditions of Assumption 3.4, 3.5, 3.6. It is important that Υ, ρ can be determined solely in terms of L, γ, β (and do not depend on the horizon length N). Practical DO problems typically have additional equality/inequality constraints that are not considered in (4.1). Thus, Corollary 4.1 may not be directly applicable to those problems. However, the results in Theorem 2.2 are applicable to such problems as long as the DO problem is a gsNLP. Specifically, under uniform strong SOSC and uLICQ, we can establish EDS using Theorem 2.2, 3.3.

4.2 Uniform Regularity from System-Theoretic Properties

Although uBLH, uSSOSC, and uLICQ are standard notions of NLP solution regularity, they are not intuitive notions from a system-theoretic perspective. However, we now show that uBLH, uSSOSC, and uLICQ can be obtained from system theoretic properties: uniformly bounded system matrices and uniform controllability and observability. We begin by defining the system matrices:

$$
\begin{aligned}
Q_i &:= \nabla^2_{s_i s_i} \mathcal{L}_i(x_i^\star, y_{i-1:i}^\star; p_i^\star) \\
R_i &:= \nabla^2_{u_i u_i} \mathcal{L}_i(x_i^\star, y_{i-1:i}^\star; p_i^\star) \\
S_i &:= \nabla^2_{s_i u_i} \mathcal{L}_i(x_i^\star, y_{i-1:i}^\star; p_i^\star) \\
E_i &:= \nabla^2_{s_i p_i} \mathcal{L}_i(x_i^\star, y_{i-1:i}^\star; p_i^\star) \\
F_i &:= \nabla^2_{u_i p_i} \mathcal{L}_i(x_i^\star, y_{i-1:i}^\star; p_i^\star) \\
A_i &:= \nabla_{s_i} g_i(x_i^\star; p_i^\star) \\
B_i &:= \nabla_{u_i} g_i(x_i^\star; p_i^\star) \\
O_i &:= \nabla_{p_i} g_i(x_i^\star; p_i^\star),
\end{aligned}
$$

and

$$A_{a:b} := \begin{cases} A_b A_{b-1} \cdots A_{a+1} A_a, \text{ if } a \leq b \\ A_b A_{b+1} \cdots A_{a-1} A_a, \text{ otherwise.} \end{cases}$$

First, we show that the uniform boundedness of system matrices implies uBLH.

Definition 4.3 (Uniformly Bounded System Matrices)**.** *The system matrices* $\{Q_i\}_{i=0}^{N}$, $\{R_i\}_{i=0}^{N-1}$, $\{S_i\}_{i=0}^{N-1}$, $\{A_i\}_{i=0}^{N-1}$, $\{B_i\}_{i=0}^{N-1}$, $\{E_i\}_{i=0}^{N-1}$, $\{F_i\}_{i=0}^{N-1}$, $\{O_i\}_{i=0}^{N-1}$ *is* C*-uniformly bounded if:*

$$\|T\|,\ \|Q_i\|,\ \|R_i\|,\ \|S_i\|,\ \|A_i\|,\ \|B_i\|,\ \|E_i\|,\ \|F_i\|,\ \|O_i\| \leq C.$$

Lemma 4.11. *If the system matrices are* C*-uniformly bounded,* $(3C+1)$*-uBLH holds.*

Proof. First we observe that:

$$\nabla^2_{z_i z_i} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) = \begin{bmatrix} Q_i & S_i^\top & -A_i^\top \\ S_i & R_i & -B_i^\top \\ -A_i & -B_i & \end{bmatrix}$$

$$\nabla^2_{z_{i-1} z_i} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) = \begin{bmatrix} 0 & 0 & I \\ 0 & 0 & 0 \\ I & 0 & 0 \end{bmatrix}$$

$$\nabla^2_{z_i p_i} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) = \begin{bmatrix} E_i \\ F_i \\ O_i \end{bmatrix}$$

$$\nabla^2_{z_i p_j} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) = 0,\ \forall j \neq i$$

$$\nabla^2_{z_i z_j} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) = 0,\ \forall j \notin \{i, i-1\}.$$

Uniform boundedness of the system matrices and by Lemma 3.8, we have that:

$$\|\nabla^2_{\boldsymbol{zz}} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N})\| \leq 3C + 1 \tag{4.5}$$

$$\|\nabla^2_{\boldsymbol{zp}} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N})\| \leq 3C. \tag{4.6}$$

This proves the desired result. □

We define the uniform controllability and uniform observability as follows.

Definition 4.4 (Uniform Controllability)**.** $(\{A_i\}_{i=1}^{N-1}, \{B_i\}_{i=0}^{N-1})$ *is* (N_c, β_c)*-uniformly controllable with* $N_c \in \mathbb{I}_{\geq 0}$ *and* $\beta_c > 0$ *(independent of* N*) if, for any* $i, j \in \mathbb{I}_{[0,N-1]}$ *with* $|i-j| \geq N_c$*,* $\mathcal{C}_{i:j}\mathcal{C}_{i:j}^\top \succeq \beta_c I$ *holds, where*

$$\mathcal{C}_{i:j} := \begin{bmatrix} A_{i+1:j}B_i & \cdots & A_jB_{j-1} & B_j \end{bmatrix}.$$

Definition 4.5 (Uniform Observability)**.** $(\{A_i\}_{i=0}^{N-1}, \{Q_i\}_{i=0}^{N})$ *is* (N_o, γ_o)*-uniformly observable with* $N_o \in \mathbb{I}_{\geq 0}$ *and* $\gamma_o > 0$ *(independent of* N*) if for any* $i, j \in \mathbb{I}_{[0,N-1]}$ *with* $|i-j| \geq N_o$*,* $\mathcal{O}_{i:j}^\top \mathcal{O}_{i:j} \succeq \gamma_o I$ *holds, where*

$$\mathcal{O}_{i:j} := \begin{bmatrix} Q_jA_{i:j-1} \\ \ddots \\ Q_{i+1}A_i \\ Q_i \end{bmatrix}.$$

Note that uniform controllability and observability are stronger versions of their standard counterparts; typically, the controllability and observability are defined for time-invariant systems [34]. One can establish the following duality between uniform controllability and observability.

Proposition 4.2. $(\{A_i\}_{i=1}^{N}, \{B_i\}_{i=0}^{N})$ *is* (N_0, α_0)*-uniformly controllable if and only if* $(\{A_i^\top\}_{i=N}^{1}, \{B_i^\top\}_{i=N}^{0})$ *is* (N_0, α_0)*-uniformly observable.*

Note that the orders of sequences $\{A_i^\top\}_{i=N}^{1}, \{B_i^\top\}_{i=N}^{0}$ are inverted.

Proof. For $i, j \in \mathbb{I}_{[0,N-1]}$ with $|i-j| \geq N_0$:

$$\mathcal{C}_{i:j} = \begin{bmatrix} A_{i+1:j}B_i & \cdots & A_jB_{j-1} & B_j \end{bmatrix} = \begin{bmatrix} B_i^\top A_{j:i+1}^\top \\ \vdots \\ B_{j-1}^\top A_j^\top \\ B_j^\top \end{bmatrix}^\top = \mathcal{O}_{j:i}^\top.$$

Duality follows from $\mathcal{O}_{j:i}^\top \mathcal{O}_{j:i} = \mathcal{C}_{i:j}\mathcal{C}_{i:j}^\top$.

□

We now aim to construct the uniform regularity conditions from the above system-theoretic properties. First, we introduce a few notation. The primal Hessian $\boldsymbol{G}_{0:N}$ of the Lagrangian and the constraint Jacobian $\boldsymbol{J}_{0:N}$ are:

$$\boldsymbol{G}_{0:N} := \nabla^2_{x_{0:N},x_{0:N}} \mathcal{L}_{0:N}(z^\star_{-1:N}; p^\star_{-1:N}) \tag{4.7a}$$

$$\boldsymbol{J}_{0:N} := \nabla_{x_{0:N}} c_{-1:N-1}(x^\star_{0:N}; p^\star_{-1:N}), \tag{4.7b}$$

where $c_{-1:N-1}(\cdot)$ is the constraint function for $\mathcal{P}_{0:N}(\cdot)$. The following technical lemma is needed to show that uniform controllability implies uLICQ.

Lemma 4.12. *Consider a block row/column operator U with block V such that $\|V\| \leq C$ of the form:*

$$U := \begin{bmatrix} I & & \\ V & I & \\ & \ddots & \\ & & I \end{bmatrix}, \begin{bmatrix} I & & V \\ & I & \\ & \ddots & \\ & & I \end{bmatrix}.$$

We have that U, U^{-1} are $(C+1)$-uniformly bounded above.

Proof. Observe that:

$$U^{-1} = \begin{bmatrix} I & & \\ -V & I & \\ & \ddots & \\ & & I \end{bmatrix}, \begin{bmatrix} I & & -V \\ & I & \\ & \ddots & \\ & & I \end{bmatrix}$$

One can easily see that $\|U\|, \|U^{-1}\| \leq 1 + \|V\| \leq 1 + C$. Thus, U, U^{-1} are $(C+1)$-uniformly bounded above. □

We now show one of the main results of the current section: the relationship between uniform controllability and uLICQ.

Lemma 4.13. *C-uniformly bounded system matrices, $TT^\top \succeq \delta I$ for $\delta > 0$, and (N_c, β_c)-uniform controllability of $(\{A_i\}_{i=1}^{N-1}, \{B_i\}_{i=0}^{N-1})$ implies (3.4), where $\beta > 0$ is a function of C, δ, N_c, β_c and independent of N.*

Proof. The Jacobian $\boldsymbol{J}_{0:N}$ has the following form:

$$\boldsymbol{J}_{0:N} = \begin{bmatrix} T & & & & & & \\ -A_0 & -B_0 & I & & & & \\ & & \ddots & & & & \\ & & -A_{N-2} & -B_{N-2} & I & & \\ & & & & -A_{N-1} & -B_{N-1} & I \end{bmatrix}$$

By inspecting the block structure of $\boldsymbol{J}_{0:N}$ and Lemma 3.10, one can see that it suffices to show that the smallest non-trivial singular value of

$$\begin{bmatrix} S & & & & & \\ -A_i & -B_i & I & & & \\ & & \ddots & & & \\ & & -A_{j-1} & -B_{j-1} & I & \\ & & & & -A_j & -B_j \end{bmatrix} \tag{4.8}$$

is $\beta^{1/2}$-uniformly bounded below for $S = T$ or I and for any $i, j \in \mathbb{I}_{[0,N-1]}$ with $N_c \leq |i-j| \leq 2N_c$, where $0 < \beta \leq 1$ is a function of C, δ, N_c, β_c. This follows from the observation that one can always partition $\mathbb{I}_{[0,N-1]}$ into a family of blocks with size between N_c and $2N_c$. For now, we assume $S = I$. By applying a set of suitable block row and column operations (in particular, first apply block row operations to eliminate $A_i, \cdots, A_j$, and then apply block column operations to eliminate $-B_i, \cdots, -A_{i:j-1}B_{j-2}$) and permutations, one can obtain the following:

$$\begin{bmatrix} I & & & & \\ & -A_{i+1:j}B_i & \cdots & -A_jB_{j-1} & -B_j \end{bmatrix}. \tag{4.9}$$

The lower-right blocks constitute the controllability matrix $\mathcal{C}_{i:j}$; from uniform controllability, the smallest non-trivial singular value of the matrix in (4.9) is uniformly lower bounded by $\min(1, \beta_c^{1/2})$. Here, we have applied block-row and block-column operations as the ones that appear in Lemma 4.12 (each multiplied block is uniformly bounded above due to C-uniform boundedness of $\{A_i\}_{i=0}^{N-1}$ and $\{B_i\}_{i=0}^{N-1}$). Also, we have applied such operations only uniformly

bounded many times (the number of operations is independent of N since the number of blocks in the matrix in (4.8) is bounded by $4(2N_c+1)(N_c+1)$, which is uniformly bounded above). We thus have that the smallest non-trivial singular value of the matrix in (4.8) is uniformly lower bounded with uniform constant ${\beta_0}^{1/2}$, and $\beta_0 > 0$ is given by a function of C, N_c, β_c. We now consider the $S = T$ case. One can observe that the smallest non-trivial singular value of the matrix in (4.8) with $S = T$ is lower bounded by that with $S = [\widetilde{T}; T]$ (here, $\widetilde{T}^\top$ is a null space matrix of T); and again, it is lower bounded by $\delta^{1/2}$ times that with $S = I$. We thus have that the smallest non-trivial singular value of the matrix in (4.8) with $S = T$ is uniformly lower bounded by $\beta_0^{1/2}\delta^{1/2}$. Therefore, we have that the smallest non-trivial singular values of the matrices in (4.8) with $S = I$ or T are $\beta^{1/2}$-uniformly lower bounded for any $i, j \in \mathbb{I}_{[0,N-1]}$ with $N_c \leq |i-j| \leq 2N_c$, where $\beta := \min(\beta_0, \delta\beta_0, 1)$. Thus, by Lemma 3.10, we have (3.4). □

If $T \in \mathbb{R}^{0\times n_x}$, the assumption $TT^\top \succeq \delta I$ holds for an arbitrary $\delta > 0$. We now show that uniform observability implies uSSOSC.

Lemma 4.14. *C-uniformly bounded system matrices, $Q_i \succeq 0$, $S_i = 0$, $R_i \succeq rI$ for $r > 0$, and (N_o, γ_o)-uniform observability of $(\{A_i\}_{i=0}^{N-1}, \{Q_i\}_{i=0}^{N})$ implies (3.3), where $\gamma > 0$ is a function of C, N_o, γ_o, r and independent of N.*

Proof. The primal Hessian $\boldsymbol{G}_{0:N}$ has the following form:

$$\boldsymbol{G}_{0:N} = \begin{bmatrix} Q_0 & & & & & \\ & R_0 & & & & \\ & & \ddots & & & \\ & & & Q_{N-1} & & \\ & & & & R_{N-1} & \\ & & & & & Q_N \end{bmatrix}.$$

By inspecting the block structure of $\boldsymbol{G}_{0:N}$ and $\boldsymbol{J}_{0:N}$ and Lemma 3.9, one can observe that it suffices to show that: first,

$$ReH\left(\begin{bmatrix}\boldsymbol{Q}_{i:j} & \\ & \boldsymbol{R}_{i:j-1}\end{bmatrix}, \begin{bmatrix}\boldsymbol{A}_{i:j} & \boldsymbol{B}_{i:j-1}\end{bmatrix}\right) \tag{4.10}$$

has γ-uniformly lower bounded smallest eigenvalue with $\gamma > 0$ for any $i, j \in \mathbb{I}_{[0,N-1]}$ with $N_o \leq |i-j| \leq 2N_o$, where:

$$\boldsymbol{A}_{i:j} := \begin{bmatrix} -A_i & I & & \\ & \ddots & \ddots & \\ & & -A_{j-1} & I \end{bmatrix}, \boldsymbol{B}_{i:j-1} := \begin{bmatrix} -B_i & & \\ & \ddots & \\ & & -B_{j-1} \end{bmatrix}$$

$$\boldsymbol{Q}_{i:j} := \begin{bmatrix} Q_i & & & \\ & Q_{i+1} & & \\ & & \ddots & \\ & & & Q_j \end{bmatrix}, \boldsymbol{R}_{i:j-1} := \begin{bmatrix} R_i & & & \\ & R_{i+1} & & \\ & & \ddots & \\ & & & R_{j-1} \end{bmatrix},$$

and second, $R_i \succeq \gamma I$ for any $i \in \mathbb{I}_{[0,N-1]}$. This follows from the observation that one can always partition $\mathbb{I}_{[0,N-1]}$ into a family of blocks with size between N_c and $2N_c$. We consider $\boldsymbol{s}_{i:j}, \boldsymbol{u}_{i:j-1}$ such that $\boldsymbol{A}_{i:j}\boldsymbol{s}_{i:j} + \boldsymbol{B}_{i:j-1}\boldsymbol{u}_{i:j-1} = 0$ holds. By uniform positive definiteness of $\{R_i\}_{i=0}^{N-1}$ and uniform boundedness of $\{B_i\}_{i=0}^{N-1}$, for $\kappa := r/2C^2$, we have

$$\begin{aligned}\frac{1}{2}\boldsymbol{u}_{i:j-1}^\top \boldsymbol{R}_{i:j-1}\boldsymbol{u}_{i:j-1} &\geq \kappa \boldsymbol{u}_{i:j-1}^\top \boldsymbol{B}_{i:j-1}^\top \boldsymbol{B}_{i:j-1}\boldsymbol{u}_{i:j-1} \\ &= \kappa \boldsymbol{s}_{i:j}^\top \boldsymbol{A}_{i:j}^\top \boldsymbol{A}_{i:j}\boldsymbol{s}_{i:j},\end{aligned}$$

where the equality follows from $\boldsymbol{A}_{i:j}\boldsymbol{s}_{i:j} + \boldsymbol{B}_{i:j-1}\boldsymbol{u}_{i:j-1} = 0$. In addition, from $\boldsymbol{Q}_{i:j} \succeq 0$, we have that:

$$\boldsymbol{s}_{i:j}^\top \boldsymbol{Q}_{i:j}^2 \boldsymbol{s}_{i:j} = (\boldsymbol{Q}_{i:j}^{1/2}\boldsymbol{s}_{i:j})^\top \boldsymbol{Q}_{i:j}(\boldsymbol{Q}_{i:j}^{1/2}\boldsymbol{s}_{i:j}) \leq C\boldsymbol{s}_{i:j}^\top \boldsymbol{Q}_{i:j}\boldsymbol{s}_{i:j},$$

where the inequality follows from that the largest eigenvalue of $\boldsymbol{Q}_{i:j}$ is bounded by C. Thus, $\boldsymbol{s}_{i:j}^\top \boldsymbol{Q}_{i:j}\boldsymbol{s}_{i:j} + \boldsymbol{u}_{i:j-1}^\top \boldsymbol{R}_{i:j-1}\boldsymbol{u}_{i:j-1}$ is not less than:

$$\min(1/C, \kappa)\boldsymbol{s}_{i:j}^\top \begin{bmatrix}\boldsymbol{Q}_{i:j} & \boldsymbol{A}_{i:j}^\top\end{bmatrix} \begin{bmatrix}\boldsymbol{Q}_{i:j} \\ \boldsymbol{A}_{i:j}\end{bmatrix} \boldsymbol{s}_{i:j} + \frac{r}{2}\|\boldsymbol{u}_{i:j-1}\|^2.$$

Observe that $\begin{bmatrix} \boldsymbol{Q}_{i:j} & \boldsymbol{A}_{i:j}^\top \end{bmatrix}$ can be permuted to:

$$\begin{bmatrix} Q_j & I & & & & & \\ & -A_{j-1} & Q_{j-1} & I & & & \\ & & & \ddots & & & \\ & & & -A_{i+1}^\top & Q_{i+1} & I & \\ & & & & & -A_i^\top & Q_i \end{bmatrix}. \tag{4.11}$$

We apply block row and column operations uniformly bounded many times to obtain:

$$\begin{bmatrix} I & & & & \\ & A_{j-1:i}^\top Q_j & \cdots & A_i^\top Q_{i+1} & Q_i \end{bmatrix}. \tag{4.12}$$

From Proposition 4.2 and the (N_o, γ_o)-uniform observability of $(\{A_i\}_{i=0}^{N-1}, \{Q_i\}_{i=0}^{N})$, we have that $(\{A_i^\top\}_{i=N-1}^{0}, \{Q_i\}_{i=N}^{0})$ is (N_o, γ_o)-uniformly controllable. We thus have that the matrix in (4.12) has $\min(1, \gamma_o)$-uniformly lower bounded smallest non-trivial singular value. This implies that the smallest non-trivial singular value of the matrix in (4.11) is uniformly lower bounded by γ', where γ' is given by a function of C, N_o, γ_o. Therefore, we have that:

$$\boldsymbol{s}_{i:j}^\top \boldsymbol{Q}_{i:j} \boldsymbol{s}_{i:j} + \boldsymbol{u}_{i:j-1}^\top \boldsymbol{F}_{i:j-1} \boldsymbol{u}_{i:j-1} \geq \gamma(\|\boldsymbol{s}_{i:j}\|^2 + \|\boldsymbol{u}_{i:j-1}\|^2),$$

where $\gamma := \min(\gamma'/C, \kappa\gamma', r/2)$. One can observe that $R_i \succeq \gamma I$ for any $i \in \mathbb{I}_{[0,N-1]}$. Consequently, the smallest eigenvalues of the matrix in (4.10) for any $i, j \in \mathbb{I}_{[0,N-1]}$ with $N_o \leq |i-j| \leq 2N_o$ and R_i are γ-uniformly lower bounded. Thus, by Lemma 3.9, (3.3) holds. One can confirm that γ is a function of C, N_o, γ_o, r and independent of N. □

The results in Corollary 4.1 and Lemma 4.13, 4.14, 4.11 allow establishing EDS based on system-theoretic properties. We now state the main result of this chapter: EDS in terms of uniformly bounded system matrices and uniform controllability and observability.

Corollary 4.2. *Suppose that Assumption 2.1, C-uniformly bounded system matrices, $Q_i \succeq 0$, $S_i = 0$, $R_i \succeq rI$ for $r > 0$, $TT^\top \succeq \delta I$ for $\delta > 0$, (N_c, β_c)-uniform controllability of $(\{A_i\}_{i=1}^{N-1}, \{B_i\}_{i=0}^{N-1})$, and (N_o, γ_o)-uniform observability of $(\{A_i\}_{i=0}^{N-1}, \{Q_i\}_{i=0}^{N})$ hold; then*

there exist uniform constants $\Upsilon > 0$ *and* $\rho \in (0,1)$ *(functions of* C*,* r*,* N_c*,* β_c*,* N_o*,* γ_o*,* δ *and independent of* N*) and neighborhoods* $\mathbb{P}_{-1:N}$ *of* $p^\star_{-1:N}$ *and* $\mathbb{Z}_{-1:N}$ *of* $z^\star_{-1:N}$ *such that* (4.4) *holds for any* $p_{-1:N}, p'_{-1:N} \in \mathbb{P}_{-1:N}$ *and* $i \in \mathbb{I}_{[-1,N]}$.

Proof. From Corollary 4.1 and Lemma 4.13, 4.14, 4.11. □

4.3 Time-Invariant Setting

Assume now that the system is time-invariant and focuses on a region around a steady state. A special case of Corollary 4.1 for such a setting is derived. We present this result since this setting is of particular interest in the MPC and MHE literature.

Consider a time-invariant system with a stage-cost function $h(\cdot)$, initial regularization function $h_b(\cdot)$, terminal cost function $h_f(\cdot)$, and dynamic mapping $g(\cdot)$. The DO problem is given by (4.1) with $g_i(\cdot) = g(\cdot)$ for $i \in \mathbb{I}_{[0,N-1]}$, $h_i(\cdot) = h(\cdot)$ for $i \in \mathbb{I}_{[1,N-1]}$, $h_0(s,u;p) = h(s,u;p) + h_b(s,u;p)$, and $h_N(s;p) = h_f(s;p)$. With these, we can write the problem (4.1) as follows:

$$\min_{\substack{s_{0:N} \\ u_{0:N-1}}} \; h_b(s_0,u_0;p_0) + \sum_{i=0}^{N-1} h(s_i,u_i;p_i) + h_f(s_N;p_N) \tag{4.13a}$$

$$\text{s.t. } Ts_0 = p_{-1} \quad (y_{-1}) \tag{4.13b}$$

$$s_{i+1} = g(s_i,u_i;p_i), i \in \mathbb{I}_{[0,N-1]} \quad (y_i). \tag{4.13c}$$

In addition, *steady-state* optimization problems are of interest. This problem is typically used for selecting the target for the MPC controller [130] or tuning the terminal cost gradients [62, 168]. The steady-state optimization problem for (4.1) can be written as:

$$\min_{s,u} h(s,u;d) \text{ s.t. } s = g(s,u;d) \quad (y). \tag{4.14}$$

For given p^s and an associated primal-dual solution $z^s := [s^s; u^s; y^s]$ of (4.14), we define:

$$Q := \nabla^2_{ss}\mathcal{L}^s(z^s;p^s)$$

$$S := \nabla^2_{su}\mathcal{L}^s(z^s;p^s)$$

$$R := \nabla^2_{uu}\mathcal{L}^s(z^s; p^s)$$
$$A := \nabla_s g(x^s; p^s)$$
$$B := \nabla_u g(x^s; p^s),$$

where $\mathcal{L}^s(z; d) := h(z; d) - y^\top s + y^\top g(z; d)$; for the initial and terminal cost functions $h_b(\cdot)$ and $h_f(\cdot)$, we define:

$$y_b := \nabla_s h_b(x^s; p^s)$$
$$Q_b := \nabla^2_{ss} h_b(x^s; p^s)$$
$$y_f := \nabla_s h_f(s^s; p^s)$$
$$Q_f := \nabla^2_{ss} h_f(s^s; p^s).$$

The quantities defined above (Q, R, etc.) are independent of N since z^s can be determined independently of N.

Corollary 4.3. *Given twice continuously differentiable $h(\cdot)$, $h_b(\cdot)$, $h_f(\cdot)$, $f(\cdot)$, and data p^s, suppose that there exists a steady-state solution z^s, at which $Q_f \succeq Q \succeq 0$, $Q_b \succeq 0$, $S = 0$, $R \succ 0$, (A, B) controllable, (A, Q) observable, $TT^\top \succ 0$, $y_b + y^s \in$ Range$(T^\top)$ and $y_f = y^s$ hold; then there exist uniform constants $\Upsilon > 0$ and $\rho \in (0, 1)$ such that the following holds: for any $N \in \mathbb{I}_{\geq 0}$, there exist neighborhoods $\mathbb{P}^s_{-1:N}$ of $p^s_{-1:N} := [Ts^s; p^s; \cdots; p^s]$ and $\mathbb{Z}^s_{-1:N}$ of $z^s_{-1:N} := [y^s_{-1}; z^s; \cdots; z^s; s^s]$ such that (4.4) holds for any $p_{-1:N}, p'_{-1:N} \in \mathbb{P}^s_{-1:N}$, where y^s_{-1} is the solution of $T^\top y^s_{-1} = y_b + y^s$.*

Proof. From the existence (follows from $h_b + h^s \in$ Range$(T^\top)$) and uniqueness (follows from $TT^\top \succ 0$) of the solution of $T^\top y^s_{-1} = y_b + y^s$, we have well-defined y^s_{-1}. From $T^\top y^s_{-1} = y_b + y^s$ and the optimality of z^s for (4.14), $z^s_{-1:N}$ satisfies the first-order optimality conditions for $\mathcal{P}_{0:N}(p^s_{-1:N})$. Moreover, all the assumptions in Lemma 4.11 are satisfied with some uniform constant C because $h(\cdot)$, $h_b(\cdot)$, $h_f(\cdot)$, $f(\cdot)$, T, z^s, and p^s are independent of N; thus, by Lemma 4.11, we have (3.2) for a uniform constant $L < \infty$. Moreover, $TT^\top \succeq \delta I$ holds for some uniform constant $\delta > 0$, and $R_i \succeq rI$ for $i \in \mathbb{I}_{[0,N-1]}$ with some uniform constant

$r > 0$, since $h(\cdot)$, z^s, p^s, T are independent of N. Similarly, (A, B) controllability implies (N_c, β_c)-uniform controllability of $(\{A_i\}_{i=1}^{N-1}, \{B_i\}_{i=0}^{N-1})$ with some uniform constant N_c, β_c, and (A, Q) observability implies (N_o, γ_o)-uniform observability of $(\{A_i\}_{i=0}^{N-1}, \{Q_i\}_{i=0}^{N})$ for some uniform constants N_o, γ_o (for now, we assume that $Q_b = 0$ and $Q_f = Q$). From Lemma 4.13, 4.14, we have (3.3) and (3.4) for uniform $\beta, \gamma > 0$. Now, observe that (3.3) for $Q_b = 0$ and $Q_f = Q$ implies (3.3) for any $Q_b \succeq 0$ and $Q_f \succeq Q$; thus, we have (3.3) with uniform $\gamma > 0$ for any Q_b, Q_f. Since the first and second order conditions of optimality and constraint qualifications are satisfied, $z^s_{-1:N}$ is a strict minimizer for $\mathcal{P}_{0:N}(p_{-1:N})$. Since we have (3.2), (3.3), and (3.4) with uniform L, γ, β, we have uBLH, uLICQ, and uSSOSC at $(z^s_{-1:N}, p^s_{-1:N})$. By applying Corollary 4.1, we can obtain (4.4). Lastly, since the parameters $C, r, N_c, \beta_c, N_o, \gamma_o$ are independent of N, so do Υ and ρ. □

Initial and terminal cost functions that satisfy the assumptions in Corollary 4.3 can be constructed as:

$$\begin{aligned} h_b(s, u; p) &:= -((I - T^+T)y^s)^\top s \\ h_f(s; p) &:= (s - s^s)^\top Q(s - s^s) + (y^s)^\top s, \end{aligned}$$

where $(\cdot)^+$ is the pseudoinverse of the argument. In particular, one can verify that

$$\begin{aligned} y_b + y^s &= \nabla_s h_b(x^s; p^s) + y^s = T^+Ty^s = T^\top(TT^\top)^{-1}Ty^s \in \text{Range}(T^\top) \\ Q_b &= \nabla^2_{ss} h_b(x^s; p^s) = 0 \succeq 0 \\ y_f &= \nabla_s h_f(s^s; p^s) = y^s \\ Q_f &= \nabla^2_{ss} h_f(s^s; p^s) = Q \succeq 0. \end{aligned}$$

One can observe that $h_b(\cdot)$ can be set to constantly zero if $T = I$.

Chapter 5

Numerical Experiments

In this chapter, we illustrate the theoretical developments with different classes of graph-structured problems. We conduct numerical experiments for four different classes of graph-structured optimization problems: dynamic optimization (storage control; Appendix B.1), stochastic optimization (stochastic storage control; Appendix B.3), PDE-constrained optimization (thin plate temperature control with Neumann boundary condition; Appendix B.4), and network optimization (alternating current optimal power flow (AC OPF); Appendix B.6). The specific problem formulations can be found in the appendix. We are particularly interested in exploring the effect of conditioning on the EDS. We note that when one of the regularity conditions (uBLH, uSSOSC, and uLICQ) are close to be violated, the conditioning may become bad (i.e., may have a large condition number). As we discussed in Section 3.3, uSSOSC is related to positive objective curvature and uLICQ is related to flexibility. Throughout the case study instances, η and b are parameters that control positive objective curvature and flexibility, respectively. In particular, η is the coefficient of regularization on the decision variables, and b is either the coefficient of the manipulate variables that appear in the constraints or the upper bound of the slack variable. We varied the values of η, b to see the effect of the satisfaction of regularity on the decay of sensitivity. Moreover, j represents the node where the data perturbation will be introduced. We point the readers to our previous publications for more numerical results on the exponential decay of sensitivity [121, 144].

Table 5.1 Variation of (η, b) in numerical studies.

	Case 1	Case 2	Case 3	Case 4
Dynamic Optimization	$(1, 1)$	$(10^{-2}, 1)$	$(1, 10^{-2})$	$(10^{-2}, 10^{-2})$
Stochastic Optimization	$(1, 1)$	$(10^{-2}, 1)$	$(1, 10^{-2})$	$(10^{-2}, 10^{-2})$
PDE Optimization	$(1, 1)$	$(10^{-2}, 1)$	$(1, 10^{-2})$	$(10^{-2}, 10^{-2})$
Network Optimization	$(10^{6}, 10)$	$(0, 10)$	$(10^{6}, 0)$	$(0, 0)$

5.1 Methods

We conduct the following numerical study for each problem instance. We consider a problem $\mathcal{P}(\boldsymbol{p}^\star)$ with the base data $\boldsymbol{p}^\star$. Then, we consider perturbed problems $\{\mathcal{P}(\boldsymbol{p}^{(m)})\}_{m\in\mathcal{M}}$ in which the data are perturbed as $\boldsymbol{p}^{(m)} = \boldsymbol{p}^\star + \Delta\boldsymbol{p}^{(m)}$, where $\Delta\boldsymbol{p}^{(m)}$ are i.i.d samples drawn from $\Delta p_j \sim \mathcal{U}([-\sigma, \sigma]^{l_j})$, and $\Delta p_i = \mathbf{0}$ if $i \neq j$. Here, $j \in \mathcal{V}$ is a selected perturbation point and $\mathcal{U}(\Omega)$ denotes the multivariate uniform distribution on Ω. We choose $\sigma = 10^{-3}$ and $|\mathcal{M}| = 30$ for all instances. Then, the *empirical* sensitivity coefficients:

$$\overline{C}_{ij} = \max_{m\in\mathcal{M}} \|z_i^\ddagger(\boldsymbol{p}^{(m)})\|/\|\Delta\boldsymbol{p}^{(m)}\|, \quad i \in \mathcal{V}$$

are computed and visualized. The empirical sensitivity $\overline{C}_{ij}$ converges to $\|\nabla_{p_j} z_i^\dagger(\boldsymbol{p}^\star)\|$ as $\sigma \to 0$ and the number of samples tends to infinity; thus, these empirical sensitivities are suitable quantities for the study of sensitivity coefficients. We recall that (η, b) are the key parameters that control the positive curvature and flexibility. We vary these parameters as shown in Table 5.1, and see how they affect the decay (spread) of the sensitivity coefficients. Here, Case 1 has sufficiently large (η, b); Case 2 has low η; Case 3 has low b; and Case 4 has low (η, b). The results can be reproduced using the scripts provided in https://github.com/zavalab/JuliaBox/tree/master/SensitivityNLP.

5.2 Results

The sensitivity results are illustrated as heat maps of the empirical coefficients (Figure 5.1) and as scatter plots of the coefficients against distance $d_{\mathcal{G}}(i, j)$ (Figure 5.2). From Figure

Figure 5.1 Spread of empirical sensitivity coefficients $\overline{C}_{ij}/\overline{C}_{jj}$ on $\mathcal{G}$ for dynamic optimization (top), stochastic optimization (second row), PDE optimization (third row), and network optimization (bottom) problem for different values of (η, b). Red circles denote perturbation point, dark blue approaches one, and white approaches zero.

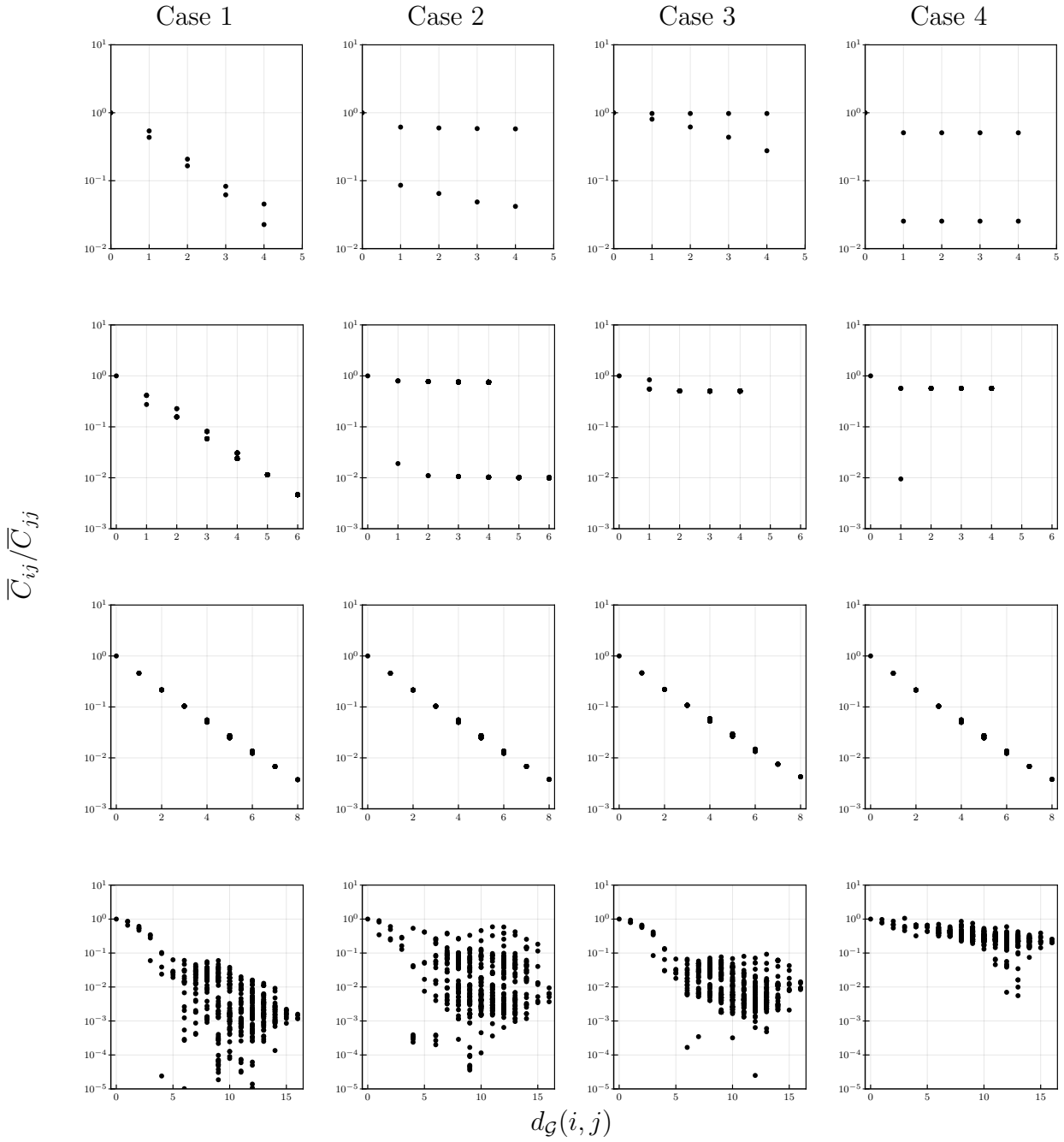

Figure 5.2 Scatter plots of sensitivity coefficients $\overline{C}_{ij}/\overline{C}_{jj}$ versus $d_{\mathcal{G}}(i,j)$ for dynamic optimization (top), stochastic optimization (second row), PDE optimization (third row), and network optimization (bottom) for different values of (η, b).

5.1 we see that, with sufficiently large (η, b) (Case 1), the empirical sensitivity coefficients decay as they move away from the perturbation location. Furthermore, from Figure 5.2, one can confirm that the sensitivity coefficients decay exponentially with distance (i.e., $\log C_{ij} \propto d_{\mathcal{G}}(i,j)$). This verifies the theoretical results in Chapter 2. If either one or both of (η, b) are not sufficiently large (Case 2, 3, 4), the decay of sensitivity is weaker or not observed (except for the PDE optimization problem). This is because, without strong curvature or flexibility, $\underline{\sigma}(\boldsymbol{H}_{(k)})$ may be close to zero, and the coefficients in Theorem 2.2 do not exhibit sufficient decay. The reason that the PDE optimization problem exhibits decay of sensitivity even in the absence of positive curvature and flexibility is that the system itself has a strong dissipative property (temperature naturally tends towards ambient temperature via convection and radiation). From these results, we can confirm that it is sufficient for problems to have strong positive curvature and flexibility in the constraints to exhibit decay of sensitivity (this confirms the theoretical results in Chapter 3). Notably, even though we cannot guarantee uniform boundedness of the multi-stage stochastic programs for the $T \to \infty$ limit, we can observe EDS for sufficiently large (η, b).

5.3 Additional Results: Quadrotor Motion Planning

Additionally, we demonstrate the results in Chapter 4 with the quadrotor motion planning problem in Appendix B.2. Figure 5.3 demonstrate the result. We have empirically tested the sensitivity behavior for $\eta = b = 1$ (Case 1) and $\eta = b = 0$ (Case 2). One can see that some of the assumptions (e.g., $S_i = 0$ in Corollary 4.2) may be violated, but qualitatively, the system is more observable and controllable in Case 1 than in Case 2. The base trajectories are shown as dashed lines, the perturbed trajectories are shown as solid gray lines, and the perturbed stages are highlighted using vertical lines. We can see that, for Case 1 ($(\eta, b) = (1, 1)$), the differences between the base and perturbed solutions become small as moving away from the perturbation point (EDS holds). On the other hand, for Case 2 ($(\eta, b) = (0, 0)$) one cannot observe EDS; this confirms that observability and controllability induce EDS.

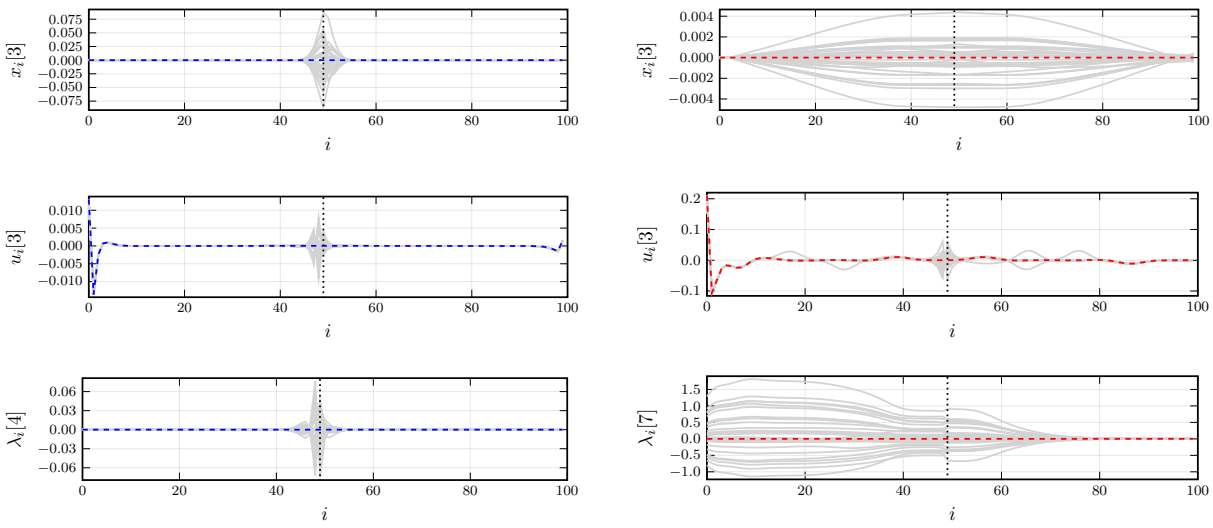

Figure 5.3 Base and perturbed solutions of the quadrotor motion planning problem. Left: Case 1 ($(\eta, b) = (1, 1)$). Right: Case 2 ($(\eta, b) = (0, 0)$).

Part II

Algorithms

Chapter 6

Overlapping Schwarz Method

This chapter presents an overlapping Schwarz method (OSM) for solving general gsNLPs. As its name suggests, OSM decomposes the full problem into *subproblems* that are defined over overlapping subdomains. Solutions for the subproblems are computed in parallel and convergence is enforced by updating primal-dual information in the overlapping regions. With this method, we can solve large-scale gsNLPs that are difficult to be solved with centralized solution algorithms.

Related Work: Diverse decomposition schemes that exploit the problem structure have been proposed in the literature to overcome scalability limits of centralized schemes [41,56,96, 128,148]. A wide range of decomposition schemes have been proposed in the literature such as Lagrangian decomposition and its variant [111], the alternating direction method of multipliers (ADMM) [27], Jacobi/Gauss-Seidel methods [150], and augmented Lagrangian based alternating direction inexact Newton method (ALADIN) [91]. The decomposition allows not directly formulating/solving the intractably large full problem. Thus, with decomposition, one can solve the large-scale gsNLP instances that cannot be solved using the off-the-shelf NLP solvers. However, due to the limited communication between the distributed agents, typically decomposition methods tend to have slow linear convergence (e.g., see [107] for a benchmark of different decomposition techniques). One exception is ALADIN, but it requires the solution of coupled quadratic programs; thus, it has scalability limitations. Schwarz algorithms were originally developed for the parallel solution of linear algebra systems arising

in PDEs, but such schemes can also be used to handle general linear systems and optimization problems by exploiting their underlying algebraic topology [30,69]. In our recent works, we have generalized OSM for graph-structured optimization problems [145,153]. Like other decomposition methods, OSM has linear convergence, but one can improve the convergence rate by using the size of overlap; this enables faster convergence in practice.

The procedure of OSM can be roughly described as follows.

1. User provides a gsNLP (full problem), an overlapping partition of the node set, the associated non-overlapping partition, and the initial guess of the primal-dual solution.
2. Formulate subproblem for each overlapping subdomain.
3. The coupled variables are fixed to the current guess of the solution.
4. Solve each subproblem in parallel to obtain the primal-dual solution over the associated overlapping subdomain.
5. For each subproblem, retain the piece of the primal-dual solutions associated with the non-overlapping subdomain and discard the rest of the solution.
6. Assemble the solution from each subproblem to make the next guess of the solution.
7. Repeat 3-6 until converged to the solution (one can check the KKT residuals to monitor the convergence).

The subproblem formulation and the algorithm will be described in more detail in the following sections. Non-overlapping partitions can be obtained by first creating a non-overlapping partition (e.g., by using a graph partitioning tool), and the overlapping partition can be obtained by expanding each non-overlapping subdomain by progressively incorporating the neighboring nodes. The restriction step 5 is necessary since the subdomains associated with the subproblems overlap; to uniquely specify the next guess of the solution, one needs to discard certain parts of the obtained subproblem solutions. We will see later that this restriction step is designed to exploit the EDS. Properly formulating the subproblem is the key

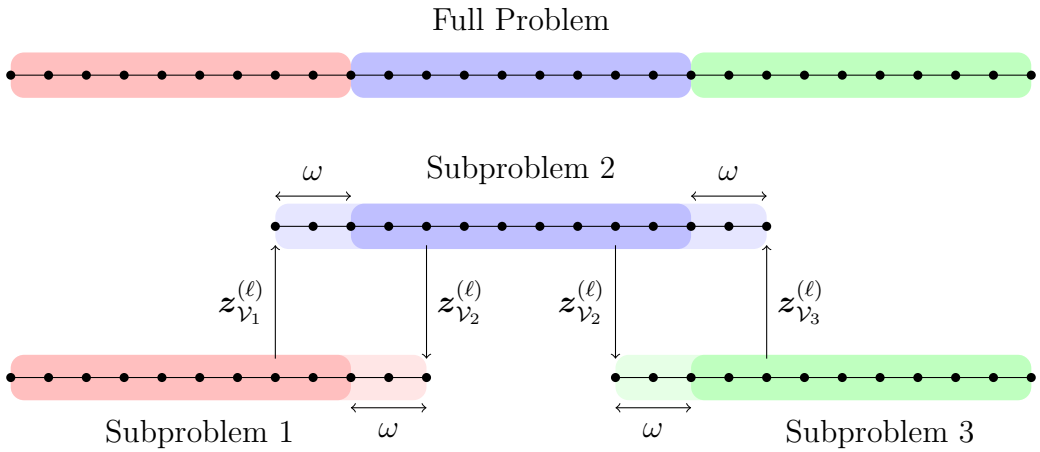

Figure 6.1 Schematic of the overlapping Schwarz method

to guaranteeing the convergence, and the improper formulation of the subproblems may lead to convergence issues. This algorithm can be implemented in a fully decentralized manner, and different updating schemes can be used (e.g., Gauss-Seidel or asynchronous) [153]. Overlapping Schwarz schemes provide a bridge between fully decentralized Jacobi/Gauss-Seidel algorithms (no overlap) and centralized algorithms (the overlap is the entire domain). A schematic of the algorithm is depicted in Figure 6.1.

In this chapter, we analyze the convergence properties of the algorithm and derive an explicit relationship between its convergence rate and the size of overlap. This result extends existing convergence results reported for unconstrained/constrained QPs [145, 153]. In particular, we show that the algorithm locally converges with a sufficiently large overlap and that the convergence rate improves exponentially with the size of overlap. This convergence result relies on EDS, discussed in Part I.

The rest of the chapter is organized as follows. In Section 6.1, we present the subproblem formulation for the OSM. We show that the proposed subproblem is consistent with the full problem and inherits the uniform regularity of the full problem. That is, the full problem satisfies the first-order optimality conditions for the subproblems, and the uniform regularity of the subproblems can be obtained from the uniform regularity of the subproblems. With these results on the subproblems and building upon the results in the previous chapter, we

establish the EDS for the subproblem solutions with respect to the primal-dual solution guess (i.e., the effect of misspecification of the primal-dual solution decays exponentially).

The remainder of the chapter is organized as follows. In Section 6.1, we establish the subproblem sensitivity result. In Section 6.2, the OSM is formally defined. In Section 6.3, we show the local convergence of the algorithm based on the sensitivity result.

6.1 Subproblem Formulation and Sensitivity

For the rest of the chapter, we will study the following modified gsNLP:

$$\mathcal{P}: \min_{\{x_i\}_{i\in\mathcal{V}}} \sum_{i\in\mathcal{V}} f_i(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) \tag{6.1a}$$

$$\text{s.t. } c_i^E(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) = 0,\ i\in\mathcal{V},\ (y_i^E) \tag{6.1b}$$

$$c_i^I(x_i) \geq 0,\ i\in\mathcal{V},\ (y_i^I). \tag{6.1c}$$

Since we study a problem without perturbations, we do not express the problem in the perturbed form as in (1.1) (i.e., we do not explicitly express the dependence on data). Another notable difference is that the the inequality constraint function is only dependent on the associated nodal variable. Any problem in (1.1) form can be reformulated as the one in (6.1) by introducing slack variables:

$$c_i^I(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) \geq 0 \iff c_i^I(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) = s_i,\ s_i \geq 0. \tag{6.2}$$

Thus, this reformulation does not deteriorate the generality in the formulation. We denote the problem in (6.1) by $\mathcal{P}$.

6.1.1 Subproblem Formulation

In this section, we discuss how to formulate the subproblems for the OSM. Properly formulating the subproblem is important because otherwise, the algorithm does not converge to the solution. To enable local convergence, the subproblem needs to

(i) *be consistent* with the full problem.

(ii) *inherits the uniform regularity* of the full problem.

To be more specific, first, the subproblems should be consistent with the full problem in the sense that the appropriate piece of the solution of the full problem should satisfy the first-order optimality conditions for each subproblem as long as the boundary data (the fixed coupled variables) are accurately specified. Second, if the full problem satisfies the uniform regularity conditions (uBLH, uSOSC, and uLICQ), it is desired that the subproblems also satisfy those uniform regularity conditions. This eventually enables guaranteeing the EDS in the subproblems. Ensuring the consistency and the inheritance of regularity requires a careful subproblem definition. Below we show that by properly incorporating the partial augmented Lagrangian, one can satisfy such consistency and inheritance requirements.

To facilitate the later discussions, we introduce a few notations. For a subset $\mathcal{V}' \subseteq \mathcal{V}$ of the node set $\mathcal{V}$, we define the variables associated with node subset $\mathcal{V}'$:

$$\boldsymbol{x}_{\mathcal{V}'} := \{x_i\}_{i\in\mathcal{V}'} \tag{6.3a}$$

$$\boldsymbol{y}_{\mathcal{V}'} := \{y_i\}_{i\in\mathcal{V}'} \tag{6.3b}$$

$$\boldsymbol{z}_{\mathcal{V}'} := \{z_i\}_{i\in\mathcal{V}'} \tag{6.3c}$$

$$\boldsymbol{c}_{\mathcal{V}'}(\boldsymbol{x}) := \{c_i(\boldsymbol{x})\}_{i\in\mathcal{V}'} \tag{6.3d}$$

$$\boldsymbol{c}^E_{\mathcal{V}'}(\boldsymbol{x}) := \{c^E_i(\boldsymbol{x})\}_{i\in\mathcal{V}'} \tag{6.3e}$$

$$\boldsymbol{c}^I_{\mathcal{V}'}(\boldsymbol{x}) := \{c^I_i(\boldsymbol{x})\}_{i\in\mathcal{V}'} \tag{6.3f}$$

$$\boldsymbol{f}_{\mathcal{V}'}(\boldsymbol{x}) := \sum_{i\in\mathcal{V}'} f_i(\boldsymbol{x}). \tag{6.3g}$$

Moreover, for a pair of subsets $(\mathcal{V}', \mathcal{V}'')$, we define $\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''} := (\boldsymbol{x}_{\mathcal{V}'}, \boldsymbol{y}^E_{\mathcal{V}''}, \boldsymbol{y}^I_{\mathcal{V}'})$. In addition, we introduce the following generalized notions of distance and neighborhood:

$$\begin{aligned}
d_{\mathcal{G}}(\mathcal{V}', \mathcal{V}'') &:= \min\{d_{\mathcal{G}}(i,j) : i \in \mathcal{V}', j \in \mathcal{V}''\} \\
d_{\mathcal{G}}(i, \mathcal{V}'') &:= d_{\mathcal{G}}(\{i\}, \mathcal{V}'') \\
d_{\mathcal{G}}(\mathcal{V}', j) &:= d_{\mathcal{G}}(\mathcal{V}', \{j\}) \\
\mathcal{N}^B_{\mathcal{G}}[\mathcal{V}'] &:= \{j \in \mathcal{V} : d_{\mathcal{G}}(j, \mathcal{V}') \leq B\}
\end{aligned}$$

$$\mathcal{N}_{\mathcal{G}}^{B}(\mathcal{V}') := \mathcal{N}_{\mathcal{G}}^{B}[\mathcal{V}'] \setminus \mathcal{V}'$$
$$\mathcal{N}_{\mathcal{G}}[\mathcal{V}'] := \mathcal{N}_{\mathcal{G}}^{1}[\mathcal{V}']$$
$$\mathcal{N}_{\mathcal{G}}(\mathcal{V}') := \mathcal{N}_{\mathcal{G}}^{1}(\mathcal{V}').$$

Now consider subsets $\mathcal{V}'', \mathcal{V}' \subseteq \mathcal{V}$ of the vertex set such that satisfy $\mathcal{N}_{\mathcal{G}}[\mathcal{V}''] \subseteq \mathcal{V}'$; we define the subproblem for $\mathcal{P}$ and a subset pair $(\mathcal{V}', \mathcal{V}'')$ as follows:

$$\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}}): \min_{\boldsymbol{x}_{\mathcal{V}'}} \boldsymbol{f}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']}(\boldsymbol{x}_{\mathcal{V}'}; \overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')}) - (\overline{\boldsymbol{y}}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''})^{\top} \boldsymbol{c}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}; \overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')}) \tag{6.4a}$$
$$+ (\mu/2)\|\boldsymbol{c}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}; \overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')})\|^2$$
$$\text{s.t. } \boldsymbol{c}^{E}_{\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}) = 0, \ (\boldsymbol{y}^{E}_{\mathcal{V}''}) \tag{6.4b}$$
$$\boldsymbol{c}^{I}_{\mathcal{V}'}(\boldsymbol{x}_{\mathcal{V}'}) \geq 0, \ (\boldsymbol{y}^{I}_{\mathcal{V}'}). \tag{6.4c}$$

where $\overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')}$ and $\overline{\boldsymbol{y}}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}$ are fixed data; μ is the penalty parameter. The problem in (6.4) is denoted as a parametric optimization problem: $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$. Note that $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ reduces to the full problem $\mathcal{P}$ if $\mathcal{V}' = \mathcal{V}'' = \mathcal{V}$. We denote the solution of (6.4) as $\boldsymbol{z}^{\star}_{\mathcal{V}',\mathcal{V}''} = (\boldsymbol{x}^{\star}_{\mathcal{V}'}, \boldsymbol{y}^{E,\star}_{\mathcal{V}''}, \boldsymbol{x}^{I,\star}_{\mathcal{V}'})$.

Problem $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ is created by modifying the full problem $\mathcal{P}$. It allows the primal variables in $\mathcal{V}'$ to vary while fixing the rest of primal variables; it enforces inequality constraints associated with $\mathcal{V}'$ and equality constraints associated with $\mathcal{V}''$ (i.e., allow the associated dual variables to vary), while relaxing others (i.e., fix the associated dual variables). Since $\mathcal{N}_{\mathcal{G}}[\mathcal{V}''] \subseteq \mathcal{V}'$, the equality constraints are only dependent on the variables in $\mathcal{V}'$ and not dependent on the variables in $\mathcal{V} \setminus \mathcal{V}'$; this allows us to write the equality constraints in (6.4b) form. Since the objective terms in $\mathcal{V} \setminus \mathcal{N}_{\mathcal{G}}[\mathcal{V}']$ do not depend on the variables in $\mathcal{V}'$, it suffices to only include the objective terms associated with $\mathcal{N}_{\mathcal{G}}[\mathcal{V}']$ as in (6.4a). Recall from (6.1) that variables in $\mathcal{V}'$ are also dependent on the equality constraints in $\mathcal{N}_{\mathcal{G}}[\mathcal{V}'] \setminus \mathcal{V}''$. Rather than enforcing them as costraints, we use partial augmented Lagrangian to incorporate them in the objective function as in (6.4a). Also, note that since this subproblem is still coupled with the neigbhoring nodes, the primal variables in $\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')$ and the dual variables in $\mathcal{N}_{\mathcal{G}}[\mathcal{V}'] \setminus \mathcal{V}''$ (in particular, those associated with equality constraints) still appear in the problem. These

coupled variables are assumed to be fixed, and incorporated as data $\overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')}$ and $\overline{\boldsymbol{y}}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}$. To mitigate the notational complexity, we express the data dependency simply as $\overline{\boldsymbol{z}}$ (i.e., we assume that we have primal-dual guess of the solution over the entire node set $\mathcal{V}$); this is allowed since $\overline{\boldsymbol{z}}$ includes $\overline{\boldsymbol{x}}_{\mathcal{N}^2_{\mathcal{G}}(\mathcal{V}')}$ and $\overline{\boldsymbol{y}}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}$.

Remark 6.4. *Our subproblem formulation in* (6.4) *is different from the ones that appear in our previous works [145]. When we were working on those papers, we were not concerned about the inheritance of the uniform regularity and only cared about the consistency. Due to this, we were not able to rigorously construct the uniform regularity of the subproblems and our convergence analysis had to rely on conjectures [145, Assumption 4]. In fact, in order to guarantee the inheritance of uniform regularity, we needed to treat the coupled equality constraints separately when constructing the subproblem. Otherwise, one cannot write the equality constraints in* (6.4b) *form; instead, the equality constraints may have fixed variables in them. This can potentially cause the violation of uLICQ in the subproblem. Thus, we need two kinds of overlapping partitions $\mathcal{V}'$ and $\mathcal{V}''$; the equality constraints in $\mathcal{V}''$ are directly enforced, the constraints coupled with $\mathcal{V}'$ but not in $\mathcal{V}''$ are relaxed and incorporated into objective function as partial augmented Lagrangian form. Our treatment of coupled equality constraints as partial augmented Lagrangian in* (6.4) *allows preventing the potential violation of the uLICQ conditions.*

6.1.2 Consistency

We now show that $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ is a consistent subproblem formulation; that is, we show that for the solution $\boldsymbol{z}^\star$ of $\mathcal{P}$, the restriction $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$ of the solution $\boldsymbol{z}^\star$ on $\mathcal{V}', \mathcal{V}''$ always satisfy the first-order optimality condition for the subproblem $P^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ (the subproblem with fixed data $\overline{\boldsymbol{z}} = \boldsymbol{z}^\star$).

Lemma 6.15. *Let $\boldsymbol{z}^\star$ be the primal-dual solution of $\mathcal{P}$; then for any $\mu \geq 0$, $\mathcal{V}', \mathcal{V}'' \subseteq \mathcal{V}$, and $\mathcal{N}[\mathcal{V}''] \subseteq \mathcal{V}'$, $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$ satisfies the first-order optimality conditions for $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$.*

Proof. From the first-order optimality conditions for $\mathcal{P}$ at $\boldsymbol{z}^\star$, we have that the following holds for $i \in \mathcal{V}$:

$$\nabla_{x_i} \mathcal{L}(\boldsymbol{z}^\star) = 0, \tag{6.5a}$$

$$c_i^E(\boldsymbol{x}^\star) = 0 \tag{6.5b}$$

$$c_i^I(\boldsymbol{x}^\star) \geq 0 \tag{6.5c}$$

$$y_i^{I,\star} \geq 0 \tag{6.5d}$$

$$\operatorname{diag}(y_i^{I,\star}) c_i^I(\boldsymbol{x}) = 0. \tag{6.5e}$$

We denote the Lagrangian function of (6.4) as

$$\begin{aligned} \mathcal{L}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = \boldsymbol{f}_{\mathcal{N}_\mathcal{G}[\mathcal{V}']}(\boldsymbol{x}_{\mathcal{V}'}; \boldsymbol{x}^\star_{\mathcal{N}^2_\mathcal{G}(\mathcal{V}')}) - (\boldsymbol{y}^{E,\star}_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''})^\top \boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}; \boldsymbol{x}^\star_{\mathcal{N}^2_\mathcal{G}(\mathcal{V}')}) \\ + (\mu/2)\|\boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}; \boldsymbol{x}^\star_{\mathcal{N}^2_\mathcal{G}(\mathcal{V}')})\|^2 \\ - (\boldsymbol{y}^E_{\mathcal{V}''})^\top \boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}_{\mathcal{V}'}) - (\boldsymbol{y}^I_{\mathcal{V}'})^\top \boldsymbol{c}^I_{\mathcal{V}'}(\boldsymbol{x}_{\mathcal{V}'}). \end{aligned} \tag{6.6}$$

By inspecting (6.6), we can see that for any $\mu \geq 0$ and $i \in \mathcal{V}'$,

$$\nabla_{x_i} \mathcal{L}(\boldsymbol{z}^\star) = \sum_{j \in \mathcal{N}_\mathcal{G}[i]} \nabla_{x_i} f_j(\boldsymbol{x}^\star) - \sum_{j \in \mathcal{N}_\mathcal{G}[i]} (\nabla_{x_i} c_j^E(\boldsymbol{x}^\star))^\top y_j^{E,\star} - (\nabla_{x_i} c_i^I(\boldsymbol{x}^\star))^\top y_i^{I,\star} \tag{6.7a}$$

$$= \nabla_{x_i} \boldsymbol{f}_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star) - (\nabla_{x_i} \boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star))^\top \boldsymbol{y}^{E,\star}_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''} \tag{6.7b}$$

$$\quad - (\nabla_{x_i} \boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}^\star))^\top \boldsymbol{y}^{E,\star}_{\mathcal{V}''} - (\nabla_{x_i} \boldsymbol{c}^I_{\mathcal{V}'}(\boldsymbol{x}^\star))^\top \boldsymbol{y}^{I,\star}_{\mathcal{V}'}$$

$$= \nabla_{x_i} \mathcal{L}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}), \tag{6.7c}$$

where the second equality follows from the definitions in (6.3) and the observation that the constraints in $\mathcal{V} \setminus \mathcal{N}_\mathcal{G}[i]$ are not dependent on x_i, and the last equality follows from the observation that the derivatives associated with the quadratic penalty term disappears due to Equation (6.5b). This and (6.5) imply that:

$$\nabla_{x_i} \mathcal{L}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = 0, \ i \in \mathcal{V}' \tag{6.8a}$$

$$c_i^E(\boldsymbol{x}^\star) = 0, \ i \in \mathcal{V}'' \tag{6.8b}$$

$$c_i^I(\boldsymbol{x}^\star) \geq 0, \ i \in \mathcal{V}' \tag{6.8c}$$

$$y_i^{I,\star} \geq 0,\ i \in \mathcal{V}' \tag{6.8d}$$

$$\operatorname{diag}(y_i^{I,\star}) c_i^I(\boldsymbol{x}) = 0,\ i \in \mathcal{V}'. \tag{6.8e}$$

Thus, the first-order conditions for $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ are satisfied at $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$. □

Lemma 6.15 establishes the desired consistency result. This result reveals the importance of incorporating the dual variable in the subproblem specification. In particular, in (6.7), the dualized coupled equality constraints allow the satisfaction of the first-order optimality conditions for the subproblem. The incorporation of the dualized constraints in the context of Jacobi/Gauss-Seidel type decomposition methods for constrained optimization problems has been studied in [144, 145, 150, 170].

6.1.3 Inheritance of Uniform Regularity

We now aim to show the inheritance of uniform regularity; that is, if the full problem $\mathcal{P}$ satisfies uniform regularity conditions, the subproblem $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ inherits the uniform regularity conditions. The following theorem establishes such a result.

Lemma 6.16. *Under Assumption 2.1, 3.4, 3.5, 3.6 for $\mathcal{P}$ at $\boldsymbol{z}^\star$, for any $\mu \geq \overline{\mu}$ (defined in (3.5)) and $\mathcal{V}', \mathcal{V}'' \subseteq \mathcal{V}$ such that $\mathcal{N}_{\mathcal{G}}[\mathcal{V}''] \subseteq \mathcal{V}'$, $(L+\mu L^2)$-BLH, β-uLICQ, and $(\gamma/2)$-uSSOSC for $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ hold at $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$.*

Proof of $L+\mu L^2$-BLH. By inspecting (6.6), one can see that the primal Hessian of Lagrangian $\mathcal{L}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''})$ of the subproblem $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ can be expressed by:

$$\begin{aligned}
&\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}} \mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) \qquad (6.9a)\\
&= \nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}} \mathcal{L}(\boldsymbol{z}^\star) + \mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}} \boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star))^\top (\nabla_{\boldsymbol{x}_{\mathcal{V}'}} \boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)).\\
&\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'}} \mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) \qquad (6.9b)\\
&= \nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'}} \mathcal{L}(\boldsymbol{z}^\star) + \mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}} \boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star))^\top (\nabla_{\boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'}} \boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)).
\end{aligned}$$

Here, note that the terms that contain the second derivatives of $\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\cdot)$ disappears due to $\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star) = 0$. Furthermore, by inspecting (6.4), we can observe that:

$$\nabla^2_{\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}\boldsymbol{y}^E_{\mathcal{V}''}} \mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}; \boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = \nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{y}^E_{\mathcal{V}''}} \mathcal{L}(\boldsymbol{z}^\star) \tag{6.10a}$$

$$\nabla^2_{\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}\boldsymbol{y}^I_{\mathcal{V}'}}\mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''};\boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = \nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{y}^I_{\mathcal{V}''}}\mathcal{L}(\boldsymbol{z}^\star) \tag{6.10b}$$

$$\nabla^2_{\boldsymbol{z}_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}\boldsymbol{y}^E_{\mathcal{V}''}}\mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''};\boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = \nabla^2_{\boldsymbol{z}_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}\boldsymbol{y}^E_{\mathcal{V}''}}\mathcal{L}(\boldsymbol{z}^\star) \tag{6.10c}$$

$$\nabla^2_{\boldsymbol{z}_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}\boldsymbol{y}^I_{\mathcal{V}'}}\mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''};\boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}) = \nabla^2_{\boldsymbol{z}_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}\boldsymbol{y}^I_{\mathcal{V}'}}\mathcal{L}(\boldsymbol{z}^\star) \tag{6.10d}$$

Moreover, we have from the L-uBLH of $\mathcal{P}$ and the fact that the constraint Jacobian matrices $\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)$ and $\nabla_{\boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)$ are the submatrices of $\nabla_{\boldsymbol{z}\boldsymbol{z}}\mathcal{L}(\boldsymbol{z}^\star)$, we have:

$$\|\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)\| \le L \tag{6.11a}$$

$$\|\nabla_{\boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_\mathcal{G}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star)\| \le L. \tag{6.11b}$$

From (6.9), (6.10), and (6.11), we have $\|\nabla^2_{\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}}\mathcal{L}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''};\boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''})\| \le L + \mu L^2$. □

Proof of β-uLICQ. By inspecting the problem formulation, one can see that the constraint Jacobian of $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ is a submatrix of the constraint Jacobian for the full problem $\mathcal{P}$, and the associated block row in the constraint Jacobian of $\mathcal{P}$ is zero. This implies that the smallest non-trivial singular value of the constraint Jacobian of $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ is bounded below by that of $\mathcal{P}$. This and β-uLICQ of $\mathcal{P}$ at $\boldsymbol{z}^\star$ imply β-uLICQ of $\mathcal{P}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ at $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$. □

Proof of $(\gamma/2)$-uSSOSC. From γ-uSSOSC of $\mathcal{P}$ and Lemma 3.6, we have:

$$\nabla^2_{\boldsymbol{x}\boldsymbol{x}}\mathcal{L}(\boldsymbol{z}^\star) + \overline{\mu}(\nabla_{\boldsymbol{x}}\boldsymbol{c}^1(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}}\boldsymbol{c}^1(\boldsymbol{x}^\star) \succeq (\gamma/2)I, \tag{6.12}$$

where $\boldsymbol{c}^1(\boldsymbol{x}) = \boldsymbol{c}(\boldsymbol{x})[\mathcal{A}^1]$, and $\mathcal{A}^1$ is the set of constraint indices that are either equalities or inequalities with nonzero dual at the solution $\boldsymbol{x}^\star$. This implies:

$$\nabla^2_{\boldsymbol{x}\boldsymbol{x}}\mathcal{L}(\boldsymbol{z}^\star) + \overline{\mu}(\nabla_{\boldsymbol{x}}\boldsymbol{c}^1(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}}\boldsymbol{c}^1(\boldsymbol{x}^\star) \succeq (\gamma/2)I.$$

for any $\mu \ge \overline{\mu}$. From this we can obtain:

$$\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}}\mathcal{L}(\boldsymbol{z}^\star) + \mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^1(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^1(\boldsymbol{x}^\star) \succeq (\gamma/2)I,$$

since the smallest eigenvalue of the diagonal submatrix of a positive definite matrix is always greater than or equal to that of the original postidive definite matrix. Next, we observe that:

$$\begin{aligned}
&ReH\left(\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}}\mathcal{L}(\boldsymbol{z}^\star)+\mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^1(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^1(\boldsymbol{x}^\star),[\boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}^\star);\boldsymbol{c}^{I1}_{\mathcal{V}'}(\boldsymbol{x}^\star)]\right)\\
&=ReH\left(\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}}\mathcal{L}(\boldsymbol{z}^\star)+\mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star),[\boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}^\star);\boldsymbol{c}^{I1}_{\mathcal{V}'}(\boldsymbol{x}^\star)]\right),
\end{aligned}$$

where $\boldsymbol{c}^{I1}_{\mathcal{V}'}(\cdot)$ denotes the constraints in $\boldsymbol{c}^I(\cdot)$ with non-zero active duals at $\boldsymbol{y}^{I,\star}$. This follows from the fact that

$$ReH(G+J_1^\top J_1,[J_1;J_2])=ReH(G,[J_1;J_2]).$$

Moreover, (6.9) implies:

$$\begin{aligned}
&ReH\left(\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}}\mathcal{L}(\boldsymbol{z}^\star)+\mu(\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star))^\top\nabla_{\boldsymbol{x}_{\mathcal{V}'}}\boldsymbol{c}^E_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}']\setminus\mathcal{V}''}(\boldsymbol{x}^\star),[\boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}^\star);\boldsymbol{c}^{I1}_{\mathcal{V}'}(\boldsymbol{x}^\star)]\right)\\
&=ReH\left(\nabla^2_{\boldsymbol{x}_{\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'}}\mathcal{L}^\mu_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''};\boldsymbol{z}^\star_{\mathcal{V}\setminus\mathcal{V}',\mathcal{V}\setminus\mathcal{V}''}),[\boldsymbol{c}^E_{\mathcal{V}''}(\boldsymbol{x}^\star);\boldsymbol{c}^{I1}_{\mathcal{V}'}(\boldsymbol{x}^\star)]\right),
\end{aligned}$$

This implies that the smallest eigenvalue of the reduced Hessian of $\mathcal{P}^\mu_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ at $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$ is lower bounded by the smallest eigenvalue of the left-hand-side of (6.12). Therefore, we have $(\gamma/2)$-uSSOSC. □

6.1.4 Subproblem Sensitivity

Now that we know the subproblems of the form (6.4) satisfy the uniform regularity conditions, we can establish the EDS for the subproblems with respect to the fixed solution data. First, the uniform regularity of the subproblems, established in Lemma 6.16, allows establishing the uniform boundedness conditions (3.1) for the regularized subproblems.

Lemma 6.17. *Under Assumption 2.1, 3.4, 3.5, 3.6 for $\mathcal{P}$ at $\boldsymbol{z}^\star$, for any $\mu\geq\overline{\mu}$ (defined in (6.16)) and hypergraph $\mathcal{V}',\mathcal{V}''\subseteq\mathcal{V}$, the following holds for the subproblem $\mathcal{P}^\mu_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ at solution $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$:*

$$\overline{\overline{\sigma}}_{\boldsymbol{H}_{\mathcal{V}',\mathcal{V}''}}(\boldsymbol{z}^\star)\leq L+\mu L^2, \tag{6.13a}$$

$$\overline{\overline{\sigma}}_{\boldsymbol{F}_{\mathcal{V}',\mathcal{V}''}}(\boldsymbol{z}^\star)\leq L+\mu L^2 \tag{6.13b}$$

$$\underline{\underline{\sigma}}_{\boldsymbol{H}_{\mathcal{V}',\mathcal{V}''}}(\boldsymbol{z}^\star)\geq\left(\frac{4}{\gamma}+\frac{64\mu(L+\mu L^2)^2}{\gamma^3\beta}+\frac{16(L+\mu L^2)}{\gamma^2\beta}\right)^{-1}\left(1+\mu(L+\mu L^2)\right)^{-1}. \tag{6.13c}$$

Proof. The result follows from Theorem 3.3 and Lemma 6.15, 6.16. □

We now are ready to state the EDS for subproblems.

Lemma 6.18. *Under Assumption 2.1, 3.4, 3.5, 3.6 for $\mathcal{P}$ at $\boldsymbol{z}^\star$, for any $\mu \geq \overline{\mu}$ (defined in (3.5)), $\mathcal{V}', \mathcal{V}'' \subseteq \mathcal{V}$ such that $\mathcal{N}_{\mathcal{G}}[\mathcal{V}'] \subseteq \mathcal{V}'$, and $\epsilon > 0$, there exist neighborhoods $\mathbb{Z}_{\mathcal{V}',\mathcal{V}''}$ of $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$ and $\mathbb{Z}$ of $\boldsymbol{z}^\star$ and a continuous function $\boldsymbol{z}^\dagger_{\mathcal{V}',\mathcal{V}''} : \mathbb{Z} \to \mathbb{Z}_{\mathcal{V}',\mathcal{V}''}$, such that for any $\boldsymbol{z} \in \mathbb{Z}$, $\boldsymbol{z}^\dagger_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z})$ is a primal-dual solution of $\mathcal{P}^\mu_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z})$. Furthermore, for any $\boldsymbol{z}, \boldsymbol{z}' \in \mathbb{Z}$ and $i \in \mathcal{V}'$,*

$$\left\| R_{i \leftarrow \mathcal{V}',\mathcal{V}''} \boldsymbol{z}^\dagger_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}) - R_{i \leftarrow \mathcal{V}',\mathcal{V}''} \boldsymbol{z}^\dagger_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}') \right\| \leq \sum_{j \in \mathcal{N}^2_{\mathcal{G}}[\mathcal{V}'] \setminus \mathcal{V}''} \Upsilon \rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1 \right\rceil_+} \| z_j - z'_j \|, \tag{6.14}$$

where Υ and ρ are defined in Theorem 2.2, $\overline{\sigma}_{\boldsymbol{H}}$, $\overline{\sigma}_{\boldsymbol{F}}$, $\underline{\sigma}_{\boldsymbol{H}}$ are defined in the right-hand-side of (6.13), and $R_{i \leftarrow \mathcal{V}',\mathcal{V}''}$ is the restriction of $\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}$ to z_i.

Proof. From Lemma 6.15, 6.16, $\boldsymbol{z}^\star_{\mathcal{V}',\mathcal{V}''}$ is a solution of $\mathcal{P}^\mu_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}^\star)$ at which $(L + \mu L^2)$-uBLH, β-uLICQ, $(\gamma/2)$-uSSOSC are satisifed. This observation and Theorem 2.2 and Lemma 6.17 indicates (6.14); here, one can observe that the data in $\mathcal{V} \setminus \mathcal{N}^2_{\mathcal{G}}[\mathcal{V}']$ has no impact on the problem; thus, it suffices to only express the sensitivity against the data in $\mathcal{N}^2_{\mathcal{G}}[\mathcal{V}'] \setminus \mathcal{V}''$. □

6.2 Algorithm

The analysis in the previous section shows that the subproblem in (6.4) is consistent with the full problem and it inherits the desired uniform regularity conditions. We now use this subproblem formulation to formally define the OSM and analyze the convergence of the algorithm.

To define the algorithm, we first need to define a set of subdomains (in particular, $\mathcal{V}'$ and $\mathcal{V}''$ that appear in (6.4)). We consider a *partition* $\{\mathcal{V}_k\}_{k \in \mathcal{K}}$ of $\mathcal{V}$: for any $k \in \mathcal{K}$ and $k' \neq k$,

$$\begin{aligned} \mathcal{V}_k &\subseteq \mathcal{V} \\ \dot{\bigcup}_{k \in \mathcal{K}} \mathcal{V}_k &= \mathcal{V}, \end{aligned}$$

where $\dot{\bigcup}$ denotes the disjoint union. Furthermore, we consider an *overlapping partition* $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$ such that:

$$\mathcal{V}_k \subseteq \mathcal{V}'_k, \mathcal{V}''_k \subseteq \mathcal{V} \tag{6.15a}$$

$$\mathcal{N}_{\mathcal{G}}[\mathcal{V}''_k] \subseteq \mathcal{V}'_k. \tag{6.15b}$$

We call $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$ a partition of $\mathcal{V}$ and $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$ overlapping partitions of $\mathcal{V}$. With the overlapping partitions $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$, one can formualte a set of subproblems $\{\mathcal{P}^{\mu}_{\mathcal{V}'_k,\mathcal{V}''_k}(\cdot)\}_{k\in\mathcal{K}}$. The non-overlapping and overlapping partitions $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$, $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$, and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$ are depicted in Figure 6.2

Constructing a set of overlapping and non-overlapping partition can be performed in a straightforward manner. First, one obtains the non-overlapping partition of the graph $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$. This can be done by the intuition of the modeler (e.g., in dynamic optimization, one can partition the graph by dividing the time horizon into intervals of equal lengths) or by using generic graph partitioning tools (e.g., METIS [103]). Then, each non-overlapping subdomains can be *expanded* to obtain the overlapping partitions. The expansion procedure is performed by progressively incorporating adjacent nodes. For example, each subdomains in the overlapping partitions can be obtained by:

$$\mathcal{V}''_k = \mathcal{N}^{\omega}[\mathcal{V}_k] \tag{6.16a}$$

$$\mathcal{V}'_k = \mathcal{N}^{\omega+1}[\mathcal{V}_k]. \tag{6.16b}$$

Here, note that a prescribed parameter ω is used to control the size of overlap; we will see later that the size of overlap becomes an important algorithmic parameter. The method in (6.16) is just one way of creating the overlapping partition, and different methods can be used as long as the requirements in (6.15) are satisfied.

We now are ready to formally define the OSM; the algorithm can be defined as:

$$\boldsymbol{z}^{(\ell+1)}_{\mathcal{V}_k} = R_{\mathcal{V}_k \leftarrow \mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^{\dagger}_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}^{(\ell)}),\ k \in \mathcal{K},\ \ell = 0, 1, \cdots, \tag{6.17}$$

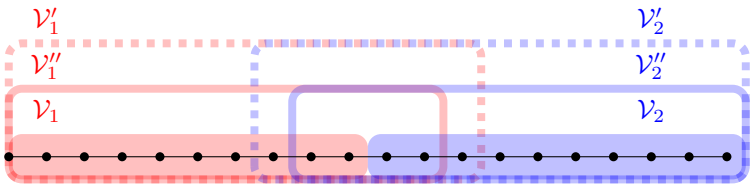

Figure 6.2 Non-overlapping and overlapping partitions

where we follow the notation in Lemma 6.18 and we denote the primal-dual solution at iteration ℓ as $\boldsymbol{z}^{(\ell)}$. Note that yet one cannot guarantee that the recursion in (6.17) is well-defined for $\ell = 0, 1, \cdots$. This is because we do not know yet that $\boldsymbol{z}^{(\ell)}$ is in the domain of the local solution mapping $\boldsymbol{z}^\dagger(\cdot)$. We will show later that under a certain condition this requirement can be satisfied for any $\ell = 0, 1, \cdots$.

Each iterate of the algorithm consists of two steps: subproblem solution and solution restriction. In the first step, one formulates the *subproblem* for the k-th subdomain as $\mathcal{P}^\mu_{\mathcal{V}_k', \mathcal{V}_k''}(\boldsymbol{z}^{(\ell)})$. The subproblem incorporates the primal-dual solution of the previous iteration step. Then, the subproblem is solved to obtain its solution $\boldsymbol{z}^\dagger_{\mathcal{V}_k', \mathcal{V}_k''}(\boldsymbol{z}^{(\ell)})$. Here, we observe that solution multiplicity exists at the overlapping region. In particular, if a certain node is associated with more than one subdomains in $\{\mathcal{V}_k'\}_{k\in\mathcal{K}}$, different subproblems may produce the solution for the same node, but they are not necessarily the same. To remove such multiplicity, we *restrict* the solution.[1] Specifically, we discard the primal solutions associated with $\mathcal{V} \setminus \mathcal{V}_k'$ and the dual solutions associated with $\mathcal{V} \setminus \mathcal{V}_k''$ and take only those solutions associated with $\mathcal{V}_k$. This procedure is represented by the restriction operator $R_{\mathcal{V}_k \leftarrow \mathcal{V}_k', \mathcal{V}_k''}$. After restriction, the solutions are assembled over $k \in \mathcal{K}$ to make the next guess of the solution $\boldsymbol{z}^{(\ell+1)}$. This concludes one iterate of overlapping Schwarz, and the algorithm is repeated until certain stopping criteria are met. One can implement stopping criteria based on the residuals to the KKT conditions.

[1]The term *restriciton* is originated from the restricted additive/multiplicative Schwarz method for sparse linear algebra [29]

6.3 Convergence

We now analyze the convergence of OSM (6.17). Our convergence analysis derives an explicit upper bound of the linear convergence rate. The convergence rate is expressed in terms of two parameters that characterize the overlapping/non-overlapping partitions. In particular, those parameters are the *size of overlap* and the *size of boundary*. We will see that the convergence rate improves exponentially with the size of overlap and deteriorates linearly with the size of boundaries. However, we will see that for mesh-like graphs, the exponential effect of the size of overlap dominates the linear effect of the size of boundaries. In what follows, we first formally define the parameters that characterize the partitions (the size of overlap and the size of boundary); then, we explain the convergence without mathematical details but with an illustrative example; finally, we formally prove the convergence.

6.3.1 Characterization of Partitions

We now define two parameters that characterize the given overlapping and non-overlapping partition, which will be crucial for the convergence analysis.

Definition 6.6. *The size of overlap ω for a pair of overlapping partition $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$ and the non-overlapping partition $\{\mathcal{V}_k''\}_{k\in\mathcal{K}}$ is defined by:*

$$\omega := \min_{k\in\mathcal{K}} d_{\mathcal{G}}(\mathcal{V}_k, \mathcal{V} \setminus \mathcal{V}_k'') - 1.$$

Observe that if $\mathcal{V}_k'' = \mathcal{V}_k$ for all $\mathcal{V}_k$, the size of overlap is zero. Also, note that the expansion approach in (6.16) indeed constructs overlapping partitions with the size of overlap ω.

We now define the second parameter, size of boundary.

Definition 6.7. *The size of boundary ψ for non-overlapping partitions $\{\mathcal{V}_k''\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}_k'\}_{k\in\mathcal{K}}$ is defined by:*

$$\psi := \max_{k\in\mathcal{K}} |\mathcal{N}_{\mathcal{G}}^2[\mathcal{V}_k'] \setminus \mathcal{V}_k''|.$$

Note that if the overlapping partitions are obtained based on the approach in (6.16), the size of boundary is equal to:

$$\psi = \max_{k \in \mathcal{K}} |\mathcal{N}_{\mathcal{G}}^{\omega+3}[\mathcal{V}_k] \setminus \mathcal{N}_{\mathcal{G}}^{\omega}[\mathcal{V}_k]| \tag{6.18}$$

6.3.2 Convergence in a Nutshell

Before we formally prove the convergence of the algorithm, we first provide high-level intuition on the convergence behavior. Figure 6.3 illustrates how EDS enables and accelerates the convergence. Here, we consider a 20-node dynamic optimization problem. For the example on the left, we apply OSM and for the example on the right, we apply a non-overlapping decomposition (block Jacobi method). The optimal solution is represented as a black dashed line, and the subproblem solutions are plotted as colored lines.

In the first iteration, both overlapping and non-overlapping decomposition needs to start the iteration with some arbitrary starting point. Due to this arbitrariness, there are significant errors in the subproblem solutions. However, one interesting observation is that as we move into the inner part of each subdomain, the solution accuracy improves. Specifically, the difference between the optimal solution and the subproblem solutions becomes small as moving towards the interior of each subdomain. This observation can be accounted for by the EDS; the solution from the neighboring subproblem enters into each subproblem as parametric perturbations, and the impact of such perturbation decays exponentially as moving away from the perturbation point. The magic of OSM is happening during the next phase: restriction. In particular, during this phase, the low-quality boundary part is discarded, and the high-quality inner parts are taken to the next iteration. So, this restriction procedure is designed to exploit the EDS. On the other hand, for non-overlapping decomposition, due to the absence of overlap, the low-quality boundary part still remains in the solution, and this adversely affects the solution quality in the next iteration step. In the next iteration, in OSM, thanks to the availability of the high-quality solution guess, the solutions can quickly close the gap from the optimal solution. But non-overlapping decomposition cannot make much progress due to the absence of such a high-quality solution guess. If we repeat this procedure

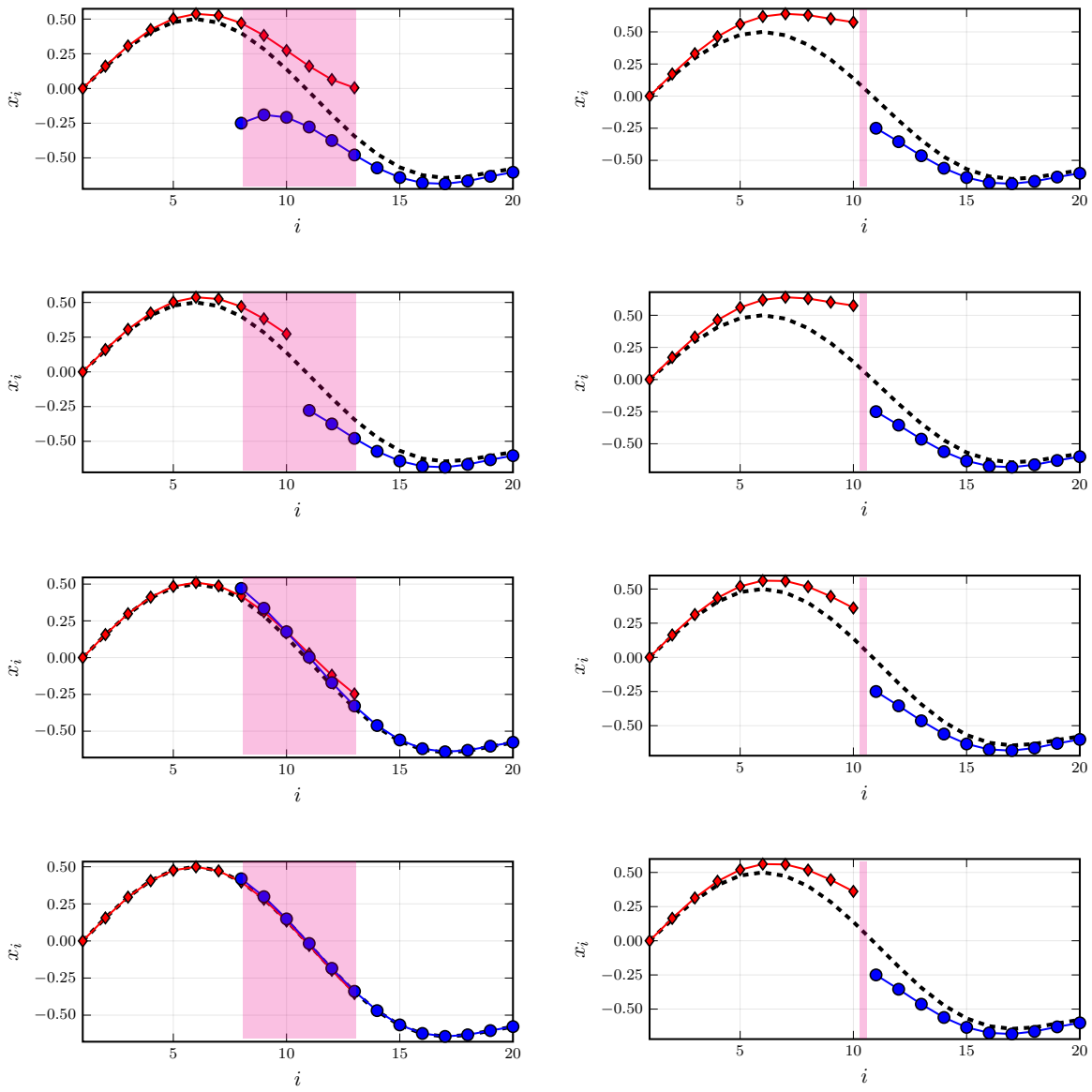

Figure 6.3 Illustration of the convergence of overlapping Schwarz method. Overlapping Schwarz method (left) and block Jacobi method (right). The solutions from the first iterate (top); the solutions from the first iterate, after the restriction (second); the solutions from the second iterate (third); The solutions from the third iterate (bottom).

once more, OSM converges to almost a perfect solution, while the non-overlapping method barely makes progress. This example illustrates how overlap enables fast convergence by exploiting the EDS.

6.3.3 Convergence Analysis

We now formally prove the convergence of the Schwarz algorithm. We will see that parametric sensitivity plays a central role in convergence behavior. In particular, by using the sensitivity result obtained in Lemma 6.18, we establish the convergence of OSM (6.17).

Theorem 6.5 (Convergence)**.** *Under Assumption 2.1, 3.4, 3.5, 3.6, given any non-overlppinag partition* $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$ *and overlapping partitions* $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$ *and* $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$ *satisfying (6.15),* $\mu \geq \overline{\mu}$ *(defined in (3.5)), and* $\epsilon > 0$*, there exists a neighbhorhood* $\mathbb{Z}$ *of* $\boldsymbol{z}^\star$ *such that the OSM (6.17) is well-defined if* $\omega \geq 4(1 + \log_{1/\rho}(\psi\Upsilon))$ *and* $\boldsymbol{z}^{(0)} \in \mathbb{Z}$*. Furthermore, the sequence* $\{\boldsymbol{z}^{(\ell)}\}_{\ell=0}^{\infty}$ *generated by OSM in (6.17) satisfies:*

$$\max_{i\in\mathcal{V}} \|z_i^{(\ell)} - z_i^\star\| \leq \alpha^\ell \max_{i\in\mathcal{V}} \|z_i^{(0)} - z_i^\star\|, \tag{6.19}$$

where $\alpha := \psi\Upsilon\rho^{\lceil\omega/4-1\rceil_+}$*,* Υ *and* ρ *are defined in Lemma 6.18,* ω *is the size of overlap, and* ψ *is the size of boundary.*

Proof. One can show that there exists $r > 0$ such that if $\boldsymbol{z} \in \mathbb{Z}$ for

$$\mathbb{Z} := \{\boldsymbol{z} : \|z_i - z_i^\star\| \leq r \text{ for any } i \in \mathcal{V}\},$$

we always have $\boldsymbol{z} \in \mathbb{Z}_k$ for any $k \in \mathcal{K}$, where $\mathbb{Z}_k$ is the $\mathbb{Z}$ in Lemma 6.18 for the subproblem index k. By the definition of the size of overlap and boundary, one can see that for any $i \in \mathcal{V}_k$ and $j \in \mathcal{N}_{\mathcal{G}}^2[\mathcal{V}'_k] \setminus \mathcal{V}'_k$,

$$d_{\mathcal{G}}(i, j) \geq \omega + 1.$$

Therefore, from Lemma 6.18 we have that for any $k \in \mathcal{K}$ and $i \in \mathcal{V}_k$,

$$\left\| R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^\dagger_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}) - R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^\dagger_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}') \right\| \leq \psi\Upsilon\rho^{\lceil\omega/4-1\rceil_+} \max_{j\in\mathcal{N}^2[\mathcal{V}'_k]\setminus\mathcal{V}''_k} \|z_j - z'_j\|. \tag{6.20}$$

We can observe that for $\omega \geq 4(1 + \log_{1/\rho}(\psi\Upsilon))$, we have that

$$\left\| R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^\dagger_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}) - R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^\dagger_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}') \right\| \leq \max_{j\in\mathcal{N}^2[\mathcal{V}'_k]\setminus\mathcal{V}''_k} \|z_j - z'_j\|. \tag{6.21}$$

Thus, by (6.21) and the fact that:

$$\boldsymbol{z}^{\dagger}_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}^\star) = \boldsymbol{z}^\star_{\mathcal{V}'_k,\mathcal{V}''_k},$$

if $\boldsymbol{z}^{(\ell)} \in \mathbb{Z}$, we have $\boldsymbol{z}^{(\ell+1)} \in \mathbb{Z}$. Therefore, the sequence $\{\boldsymbol{z}^{(\ell)}\}_{\ell=0}^{\infty}$ is well-defined. Now that we know $\boldsymbol{z}^{(0)}, \boldsymbol{z}^{(1)}, \cdots \in \mathbb{Z}$, we can apply (6.20) to each iterate, each $k \in \mathcal{K}$, and each $i \in \mathcal{V}_k$. This yields:

$$\max_{i\in\mathcal{V}} \|z_i^{(\ell+1)} - z_i^\star\| \leq \alpha \max_{i\in\mathcal{V}} \|z_i^{(\ell)} - z_i^\star\|. \tag{6.22}$$

The convergence result (6.19) can be obtained from (6.22). □

Theorem 6.5 indicates that the rate of convergence improves exponentially with the size of overlap ω and deteriorates linearly with the size of boundary ψ. Thus, the relationship between ψ and ω plays an important role in guaranteeing the convergence of the algorithm. Based on the topology of the graph, one can roughly estimate how fast the size of boundary grows with respect to the size of overlap (assuming that the overlapping partitions are constructed by (6.16); thus, the size of boundary is $\psi = \max_{k\in\mathcal{K}} |\mathcal{N}_{\mathcal{G}}^{\omega+3}[\mathcal{V}_k] \setminus \mathcal{N}_{\mathcal{G}}^{\omega}[\mathcal{V}_k]|$). For example, if the graph is linear (e.g., dynamic optimization or PDE-based optimization on 1-D space), the size of boundary is always less than or equal to 6 even when the size of overlap grows. When the graph is an N-dimensional mesh, the size of boundaries grows polynomially with the size of overlap. When the graph is a tree, the size of boundaries grows exponentially with the size of overlap.

Therefore, we can say that when the graph topology is similar to a mesh in a finite-dimensional space, if the problem satisfies uniform regularity condition, the rate of convergence improves exponentially with the size of overlap. This is because even if ψ grows polynomially, the exponential decay of $\rho^{\omega/4}$ eventually outweighs the polynomial growth. On the other hand, if the graph is a tree, the convergence rate may not decay with ω as ψ also grows exponentially in ω.

In many practical instances, however, such analysis of the limiting behavior may not be relevant because the size of the graph may not be sufficiently large to allow observing such

a limiting behavior. In such a case, even for a moderate size of overlap ω, each subproblem may become the full problem. If each subproblem becomes the full problem, the algorithm will trivially converge to the solution in one iteration step. In other words, the results in Theorem 6.5 become nontrivial only if the gsNLPs are sufficiently large (in diameter).

The key observation in the proof is that after applying the restriction operator $R_{\mathcal{V}\leftarrow\mathcal{V}',\mathcal{V}''}$, the solution can always enjoy the exponential decrease in the error introduced by the error in the solution guess. Equation (6.20) reveals such an observation. In other words, this restriction step discards the "bad" part of the solution, which is strongly affected by the error in the solution guess, and only takes the good part of the solution, where such adverse effect is adequately damped out. This indicates that the restriction step is designed to exploit the EDS.

While Theorem 6.5 implies that the convergence rate of the algorithm improves exponentially with the size of overlap for a certain class of graphs, it does not mean that the solution time decreases as the size of overlap is increased. In practice, the solution time is affected both by the convergence rate and the solution time for each subproblem. We can observe that as we increase the size of overlap, the subproblem size also increases (assuming that the overlapping partitions are constructed by (6.16)). Thus, as we increase ω, each iteration may take a longer time. Thus, there exists a trade-off between the size of overlap and subproblem complexity. In practice, one needs to tune the size of overlap to achieve the best performance. If one has a complexity model for the subproblems, one can use the result in Theorem 6.5 to compute the optimal size of overlap. However, such a complexity model can be vastly different from problem to problem, so here we do not study finding the optimal overlap size.

The global convergence of the OSM might be of interest; unfortunately, studying the global convergence of the OSM is very challenging under nonconvex settings because our local convergence analysis entirely relies on the local sensitivity of the solution, and there is no global convergence metric such as merit functions. Thus, to enforce the global convergence,

the algorithm may need to be supplemented with a certain line-search or trust-region-based strategy.

Alternatively, one may consider a strategy where we use globally-convergent nonlinear optimization algorithms (e.g., augmented Lagrangian method (ALM), sequential quadratic programming (SQP) method, and interior point method (IPM) with line-search or trust-region strategies [124]) and apply OSM to solve the subproblems (e.g., QP subproblems or linear systems) that appears within such globally convergent methods. This motivates us to study the convergence of OSM under simpler settings, in particular, quadratic programming and linear systems, and seek to establish global convergence. In the next two chapters, we will discuss the specialization of OSM (6.17) for quadratic programming and linear systems and study the global convergence properties.

Chapter 7

Quadratic Programming

In this chapter, we discuss OSM for graph-structured quadratic programs (gsQPs). Since quadratic programs are nonlinear programs, the local convergence results in Section 6.3 holds for gsQPs. However, by exploiting the nature of quadratic programs, one can establish a stronger convergence result. In particular, in this section, we aim to establish the global convergence of the OSM under stronger regularity assumptions: global SSOSC and global LICQ. Note that linear systems are also quadratic programs in that they can be treated as QPs; for example, positive definite systems can be regarded as unconstrained QPs and non-symmetric systems can be regarded as an equality-constrained QPs with constant objective functions. We discuss the OSM for linear systems further in the next chapter.

Related Work: The convergence of OSM for solving quadratic programs has been studied in our previous works [145, 153]. In particular, we have studied the convergence behavior under unconstrained QP setting in [153] and that under constrained QP setting in [145]. However, as in [145], our subproblem definition did not take into account the inheritance of uniform regularity; as such, we needed to rely on a conjecture on the uniform boundedness of the singular value of the KKT matrix [145, Assumption 4]. Furthermore, we could only establish the local convergence based on such a conjecture. The result in the current dissertation extends this result in that we deal with constrained QPs and we do not make such an ad-hoc conjecture. Furthermore, we establish global convergence.

In the context of (6.1), we consider the following quadratic programming setting:

$$f_i(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) := \sum_{j\in\mathcal{N}_\mathcal{G}[i]} \frac{1}{2} x_i^\top G_{i,j} x_j - g_i^\top x_i,\ i \in \mathcal{V} \tag{7.1a}$$

$$c_i^E(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) := \sum_{j\in\mathcal{N}_\mathcal{G}[i]} \frac{1}{2} J_{i,j}^E x_j - h_i^E,\ i\in\mathcal{V} \tag{7.1b}$$

$$c_i^I(\{x_j\}_{j\in\mathcal{N}_\mathcal{G}[i]}) := \begin{bmatrix} J_{i,i}^I \\ -J_{i,i}^I \end{bmatrix} x_i - \begin{bmatrix} h_i^L \\ -h_i^U \end{bmatrix},\ i\in\mathcal{V}, \tag{7.1c}$$

Here, $G_{i,j}\in\mathbb{R}^{n_{x_i}\times n_{x_j}}$, $J_{i,j}^E\in\mathbb{R}^{n_{y_i^E}\times n_{x_j}}$, $J_{i,j}^I\in\mathbb{R}^{n_{y_i^I}\times n_{x_j}}$, $g_i\in\mathbb{R}^{n_{x_i}}$, $h_i^E\in\mathbb{R}^{n_{y_i^E}}$, and $h_i^L, h_i^U\in\mathbb{R}^{n_{y_i^I}}$. We assume that :

$$G_{i,j} = G_{j,i}^\top,\ i,j\in\mathcal{V} \tag{7.2a}$$

$$h_i^U > h_i^L,\ i\in\mathcal{V}. \tag{7.2b}$$

With the definitions in (7.1), the gsQP can be formulated as follows:

$$\mathcal{Q}:\ \min_{\{x_i\}_{i\in\mathcal{V}}} \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_\mathcal{G}[i]} \frac{1}{2} x_i^\top G_{i,j} x_j - \sum_{i\in\mathcal{V}} g_i^\top x_i \tag{7.3a}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}_\mathcal{G}[i]} J_{i,j}^E x_j = h_i^E,\ (y_i^E),\ i\in\mathcal{V} \tag{7.3b}$$

$$h_i^L \le J_{i,i}^I x_i \le h_i^U,\ (y_i^I),\ i\in\mathcal{V}. \tag{7.3c}$$

We define $J_{i,j} := [J_{i,j}^E; J_{i,j}^I]$, $H_{i,j} := [G_{i,j}\ \ J_{i,j}^\top; J_{i,j}\ \ 0]$, $y_i := [y_i^E; y_i^I]$, $z_i := [x_i; y_i]$, $n_{y_i} := n_{y_i^E} + n_{y_i^I}$, and $n_{z_i} := n_{x_i} + n_{y_i}$; here, we assume $J_{i,j}^I = 0$ if $i\neq j$ for convenience. We denote the problem in (7.3) as $\mathcal{Q}$, and the subproblems (6.4) associated with subdomains $\mathcal{V}', \mathcal{V}''$, data $\overline{\boldsymbol{z}}$, and the penalty parameter μ is denoted by $\mathcal{Q}^\mu_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$. Note that it suffices to define one dual variable for each two-sided inequality in (7.3c); this is because if (7.2b) holds, due to the complemenatarity slackness, one of the dual variables is always zero. Thus, one can represent the duals for the upper and lower bound constraint in a single scalar value.

We define the following additional notation: for subsets $\mathcal{V}', \mathcal{V}''\subseteq\mathcal{V}$, $\boldsymbol{J}^E_{\mathcal{V}',\mathcal{V}''} := \{J_{i,j}^E\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$, $\boldsymbol{J}^I_{\mathcal{V}',\mathcal{V}''} := \{J_{i,j}^I\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$, $\boldsymbol{J}_{\mathcal{V}',\mathcal{V}''} := \{J_{i,j}\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$, $\boldsymbol{G}_{\mathcal{V}',\mathcal{V}''} := \{G_{i,j}\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$, $\boldsymbol{H}_{\mathcal{V}',\mathcal{V}''} := \{H_{i,j}\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$, $\boldsymbol{g}_{\mathcal{V}'} := \{g_i\}_{i\in\mathcal{V}'}$, $\boldsymbol{h}^E_{\mathcal{V}'} := \{h_i^E\}_{i\in\mathcal{V}'}$, $\boldsymbol{h}^L_{\mathcal{V}'} := \{h_i^L\}_{i\in\mathcal{V}'}$, and $\boldsymbol{h}^U_{\mathcal{V}'} := \{h_i^U\}_{i\in\mathcal{V}'}$, where we assume that $G_{i,j}^E = 0$ and $J_{i,j}^E = 0$ if $d_\mathcal{G}(i,j) > 2$ and $J_{i,j}^I = 0$ if $i\neq j$. Also, $\boldsymbol{J}^E := \boldsymbol{J}^E_{\mathcal{V},\mathcal{V}}, \boldsymbol{J}^I := \boldsymbol{J}^I_{\mathcal{V},\mathcal{V}}, \boldsymbol{J} := \boldsymbol{J}_{\mathcal{V},\mathcal{V}}, \boldsymbol{G} := \boldsymbol{G}_{\mathcal{V},\mathcal{V}}, \boldsymbol{H} := \boldsymbol{H}_{\mathcal{V},\mathcal{V}}, \boldsymbol{h} := \boldsymbol{h}_\mathcal{V}, \boldsymbol{h}^L := \boldsymbol{h}^L_\mathcal{V}$, and $\boldsymbol{h}^U := \boldsymbol{h}^U_\mathcal{V}$.

7.1 Global Regularity Conditions

To establish the global convergence, we strengthen the uniform regularity assumptions (Assumption 3.4, 3.5, 3.6). These strengthened regularity conditions are only relevant to gsQP settings and cannot be generalized to gsNLPs because they are only relevant if the Lagrangian Hessian and constraint Jacobian are constant. Thus, the subsequent result exploits the particular structure in gsQP. In what follows, we define globally bounded Lagrangian Hessian (gBLH), global strong second-order sufficiency conditions (gSSOSC), and global linear independence constraint qualifications (gLICQ).

Assumption 7.13 (L-gBLH)**.** $\|\boldsymbol{H}\| \leq L$.

Assumption 7.14 (γ-gSSOSC)**.** $ReH(\boldsymbol{G}, \boldsymbol{J}^E) \succeq \gamma \boldsymbol{I}$ *for* $\gamma > 0$.

Assumption 7.15 (β-gLICQ)**.** $\boldsymbol{J}\boldsymbol{J}^\top \succeq \beta \boldsymbol{I}$ *for* $\beta > 0$.

One can see that the global regularity conditions are certainly more restrictive than the uniform regularity conditions. This is because we enforce that the reduced Hessian is positive definite with a uniformly bounded smallest eigenvalue *in the absence of active inequality constraints*, and we enforce that the constraint Jacobian is linearly independent with a uniformly bounded smallest singular value *even when all the inequality constraints are active*. This is different from uniform regularity setting, which only enforce those conditions for particular inequality sets (recall Assumption 3.5, 3.6). In the absence of inequality constraints, the global regularity condition reduces to the normal uniform regularity condition.

Note that we can apply Lemma 6.15 (consitency of the subproblems) to the subproblems of $\mathcal{Q}$; thus, we do not need to prove it again. We now show that under Assumption 7.13, 7.14, 7.15, any subproblem $\mathcal{Q}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\cdot)$ of $\mathcal{Q}$ has a solution at which uniform regularity conditions are satisified (thus the solution is unique) with the common uniform parameters. This replaces Lemma 6.16 (inheritance of uniform regularity).

Lemma 7.19. *Under Assumption 7.13, 7.14, 7.15 for a feasible $\mathcal{Q}$ satisfying (7.2), for any $\mu \geq \overline{\mu}$ (defined in (3.5)), $\mathcal{V}', \mathcal{V}'' \subseteq \mathcal{V}$ satisfying $\mathcal{N}_{\mathcal{G}}[\mathcal{V}''] \subseteq \mathcal{V}'$, and any $\overline{\boldsymbol{z}}$, there exists a solution of $\mathcal{Q}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ at which $L + \mu L^2$-uBLH, $(\gamma/2)$-uSSOSC, and β-uLICQ holds.*

Proof. The feasibility, γ-gSSOSC, and β-gLICQ implies that the full problem has a unique primal-dual solution at which L-uBLH and γ-uSSOSC holds. Furthermore, the Jacobian of constraints with nonzero dual is a submatrix of $\boldsymbol{J}$ constructed by collecting a subset of the rows of $\boldsymbol{J}$. Thus, we also have β-uLICQ. Since the original problem is feasible, the subproblem $\mathcal{Q}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ is also feasible; this is because the set of constraints in $\mathcal{Q}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ is only a subset of the set of constraints in $\mathcal{Q}$; thus, if $\boldsymbol{x}$ is feasible for $\mathcal{Q}$, we have:

$$\boldsymbol{J}^{E}_{\mathcal{V}'',\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'} = \boldsymbol{h}^{E}_{\mathcal{V}''}$$
$$\boldsymbol{h}^{L}_{\mathcal{V}',\mathcal{V}'} \leq \boldsymbol{J}^{I}_{\mathcal{V}',\mathcal{V}'}\boldsymbol{x}_{\mathcal{V}'} \leq \boldsymbol{h}^{U}_{\mathcal{V}'}.$$

By Lemma 3.6, we have that $\boldsymbol{G} + \mu(\boldsymbol{J}^{E})^{\top}\boldsymbol{J}^{E} \succeq (\gamma/2)\boldsymbol{I}$. By extracting the submatrix associated with $\mathcal{V}'$ and noting that $\boldsymbol{J}^{E}_{\mathcal{V}',\mathcal{V}\setminus\mathcal{N}_{\mathcal{G}}[\mathcal{V}']} = \boldsymbol{0}$, we obtain:

$$\boldsymbol{G}_{\mathcal{V}',\mathcal{V}'} + \mu(\boldsymbol{J}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}'],\mathcal{V}'})^{\top}\boldsymbol{J}^{E}_{\mathcal{N}_{\mathcal{G}}[\mathcal{V}'],\mathcal{V}'} \succeq (\gamma/2)\boldsymbol{I};$$

and finally we have

$$ReH(\boldsymbol{G}_{\mathcal{V}',\mathcal{V}'} + \mu(\boldsymbol{J}^{E}_{\mathcal{N}[\mathcal{V}']\setminus\mathcal{V}'',\mathcal{V}'})^{\top}\boldsymbol{J}^{E}_{\mathcal{N}[\mathcal{V}']\setminus\mathcal{V}'',\mathcal{V}'}, \boldsymbol{J}^{E}_{\mathcal{V}'',\mathcal{V}'}) \succeq (\gamma/2)\boldsymbol{I}.$$

Therefore, the subproblem is strongly convex, and there exists a unique solution. This implies that $(\gamma/2)$-uSSOSC holds at the soluiton. Moreover, L-gBLH implies $L + \mu L^2$-uBLH at the soluiton (here, μL^2 term comes from the quadratic penalty). For such a unique solution, the constraint Jacobian is a submatrix of $\boldsymbol{J}$ whose associated block row is zero. Thus, β-uLICQ holds at the solution. □

The subproblems of globally regular gsQPs always satisfy the uniform regularity at the solution with common uniform constants, $L+\mu L^2$, $\gamma/2$, and β. Comparing Lemma 7.19 with Lemma 6.16, we observe that the twice continuous differentiability assumption (Assumption 2.1) is dropped since QPs always satisfy such an assumption.

7.2 Subproblem Sensitivity

We now establish the EDS for the subproblems of $\mathcal{Q}$. Here, since uBLH, uSSOSC, and uLICQ hold with uniform parameters that do not depend on the data $\overline{\boldsymbol{z}}$, one can establish *global* EDS. That is, as opposed to the local results in Theorem 7.6, the sensitivity result applies to any $\overline{\boldsymbol{z}} \in \mathbb{R}^{\boldsymbol{n_z}}$, and the decay parameters Υ, ρ do not depend on the value of $\overline{\boldsymbol{z}}$. The theorem is stated as follows.

Theorem 7.6 (EDS in Subproblems of $\mathcal{Q}$)**.** *Under Assumption 7.13, 7.14, 7.15 for $\mathcal{Q}$ satisfying (7.2), for any $\mu \geq \overline{\mu}$ (defined in 3.5), $\mathcal{V}', \mathcal{V}'' \subseteq \mathcal{V}$ satisfying $\mathcal{N}_{\mathcal{G}}[\mathcal{V}''] \subseteq \mathcal{V}'$, there exists a continuous function $\boldsymbol{z}^{\dagger}_{\mathcal{V}',\mathcal{V}''} : \mathbb{R}^{\boldsymbol{n_z}} \to \mathbb{R}^{\boldsymbol{n}_{\boldsymbol{z}_{\mathcal{V}',\mathcal{V}''}}}$, such that for any $\overline{\boldsymbol{z}}$, $\boldsymbol{z}^{\dagger}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$ is a primal-dual solution of $\mathcal{Q}^{\mu}_{\mathcal{V}',\mathcal{V}''}(\overline{\boldsymbol{z}})$. Furthermore, for any $\boldsymbol{z}, \boldsymbol{z}' \in \mathbb{Z}$,*

$$\left\| R_{i \leftarrow \mathcal{V}',\mathcal{V}''} \boldsymbol{z}^{\dagger}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}) - R_{i \leftarrow \mathcal{V}',\mathcal{V}''} \boldsymbol{z}^{\dagger}_{\mathcal{V}',\mathcal{V}''}(\boldsymbol{z}') \right\| \leq \sum_{j \in \mathcal{N}^2_{\mathcal{G}}[\mathcal{V}'] \setminus \mathcal{V}''} \Upsilon \rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4} - 1 \right\rceil_+} \|z_j - z'_j\|, \tag{7.4}$$

where $\Upsilon := \dfrac{\overline{\sigma}_{\boldsymbol{H}} \overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}^2_{\boldsymbol{H}}}$, $\rho := \dfrac{\overline{\sigma}^2_{\boldsymbol{H}} - \underline{\sigma}^2_{\boldsymbol{H}}}{\overline{\sigma}^2_{\boldsymbol{H}} + \underline{\sigma}^2_{\boldsymbol{H}}}$, *and*

$$\begin{aligned}
\overline{\sigma}_{\boldsymbol{H}} &:= L + \mu L^2 \\
\overline{\sigma}_{\boldsymbol{F}} &:= L + \mu L^2 \\
\underline{\sigma}_{\boldsymbol{H}} &:= \left(\frac{4}{\gamma} + \frac{64\mu(L+\mu L^2)^2}{\gamma^3 \beta} + \frac{16(L+\mu L^2)}{\gamma^2 \beta} \right)^{-1} \left(1 + \mu(L + \mu L^2)\right)^{-1}.
\end{aligned}$$

Proof. The existence of the solution mapping follows from Lemma 7.19, and its continuity can be obtained by applying Lemma 2.2 to each point $\overline{\boldsymbol{z}} \in \mathbb{R}^{\boldsymbol{n_z}}$. By applying Lemma 2.3, one can show that the directional derivative of the solution with respect to the data perturbation satisfies (2.20). By using $(L + \mu L^2)$-gBLH, $(\gamma/2)$-gSSOSC, β-gLICQ, establisehd in Lemma 7.19 and Lemma 6.17, one can show that for any data $\boldsymbol{p} = \overline{\boldsymbol{z}}$ and direction $\boldsymbol{q}$, we have:

$$\overline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p}, \boldsymbol{q}) \leq L + \mu L^2 \tag{7.5a}$$

$$\overline{\sigma}_{\boldsymbol{F}}(\boldsymbol{p}, \boldsymbol{q}) \leq L + \mu L^2 \tag{7.5b}$$

$$\underline{\sigma}_{\boldsymbol{H}}(\boldsymbol{p}, \boldsymbol{q}) \geq \left(\frac{4}{\gamma} + \frac{64\mu(L+\mu L^2)^2}{\gamma^3 \beta} + \frac{16(L+\mu L^2)}{\gamma^2 \beta} \right)^{-1} \left(1 + \mu(L + \mu L^2)\right)^{-1}. \tag{7.5c}$$

Thus, we can rewrite (2.20) as follows:

$$\|R_{i\leftarrow\mathcal{V}_k',\mathcal{V}_k''}D_{\boldsymbol{q}}\boldsymbol{z}^{\dagger}_{\mathcal{V}_k',\mathcal{V}_k''}(\boldsymbol{p},\boldsymbol{q})\| \leq \sum_{j\in\mathcal{V}} \frac{\overline{\sigma}_{\boldsymbol{H}}\overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}^2_{\boldsymbol{H}}} \left(\frac{\overline{\sigma}^2_{\boldsymbol{H}}-\underline{\sigma}^2_{\boldsymbol{H}}}{\overline{\sigma}^2_{\boldsymbol{H}}+\underline{\sigma}^2_{\boldsymbol{H}}}\right)^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4}-1\right\rceil_+} \|q_j\|.$$

Finally, from

$$\begin{aligned}\|R_{i\leftarrow\mathcal{V}_k',\mathcal{V}_k''}z^{\dagger}_{\mathcal{V}_k',\mathcal{V}_k''}(\boldsymbol{p}) - R_{i\leftarrow\mathcal{V}_k',\mathcal{V}_k''}z^{\dagger}_{\mathcal{V}_k',\mathcal{V}_k''}(\boldsymbol{p}')\| &\leq \left\|\int_0^1 R_{i\leftarrow\mathcal{V}_k',\mathcal{V}_k''}D_{\boldsymbol{p}'-\boldsymbol{p}}z^{\dagger}_{\mathcal{V}_k',\mathcal{V}_k''}((1-t)\boldsymbol{p}+t\boldsymbol{p}')dt\right\| \\ &\leq \int_0^1 \|R_{i\leftarrow\mathcal{V}_k',\mathcal{V}_k''}D_{\boldsymbol{p}'-\boldsymbol{p}}z^{\dagger}_{\mathcal{V}_k',\mathcal{V}_k''}((1-t)\boldsymbol{p}+t\boldsymbol{p}')\|dt \\ &\leq \sum_{j\in\mathcal{V}} \Upsilon\rho^{\left\lceil \frac{d_{\mathcal{G}}(i,j)}{4}-1\right\rceil_+}\|p_j-p_j'\|,\end{aligned}$$

one can obtain (7.4). □

Note that the ϵ term in Theorem 2.2 does not appear in Theorem 7.6. This is because the ϵ term originates from the error in singular values of the Hessian matrix, but for QPs, such singular values do not change by the perturbation; thus, we do not need to rely on the argument based on continuity of singular values, and thus, the ϵ term does not appear.

7.3 Convergence

Now that we have the global EDS, we can apply the same proof technique used in Theorem 6.5 to prove the global convergence of the OSM (6.17). The convergence theorem is stated as follows.

Theorem 7.7 (Global Convergence for QP)**.** *Under Assumption 7.13, 7.14, 7.15 for $\mathcal{Q}$ satisfying (7.2), given any non-overlppinag partition $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$ and overlapping partitions $\{\mathcal{V}_k'\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}_k''\}_{k\in\mathcal{K}}$ satisfying (6.15), $\mu\geq\overline{\mu}$ (defined in (3.5)), and any $\boldsymbol{z}^{(0)}$, the sequence $\{\boldsymbol{z}^{(\ell)}\}_{\ell=0}^{\infty}$ generated by OSM in (6.17) satisfies (6.19), where $\alpha := \psi\Upsilon\rho^{\lceil\omega/4-1\rceil_+}$, Υ and ρ are defined in Theorem 7.6, ω is the size of overlap, and ψ is the size of boundary.*

Proof. The sequence is well-defined for any $\boldsymbol{z}^{(0)}$ since by Lemma 7.19 each subproblem has a unique solution regardless of the data. By Lemma 6.15, the uniqueness of the solution, and

the definition of the size of overlap and boundary, we have

$$\left\| R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^{\dagger}_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}) - R_{i\leftarrow\mathcal{V}'_k,\mathcal{V}''_k} \boldsymbol{z}^{\dagger}_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}') \right\| \leq \psi \Upsilon \rho^{\lceil \omega/4-1 \rceil_+} \max_{j\in\mathcal{N}^2[\mathcal{V}'_k]\setminus\mathcal{V}''_k} \|z_j - z'_j\|, \tag{7.6}$$

for each $i \in \mathcal{V}_k$ and $k \in \mathcal{K}$. The recursion in (7.6) and the fact that:

$$\boldsymbol{z}^{\dagger}_{\mathcal{V}'_k,\mathcal{V}''_k}(\boldsymbol{z}^{\star}) = \boldsymbol{z}^{\star}_{\mathcal{V}'_k,\mathcal{V}''_k}$$

yields:

$$\max_{i\in\mathcal{V}} \|z_i^{(\ell+1)} - z_i^{\star}\| \leq \alpha \max_{i\in\mathcal{V}} \|z_i^{(\ell)} - z_i^{\star}\|. \tag{7.7}$$

The convergence result (6.19) can be obtained from (7.7). □

Theorem 7.7 indicates that the OSM has stronger convergence properties when the gsNLP is actually a convex quadratic program. This motivates us to hypothesize that OSM can be used for solving gsQPs that appear within the SQP method. When the SQP method is applied for solving gsNLPs, each QP subproblem reduces to a gsQP. Thus, one may use OSM to speed up the subproblem solution within the SQP algorithm. In Chapter 9, we will discuss the possible implementation of OSM within the SQP method. Comparing with the convergence theorem for gsNLPs (Theorem 6.5), in Theorem 7.7, it was not necessary to prove the well-definedness of the sequence generated by OSM (6.17) because the subproblems are always feasible and have a unique solution. Thus, regardless of its convergence, the sequence generated by (6.17) is always well-defined.

Chapter 8

Linear Systems

In this chapter we analyze the convergence of the OSM for solving graph-structued linear systems (gsLSs):

$$\sum_{j\in\mathcal{N}_{\mathcal{G}}[i]} J_{i,j}x_j = h_i,\ i\in\mathcal{V}, \tag{8.1}$$

where $\mathcal{G}=(\mathcal{V},\mathcal{E})$ is a graph, $x_i\in\mathbb{R}^{n_{x_i}}$ is the nodal variable at node i and $J_{i,j}\in\mathbb{R}^{n_{x_i}\times n_{x_j}}$ for $i,j\in\mathcal{V}$. Note that we have assumed that the diagonal blocks $J_{i,i}$ are square. The problem can be written in a compact form:

$$\boldsymbol{J}\boldsymbol{x}=\boldsymbol{h}, \tag{8.2}$$

where $\boldsymbol{J}:=\{\boldsymbol{J}_{i,j}\}_{i\in\mathcal{V},j\in\mathcal{V}}$ and $\boldsymbol{x}:=\{\boldsymbol{x}_i\}_{i\in\mathcal{V}}$. Note that $\boldsymbol{J}$ is a square matrix because its diagonal blocks are square. As before, we define:

$$\boldsymbol{J}_{\mathcal{V}',\mathcal{V}''}:=\{J_{i,j}\}_{i\in\mathcal{V}',j\in\mathcal{V}''}$$
$$\boldsymbol{h}_{\mathcal{V}'}:=\{h_i\}_{i\in\mathcal{V}'}.$$

The OSM for QP studied in Chapter 7 can be certainly applied to solve (8.2). For example, if $\boldsymbol{J}$ is positive definite, one can view $\boldsymbol{J}$ as $\boldsymbol{G}$ and apply the OSM for unconstrained QPs; if $\boldsymbol{J}$ has a KKT-matrix-like structure, one can apply the OSM for equality-constrained QPs. However, neither of those stucture may exist in the problem. Still, the linear systems can be considered as the constraints in QPs; however, when they are treated as QPs, one

needs to solve the augmented system:

$$\begin{bmatrix} & \boldsymbol{J}^\top \\ \boldsymbol{J} & \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{h} \end{bmatrix},$$

which may not be desireable from the numerical stand point (the dimension of the problem doubles).

As such, instead of attempting to apply the QP-based algorithm, in this chapter, we discuss how to solve (8.2) from a purely linear algebra perspective. The algorithm is equivalent to the OSM for QPs if $\boldsymbol{J}$ is positive definite and we view the problem as a minimization problem: $\min_{\boldsymbol{x}} (1/2)\boldsymbol{x}^\top \boldsymbol{J}\boldsymbol{x} - \boldsymbol{h}^\top \boldsymbol{x}$. Moreover, the presented algorithm actually is equivalent to the restricted additive Schwarz (RAS) method, which has been extensively studied in the literature. Our graph-structured matrix analysis in Theorem 2.1 and the convergence proof technique based on EDS (Theorem 6.5) allows the characterization of the convergence rate. In particular, we show that the convergence rate improves exponentially with the size of overlap. However, it should be caveated that the inheritance of regularity cannot be guaranteed under this method; that is, even if the full problem (8.2) is well-conditioned, the subproblem may become arbitrarily ill-conditioned. This method can be used for solving linear systems that arise within optimization algorithms (e.g., interior point method). In the rest of the chapter, we will present the algorithm and study its global convergence property.

Related Work: Overlapping Schwarz method, also called overlapping domain decomposition or additive/multiplicative Schwarz method, has been widely studied in the context of PDEs [31, 53, 116, 127, 160]. Compared to non-overlapping versions (block Jacobi or Gauss-Seidel), convergence can be accelerated with this scheme by incorporating the overlap [52, 116, 140]. Classical works on overlapping Schwarz have focused on linear systems that arise from the discretized PDEs. Subsequently, the OSM has been generalized so as to be applicable to general sparse linear systems [33]. Finally, the *restricted* additive Schwarz method (we will see that this algorithm is equivalent to ours) is proposed [30] and demonstrated to outperform its non-restricted counterpart [55]. One sweep of RAS iteration can be interpreted as applying a preconditioner; this interpretation is beneficial since sophisticated

iterative solution methods (e.g., generalized minimal residual algorithm; GMRES) can be used along with Schwarz preconditioner [138, 139]. A number of works in the literature have used such overlapping Schwarz-based preconditioning strategies [28, 30, 154]. The OSM for general linear algebra systems are available in powerful scientific computing packages such as `PETSc` [12]. Most works reported on general linear systems, however, have only analyzed empirical convergence behavior [29, 30, 32, 71]. The work in [68, 69, 101] provides theoretical convergence analysis for general linear systems but does not establish a relationship between the convergence rate and the size of the overlap. In this context, the contribution of the contents in this chapter can be summarized as follows: (i) We provide a theoretical convergence analysis for the OSM for general (graph-structured) linear systems. (ii) We establish an explicit dependence between the convergence rate and the size of the overlap (in particular, exponential improvement). A similar form of the bound of convergence rate (exponential improvement with respect to the size of overlap) is also observed in the PDE literature [50], but our result is more generally applicable.

8.1 Algorithm

We now describe the OSM to solve (8.2). Note that this algorithm is different from (6.17), where the partial augmented Lagrangian strategy is used. For each subdomain $k \in \mathcal{K}$, we consider a non-overlapping partitiong $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$ and an overlapping partition $\{\mathcal{V}'\}_{k\in\mathcal{K}}$. Note that we only have one overlapping partition, as oppose to the algorithms for gsNLPs and gsQPs, where a pair of overlapping partitions $(\{\mathcal{V}'_k\}_{k\in\mathcal{K}}, \{\mathcal{V}''_k\}_{k\in\mathcal{K}})$ are used. The OSM for solving (8.2) is defined as follows:

$$\boldsymbol{x}_{\mathcal{V}_k}^{(\ell+1)} = R_{\mathcal{V}_k \leftarrow \mathcal{V}'_k} (\boldsymbol{J}_{\mathcal{V}'_k, \mathcal{V}'_k})^{-1} (\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k, \mathcal{V}\setminus\mathcal{V}'_k} \boldsymbol{x}_{\mathcal{V}\setminus\mathcal{V}'_k}^{(\ell)}), \ k \in \mathcal{K}, \ \ell = 0, 1, \cdots, \tag{8.3}$$

where the solution at iteration ℓ is denoted by $\boldsymbol{x}^{(\ell)} \in \mathbb{R}^n$. After solving the subsystem for $\mathcal{V}'_k$, the solution is restricted to $\mathcal{V}_k$ to obtain the next iterate. Note that (8.3) for each subsystem k can be solved in parallel. By solving (8.3) for each index $k \in \mathcal{K}$, one can construct the entire iterate vector $\boldsymbol{x}^{(\ell+1)}$. When $\omega = 0$, the overlapping Schwarz scheme reduces to a

block-Jacobi scheme (decentralized) while, when ω is maximal ($\mathcal{V}'_k = \mathcal{V}$), the overlapping Schwarz scheme becomes a direct solution method (centralized). In this sense, overlapping Schwarz provides a bridge between fully decentralized and fully centralized schemes (thus providing flexibility).

The iteartive algorithm in (8.3) can be rewritten as:

$$\boldsymbol{x}^{(\ell+1)} = \boldsymbol{S}\boldsymbol{x}^{(\ell)} + \boldsymbol{s}, \tag{8.4}$$

where the blocks of $\boldsymbol{S}$ and $\boldsymbol{s}$ are defined as follows: for $k \in \mathcal{K}$,

$$\boldsymbol{S}_{\mathcal{V}'_k,\mathcal{V}'_k} = \boldsymbol{0} \tag{8.5a}$$

$$\boldsymbol{S}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k} = R_{\mathcal{V}_k\leftarrow\mathcal{V}'_k}\boldsymbol{J}^{-1}_{\mathcal{V}'_k,\mathcal{V}'_k}\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k} \tag{8.5b}$$

$$\boldsymbol{s}_{\mathcal{V}'_k} = R_{\mathcal{V}_k\leftarrow\mathcal{V}'_k}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^{-1}\boldsymbol{h}_{\mathcal{V}'_k}. \tag{8.5c}$$

Therefore, the OSM can be interpreted as a typical iterative solution scheme for a linear system.

The iteration scheme in (8.3) can alternatively expressed in the following form:

$$\begin{aligned}
\boldsymbol{x}^{(\ell+1)} &= \sum_{k\in\mathcal{K}} R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^{-1}(\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k}\boldsymbol{x}^{(\ell)}_{\mathcal{V}\setminus\mathcal{V}'_k}) \\
&= \sum_{k\in\mathcal{K}} R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^{-1}(\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k}\boldsymbol{x}^{(\ell)}_{\mathcal{V}\setminus\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}\boldsymbol{x}^{(\ell)}_{\mathcal{V}'_k} + \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}\boldsymbol{x}^{(\ell)}_{\mathcal{V}'_k}) \\
&= \sum_{k\in\mathcal{K}} R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k}\left(\boldsymbol{x}^{(\ell)}_{\mathcal{V}'_k} + (\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^{-1}(\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}}\boldsymbol{x}^{(\ell)})\right) \\
&= \boldsymbol{x}^{(\ell)} + \sum_{k\in\mathcal{K}} R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^{-1}(\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}}\boldsymbol{x}^{(\ell)}),
\end{aligned}$$

were, $R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k} := R^{\top}_{\mathcal{V}_k\leftarrow\mathcal{V}}R_{\mathcal{V}_k\leftarrow\mathcal{V}'_k}$. Finally, one can write:

$$\boldsymbol{x}^{(\ell+1)} = \boldsymbol{x}^{(\ell)} + \underbrace{\left(\sum_{k\in\mathcal{K}} R_{\mathcal{V}\leftarrow\mathcal{V}_k\leftarrow\mathcal{V}'_k}\boldsymbol{J}^{-1}_{\mathcal{V}'_k,\mathcal{V}'_k}R_{\mathcal{V}'_k\leftarrow\mathcal{V}}\right)}_{\boldsymbol{P}^{-1}}(\boldsymbol{h} - \boldsymbol{J}\boldsymbol{x}^{(\ell)}). \tag{8.6}$$

The RAS method for linear systems is typically expressed in (8.6) form. Thus, we can see the equivalence of (8.3) with the standard RAS method, which shows the connection between

overlapping Schwarz for gsNLPs and the standard RAS scheme. In addition, in this way, we can consider $\boldsymbol{P}$ as a preconditioner of an iterative linear solver. Algorithm (8.6) uses a simple static iteration (also called a Richardson iteration [139]), but more sophisticated iterative methods (e.g., GMRES) can also be used by treating $\boldsymbol{P}$ as a preconditioner. Although (8.3) and (8.6) are mathematically equivalent, there can be a difference from the numerical standpoint due to the finite precision of the computers.

8.2 Convergence

We now study the convergence of the algorithm in (8.3). For the analysis, we mainly use the form (8.3) and (8.4), but the same result also applies to Algorithm (8.6) as they are mathematically equivalent. First, we state a well-known convergence condition for iterative linear solvers.

Lemma 8.20. *Consider nonsingular $\boldsymbol{J}$ with nonsingular $\{\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}\}_{k\in\mathcal{K}}$. The following are equivalent.*

(a) $I-\boldsymbol{S}$ is nonsingular, and the sequence generated by (8.3) converges to the solution of (8.2) as $\ell\to\infty$ (for any $\boldsymbol{x}^{(0)}$).

(b) $SR(\boldsymbol{S})<1$.

Here, $SR(\cdot)$ denotes the spectral radius of the argument.

Proof. It is well known (see [17, Proposition 6.1, p144]) that $SR(\boldsymbol{S})<1$ if and only if $I-\boldsymbol{S}$ is nonsingular and the iteration of the form (8.3) converges to its fixed point. That is, the iteration converges to the solution of

$$\boldsymbol{x}=\boldsymbol{S}\boldsymbol{x}+\boldsymbol{s}. \tag{8.7}$$

By the assumption that $\boldsymbol{J}$ is nonsingular, (8.2) has a unique solution. For both conditions (a) and (b), $I-\boldsymbol{S}$ is nonsingular, and thus (8.7) has a unique solution. Hence, we need only to show that the solution of (8.2) is the solution of (8.7). Let $\boldsymbol{x}^\star$ be the solution of (8.2).

Then we have

$$\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}\boldsymbol{x}^\star_{\mathcal{V}'_k} = -\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k}\boldsymbol{x}^\star_{\mathcal{V}\setminus\mathcal{V}'_k} + \boldsymbol{h}_{\mathcal{V}'_k} \tag{8.8}$$

for $k \in \mathcal{K}$. Since $\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}$ is nonsingular, we can obtain:

$$\boldsymbol{x}^\star_{\mathcal{V}_k} = R_{\mathcal{V}_k\leftarrow\mathcal{V}'_k}\boldsymbol{J}^{-1}_{\mathcal{V}'_k,\mathcal{V}'_k}\left(\boldsymbol{h}_{\mathcal{V}'_k} - \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{V}'_k}\boldsymbol{x}^\star_{\mathcal{V}\setminus\mathcal{V}'_k}\right) \tag{8.9}$$

Equation (8.9) takes the same form with the fixed-point equation for (8.7). Therefore, $\boldsymbol{x}^\star$ solves (8.7). This proves that the unique solutions of (8.2) and (8.7) are equal. □

Now, we analyze the convergence of OSM (8.3) to solve (8.2). First we need to prove the following technical lemma:

Lemma 8.21. *We define* $\|\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} := \max_{k\in\mathcal{K}}\|\boldsymbol{x}_{\mathcal{V}_k}\|$. *The following holds.*

(a) $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$ *is a vector norm.*

(b) The matrix norm induced by $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$ *satisfies:*

$$\|\boldsymbol{A}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} \le \max_{k\in\mathcal{K}}\|\boldsymbol{A}_{\mathcal{V}_k,\mathcal{V}}\|. \tag{8.10}$$

Proof of (a). Subadditivity: For any $\boldsymbol{x}, \boldsymbol{x}'$, we have that:

$$\begin{aligned}
\|\boldsymbol{x}+\boldsymbol{x}'\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} &= \max_{k\in\mathcal{K}}\|\boldsymbol{x}_{\mathcal{V}_k} + \boldsymbol{x}'_{\mathcal{V}_k}\| \\
&\le \max_{k\in\mathcal{K}}(\|\boldsymbol{x}_{\mathcal{V}_k}\| + \|\boldsymbol{x}'_{\mathcal{V}_k}\|) \\
&\le \max_{k\in\mathcal{K}}\|\boldsymbol{x}_{\mathcal{V}_k}\| + \max_{k\in\mathcal{K}}\|\boldsymbol{x}'_{\mathcal{V}_k}\| \\
&\le \|\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} + \|\boldsymbol{x}'\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}},
\end{aligned}$$

where the first equality and last inequality follows from the definition of $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$, the first inequality follows from the triangle inequality and the second inequality follows from the property of max operator.

Absolute homogeneity: For any $a \in \mathbb{R}$ and $\boldsymbol{x}$,

$$\|a\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} = \max_{k\in\mathcal{K}}\|a\boldsymbol{x}_{\mathcal{V}_k}\|$$

$$
\begin{aligned}
&= \max_{k\in\mathcal{K}} |a| \|\boldsymbol{x}_{\mathcal{V}_k}\| \\
&= |a| \max_{k\in\mathcal{K}} \|\boldsymbol{x}_{\mathcal{V}_k}\| \\
&= |a| \|\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}},
\end{aligned}
$$

where the first and last equality follows from the definition of $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$, the second equality follows from the absolute homogeneity of 2-norm, and the third equality follows from the property of max operator.

Positive definiteness: If $\boldsymbol{x} = 0$, each $\boldsymbol{x}_{\mathcal{V}_k} = 0$. Thus, $\|\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} = 0$. Conversely, if $\|\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} = 0$, we have that $\|\boldsymbol{x}_{\mathcal{V}_k}\| = 0$, which comes from the positive definiteness of $\|\cdot\|$. Thus, we have $\boldsymbol{x} = 0$. □

Proof of (b). The desired result follows from:

$$
\begin{aligned}
\|\boldsymbol{A}\boldsymbol{x}\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} &= \max_{k\in\mathcal{K}} \|\boldsymbol{A}_{\mathcal{V}_k,\mathcal{V}}\boldsymbol{x}\| \\
&\leq \max_{k\in\mathcal{K}} \|\boldsymbol{A}_{\mathcal{V}_k,\mathcal{V}}\|,
\end{aligned}
$$

where the equality follows from the definition of $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$, and the inequality follows from the definition of induced matrix norm. □

We now define the paramters that characterzie the overlapping partition $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$. For the linear systems, the size of overlap ω is defined as follows:

$$
\omega := \min_{k\in\mathcal{K}} d_{\mathcal{G}}(\mathcal{V}_k, \mathcal{V} \setminus \mathcal{V}'_k) - 1. \tag{8.11}
$$

Note that there is some difference compared to Definition 6.6; this difference comes from the fact that we only have one overlapping subdomain $\mathcal{V}_k$ for each k (we do not have $\mathcal{V}''_k$). We now are in the position to state the main result of the current chapter.

Theorem 8.8. *Consider nonsingular $\boldsymbol{J}$ with nonsingular $\{\boldsymbol{J}_{\mathcal{V}'_k,}\}_{k\in\mathcal{K}}$. We have that*

$$
SR(\boldsymbol{S}) \leq \alpha := \frac{\overline{\sigma}_{\boldsymbol{J}}\overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}_{\boldsymbol{J}}} \left(\frac{\overline{\sigma}^2_{\boldsymbol{J}} - \underline{\sigma}^2_{\boldsymbol{J}}}{\overline{\sigma}^2_{\boldsymbol{J}} + \underline{\sigma}^2_{\boldsymbol{J}}} \right)^{\lceil \omega/4 - 1 \rceil_+} \tag{8.12}
$$

where $\overline{\sigma}_{\boldsymbol{J}} := \max_k \overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})$, $\overline{\sigma}_{\boldsymbol{F}} := \max_k \overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{N}_{\mathcal{G}}(\mathcal{V}'_k)})$, $\underline{\sigma}_{\boldsymbol{J}} := \min_k \underline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})$, *and* ω *is the size of overlapdefined in* (8.11). *Moreover, the sequence generated by* (8.3) *satisfies:*

$$\left\|\boldsymbol{x}^{(\ell)} - \boldsymbol{x}^{\star}\right\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} \leq \alpha^{\ell} \left\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^{\star}\right\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}. \tag{8.13}$$

Proof. From Lemma 8.21 and [90, Theorem 5.6.9], the spectral radius $SR(\boldsymbol{S})$ satisfies:

$$SR(\boldsymbol{S}) \leq \max_{k\in\mathcal{K}} \|\boldsymbol{S}_{\mathcal{V}_k,\mathcal{V}}\|. \tag{8.14}$$

Thus, it suffices to show $\|\boldsymbol{S}_{\mathcal{V}_k,\mathcal{V}}\| \leq \alpha$. From (8.5), we have that for any $k \in \mathcal{K}$,

$$\begin{aligned}
\|\boldsymbol{S}_{\mathcal{V}_k,\mathcal{V}}\| &\leq \|R_{\mathcal{V}_k \leftarrow \mathcal{V}'_k} \boldsymbol{J}^{-1}_{\mathcal{V}'_k,\mathcal{V}'_k} \boldsymbol{J}_{\mathcal{V}'_k,\mathcal{N}_{\mathcal{G}}(\mathcal{V}'_k)}\| \\
&\leq \frac{\overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})\overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{N}_{\mathcal{G}}(\mathcal{V}'_k)})}{\underline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^2} \left(\frac{\overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^2 - \underline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^2}{\overline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^2 + \underline{\sigma}(\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k})^2}\right)^{\lceil \omega/4-1\rceil_+} \\
&\leq \frac{\overline{\sigma}_{\boldsymbol{J}}\overline{\sigma}_{\boldsymbol{F}}}{\underline{\sigma}_{\boldsymbol{J}}^2} \left(\frac{\overline{\sigma}_{\boldsymbol{J}}^2 - \underline{\sigma}_{\boldsymbol{J}}^2}{\overline{\sigma}_{\boldsymbol{J}}^2 + \underline{\sigma}_{\boldsymbol{J}}^2}\right)^{\lceil \omega/4-1\rceil_+},
\end{aligned} \tag{8.15}$$

Here the first inequality follows from (8.5b) and the fact that $\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}\setminus\mathcal{N}_{\mathcal{G}}[\mathcal{V}'_k]} = 0$, the second inequality follows from Theorem 2.1 and the observation that $d_{\mathcal{G}}(\mathcal{V}_k, \mathcal{N}_{\mathcal{G}}(\mathcal{V}'_k)) \geq \omega + 1$ and the bandwidth of $\boldsymbol{J}$ is less than or equal to 2; when applying Theorem 2.1, we aggregate the nodes in $\mathcal{V}_k$ and $\mathcal{N}_{\mathcal{G}}(\mathcal{V}'_k)$ and apply the theorem on the aggregated graph; and the third inequality follows from the definitions of $\overline{\sigma}_{\boldsymbol{J}}$, $\overline{\sigma}_{\boldsymbol{F}}$, and $\underline{\sigma}_{\boldsymbol{J}}$. This yields (8.12). We now analyze the convergence rate of the algorithm. We note, from (8.3), that

$$\boldsymbol{x}^{(\ell)} - \boldsymbol{x}^{\star} = (\boldsymbol{S})^{\ell} \left(\boldsymbol{x}^{(0)} - \boldsymbol{x}^{\star}\right). \tag{8.16}$$

By taking $\|\cdot\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}}$ on the both side of (8.16), we have:

$$\max_{k\in\mathcal{K}} \left\|\boldsymbol{x}^{(\ell)}_{\mathcal{V}_k} - \boldsymbol{x}^{\star}_{\mathcal{V}_k}\right\| \leq \left\|\boldsymbol{S}^{\ell}(\boldsymbol{x}^{(0)} - \boldsymbol{x}^{\star})\right\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} \tag{8.17a}$$

$$\leq \|\boldsymbol{S}\|^{\ell}_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} \left\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^{\star}\right\|_{\{\mathcal{V}_k\}_{k\in\mathcal{K}}} \tag{8.17b}$$

$$\leq \left(\max_{k\in\mathcal{K}} \|\boldsymbol{S}_{\mathcal{V}_k,\mathcal{V}}\|\right)^{\ell} \max_{k\in\mathcal{K}} \left\|\boldsymbol{x}^{(0)}_{\mathcal{V}_k} - \boldsymbol{x}^{\star}_{\mathcal{V}_k}\right\| \tag{8.17c}$$

$$\leq \alpha^{\ell} \max_{k\in\mathcal{K}} \left\|\boldsymbol{x}^{(0)}_{\mathcal{V}_k} - \boldsymbol{x}^{\star}_{\mathcal{V}_k}\right\|, \tag{8.17d}$$

where the second inequality follows from the definition and submultiplicativity of induced matrix norm, and the third inequality follows from (8.21)(b), and the last inequality follows (8.15). This proves the convergence. □

The theorem suggests that the convergence rate decreases (improves) exponentially with the size of overlap ω. This coincides with the results for gsNLPs and gsQPs (Theorem 6.5, 7.7). This result formally proves the empirical observation that the convergence of the RAS algorithm improves as the size of overlap ω increases and as the conditioning of $\boldsymbol{M}$ improves [29].

However, the result in Theorem 8.8 is weaker than Theorem 6.5, 7.7 in that the non-singularity of the smaller blocks $\{\boldsymbol{J}_{\mathcal{V}'_k,\mathcal{V}'_k}\}_{k\in\mathcal{K}}$ is not guaranteed from the non-singularity of the original matrix $\boldsymbol{J}$. Thus, the conditioning of the subproblem may become arbitrarily bad, and $\overline{\sigma}_{\boldsymbol{J}}/\underline{\sigma}_{\boldsymbol{J}}$ may become arbitrarily large; this eventually makes ρ arbitrarily close to one. This is different from the previous theorems where the subproblems inherit the regularity conditions from the original problem.

Chapter 9

Implementation

In this chapter, we discuss the implementation of modeling and the solution of gsNLPs. First, we discuss modeling platforms for gsNLPs. In the context of gsNLPs, modeling and solution are not separate issues because of the following reasons. First, in OSM, the subproblems are created by reformulating the original problem into a set of subproblems, requiring a flexible algebraic modeling language that allows manipulating the algebraic expressions in constraints and objectives (e.g., see (6.4)). In addition, the graph structure of the model should be communicated with the solution algorithms to obtain the overlapping and non-overlapping partition structures. For example, when constructing the partitions $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$, $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$, and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$, it is necessary to have the structural information of the problem. Thus, the implementation of OSM can benefit from a specialized algebraic, graph-based modeling interface. To allow implementing OSM as a generic nonlinear programming solver, we have implemented algebraic modeling language in `SimpleNL.jl` [4]. It has algebraic modeling and basic graph-based modeling capabilities. In addition, we have implemented graph-based modeling tool `Plasmo.jl` [2], which uses `JuMP.jl` [54] as an algebraic modeling backend. The abstraction in `Plasmo.jl` naturally exposes problem structure to algorithms and provides a modular approach to construct models.

Second, we discuss various implementations of OSM. In particular, OSM can be applied at three different levels: problem-level, subproblem-level, and linear algebra-level. We introduce the implementation of problem-level decomposition with SimpleSchwarz.jl [5]. This package is interfaced with flexible algebraic modeling language `SimpleNL.jl` and applies OSM at the

problem level. We introduce the implementation of linear algebra-level decomposition with `MadNLP.jl` [1]. `MadNLP.jl` is interfaced with graph-based modeling language `Plasmo.jl` and is capable of applying the OSM at the linear algebra level within the interior point method. We currently do not have a working example of subproblem level decomposition, but we discuss several possibilities.

9.1 Modeling

In this section, we introduce the algebraic and graph-based modeling interface that facilitates the implementation of OSM at different levels. As the main focus of the current dissertation is on the solution algorithms for gsNLPs, we focus on high-level ideas and showcase some of the basic syntaxes of those platforms, rather than explaining the full details of the implementation. For the details of the implementation of the modeling interfaces, the readers are referred to [97, 98] and the repositories [2, 4].

9.1.1 Algebraic Modeling Language

Algebraic modeling languages allow for creating optimization models that can later be interfaced with optimization solvers. Many modern algebraic modeling platforms, such as JuMP [54], CasADi [10], AMPL [66], GAMS [42], and Pyomo [85], provide convenient symbolic modeling interface for the user and are equipped with (either internal or external) automatic differentiation (AD) capabilities, so that it can communicate model functions and derivatives with the optimization solvers. Algebraic modeling languages streamline the modeling process and facilitate the implementation of advanced decomposition algorithms. In the context of graph-structured optimization, the algebraic modeling language is particularly useful because (i) the problems of interest have highly complicated algebraic structures, so hand-coding model functions and derivatives are practically difficult and prone to error, and (ii) the stored algebraic expressions allow further manipulations for flexible subproblem formulation.

```
using SimpleNL, Ipopt

m = SimpleNL.Model(Ipopt.Optimizer; print_level=5)

x = [variable(m; start=mod(i,2)==1 ? -1.2 : 1.) for i=1:1000]
objective(m, sum(100(x[i-1]^2-x[i])^2+(x[i-1]-1)^2 for i=2:1000))
for i=1:998
  constraint(m, 3x[i+1]^3+2*x[i+2]-5+sin(x[i+1]-x[i+2])sin(x[i+1]+x[i
 +2])+4x[i+1]-x[i]exp(x[i]-x[i+1])-3 == 0)
end

optimize!(m)
```

Code 9.1 Algebraic modeling using `SimpleNL.jl`.

We have implemented a new algebraic modeling language `SimpleNL.jl` [4], to facilitate the modeling of gsNLPs and the implementation of OSM. Like other algebraic modeling languages such as `JuMP.jl` and CasADi, SimpelNL.jl has a symbolic user interface and automatic differentiation backend, which communicates with the user-specified nonlinear optimization solver. Compared to `JuMP.jl`, an algebraic modeling language implemented in the same programming language Julia [19], `SimpleNL.jl` allows more flexible manipulation of the algebraic expressions (as of JuMP v0.21.6 and SimpleNL v0.1.0). Thus, it is more suitable for implementing OSM, which requires the manipulation of various nonlinear expressions. Furthermore, there is no performance compromise of using `SimpleNL.jl` instead of the other popular packages. `SimpleNL.jl` implements efficient reusable computational graph-based reverse-mode automatic differentiation; this method accelerates the derivative evaluation by reducing the overhead of repeatedly creating the computational graph. For the problems that we used for the case study in Chapter 10, it was found that `SimpleNL.jl` is as fast as `JuMP.jl` and AMPL if not faster. A code snippet of modeling syntax of `SimpleNL.jl` is provided in Code 9.1.

9.1.2 Graph-Based Modeling Language

gsNLPs can be conveniently modeled using specialized modeling platforms such as `Plasmo.jl` [97, 98]. `Plasmo.jl` represents any optimization model as a hierarchical graph wherein the nodes contain optimization models with corresponding objectives, variables, constraints, and data, and the edges contain the linking constraints across different nodes. This implementation allows handling hundreds of thousands to millions of nodes and edges. The structural information stored in `Plasmo.jl` can be communicated with the optimization solvers to enable structure-exploiting algorithms such as OSM and others [27,36,136,145]. In addition, `Plasmo.jl` enables the modular construction and analysis of highly complex models; this platform also leverages the algebraic modeling capabilities of `JuMP.jl` [54] and facilitates access to infrastructure modeling tools such as `GasModels.jl` and `PowerModels.jl` [14,38]. A code snippet of `Plasmo.jl` modeling syntax is provided in Code 9.2

9.2 Solution

In this section, we discuss the implementation of OSM at different levels. In particular, we discuss problem-level, subproblem-level, and linear algebra-level decomposition.

9.2.1 Problem-Level Decomposition

In this section, we discuss the implementation of OSM as a problem-level decomposition method. This method directly applies OSM to solve gsNLPs. As the convergence of OSM for gsNLP is only local (Theorem 6.5), it should be caveated that the convergence issue can be faced due to the nonlinearity. Below we explain the details of implementing the scheme (6.17); in particular, the implementation in SimpleSchwarz.jl is explained.

Obtaining Problem Graph: First, for a given nonlinear optimization problem modeled in an algebraic modeling platform, we construct a graph associated with the model. We assume that from the algebraic model, we have access to the objective terms $\{f_i(\cdot)\}$ and constraints $\{c_i(\cdot)\}$, each of which is a scalar function that is dependent on the decision variable vector $\boldsymbol{x}$. We assume that a high-level graph implementation (e.g., `LightGraphs.jl`

```
using Plasmo, MadNLP, GasModels

# Construct OptiGraph objects using Plasmo
graph = Plasmo.OptiGraph()
nodes = [add_node!(graph) for k=1:24]
edges = [add_edge!(graph,nodes[k],nodes[k+1])
for k=1:23]

# Construct node/edge models using GasModels
data  = GasModels.parse("data.m")
build_model(nodes[1], data)
for node in nodes
  build_node_model(node, data)
end
for edge in edges
  build_edge_model(edge, data)
end

# Solve OptiGraph model using MadNLP
MadNLP.optimize!(graph, linear_solver="schwarz")
```

Code 9.2 Graph-based modeling using `Plasmo.jl`.

[142]) is available. The graph construction procedure is performed with Algorithm 1. Here, the function getobjectives($\cdot$) and getconstraints($\cdot$) return the collection of objective terms and constraint functions; numvariables($\cdot$) function returns the number of variables in the model. Note that the graph $\mathcal{G}$ produced by Algorithm 1 considers each variable as a node; thus, the number of nodes is equal to the number of variables. The function sparsity($\cdot$) returns the index of variables that are associated with the argument. For example, if $f(x) = x[1] + x[4]$, sparsity$(f) = [1; 4]$. By inspecting the sparsity patterns of $\{f_i\}$ and $\{c_i\}$ (as done in Algorithm 1), one can obtain the graph for the entire problem. With the constructed graph, we now can consider the algebraic model as a gsNLP induced by the graph $\mathcal{G}$.

Algorithm 1 Obtaining Problem Graph

Require: *model*

$\{f_i(\cdot)\}_{i=1}^{n_f} \leftarrow$getobjectives(*model*)

$\{c_i(\cdot)\}_{i=1}^{n_c} \leftarrow$getconstraints(*model*)

Create a graph $\mathcal{G}$ with numvariables(*model*) nodes.

for $i = 1, \cdots, n_f$ **do**

  $js \leftarrow$ spartisty$(f_i(\cdot))$

  for j in js **do**

    add edge $\{i, j\}$ to $\mathcal{G}$

  end for

end for

for $i = 1, \cdots, n_c$ **do**

  $js \leftarrow$ spartisty$(c_i(\cdot))$

  for j in js **do**

    add edge $\{i, j\}$ to $\mathcal{G}$

  end for

end for

Ensure: $\mathcal{G}$

Obtaining Subproblems: To apply (6.17), we need to construct the overlapping partitions $\{\mathcal{V}'_k\}_{k\in\mathcal{K}}$ and $\{\mathcal{V}''_k\}_{k\in\mathcal{K}}$. We assume that the non-overlapping partition $\{\mathcal{V}_k\}_k$ is available from the user; the user can manually provide this by using a graph-based modeling interface (`SimpleNL.jl` has a basic graph-based modeling capability where the user can specify the subdomain for each variable) or use graph partitioning tool to obtain the partition. Furthermore, a relative size of overlap (defined in (9.1)) $\widetilde{\omega}$ is provided by the user. Note that in practice, the *relative size of overlap* might be more useful than the absolute size of overlap defined in Definition 6.6. The overall conditioning of the problem varies with the size of discretization mesh size (in the case of dynamic and PDE optimization). Thus, the proper size of overlap also varies with the discretization mesh size. Adopting the notion of the

relative size of overlap can handle such dependence on the discretization mesh size, and thus requires less amount of tuning efforts in practice. The relative size of the overlap is defined as:

$$\widetilde{\omega} = \max_k |\mathcal{V}'_k \setminus \mathcal{V}_k| / |\mathcal{V}_k| - 1. \tag{9.1}$$

For each $k \in \mathcal{K}$, one can obtain the overlapping partition $\mathcal{V}'_k$ and $\mathcal{V}''_k$ using Algorithm 2. Here, the function `neighbors`$(\cdot, \cdot)$ returns the set of node indices that are neighbored by the second argument on the graph given by the first argument. Also, the syntax `bool` ? A : B is used as a control flow statement that returns A if `bool` is true and B otherwise. Note that $\mathcal{V}'_k$ includes one extra layer of the neighborhoods compared to $\mathcal{V}''$. As such, we have the desired conditions in (6.15). Note that the function objective$(\cdot)$ adds an objective term and constraint$(\cdot)$ adds a constraints to *model*. The algebraic model *model* contains all the objective and constraint information. Note that the variables for $i \notin \mathcal{V}'_k$ are declared as parameters rather than decision variables. These parameters will be updated in each iteration of OSM.

Solution Algorihtm: Finally, we describe the implementation of the overall solution algorithm (6.17). We take *model*, $\{\mathcal{V}_k\}_{k\in\mathcal{K}}, \{\mathcal{W}_k\}_{k\in\mathcal{K}}, \widetilde{\omega}, \mu$, and tol (tolerance of KKT residual) as inputs. Here, $\mathcal{W}_k$ is the partition of constraint index sets; one way of constructing this is to assign each constraint to the block of the first associated variable. Algorithm 3 describes the overall procedure. Here, graph$(\cdot)$ and subproblem$(\cdot)$ denotes the function described in Algorithms 1 and 2; KKTError$(\cdot)$ is the function that evaluates the KKT residual of the problem; and setvalue$(\cdot, \cdot)$ sets the value of the first argument to the value of the second argument. We first call Algorithm 1, 2 to obtain the graph and the subproblems. Then, we apply the iterative procedure (6.17) to solve the problem with the updates of the information. Then, in each iteration step, we (i) set the parameter values for the subproblem to the solution values from the full problem, (ii) solve the subproblem, and (iii) update the solution of the full problem with the solution of the subproblem. The solution of the subproblem is performed by using off-the-shelf nonlinear optimization solvers (e.g., Ipopt [161]

Algorithm 2 Obtaining Subproblem

Require: $model$, $\mathcal{G}$, $\mathcal{V}_k$, $\widetilde{\omega}$, μ

$\{f_i(\cdot)\}_{i=1}^{n_f} \leftarrow$getobjectives($model$)

$\{c_i(\cdot)\}_{i=1}^{n_c} \leftarrow$getconstraints($model$)

while $|\mathcal{V}_k'| \leq (1+\widetilde{\omega})|\mathcal{V}_k|$ **do**

$\mathcal{V}_k'' \leftarrow \mathcal{V}_k'$

$\mathcal{V}_k' \leftarrow \mathcal{V}_k' \cup$ `neighbors`$(\mathcal{G},\mathcal{V}_k)$

end while

initialize $submodel_k$

$\boldsymbol{x}_k = [i \in \mathcal{V}_k' \ ?\ \text{variable}(submodel_k) : \text{parameter}(submodel_k)$ **for** $i = 1, \cdots, n_{\boldsymbol{x}}]$

$\boldsymbol{y}_k = [\text{parameter}(submodel_k)$ **for** $i = 1, \cdots, n_{\boldsymbol{y}}]$

for $i = 1, \cdots, n_f$ **do**

if $\text{sparsity}(f_i) \cap \mathcal{V}_k' \neq \emptyset$ **then**

$\text{objective}(submodel_k, f_i(\boldsymbol{x}_k))$

end if

end for

for $i = 1, \cdots, n_c$ **do**

if $\text{sparsity}(c_i) \subseteq \mathcal{V}_k''$ **then**

$\text{constraint}(submodel_k, c_i(\boldsymbol{x}_k))$

else if $\text{sparsity}(c_i) \cap \mathcal{V}_k' \neq \emptyset$ **then**

$\text{objective}(submodel_k, -y_i c_i(\boldsymbol{x}_k) + \mu c_i(\boldsymbol{x}_k)^2)$

end if

end for

Ensure: $submodel_k$

or `MadNLP.jl` [1] with HSL linear solvers [92]). The parallelization in `SimpleNL.jl` is performed using multi-thread parallelism. The iteration is repeated until the KKT error is within the user-provided tolerance.

Algorithm 3 Problem-Level Decomposition Algorithm

Require: $model$, $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$, $\{\mathcal{W}_k\}_{k\in\mathcal{K}}$, $\widetilde{\omega}$, μ, tol.

$\mathcal{G} \leftarrow$graph($model$)

for $k \in \mathcal{K}$ (in parallel) **do**

$submodel_k \leftarrow$ subproblem($model, \mathcal{G}, \mathcal{V}_k, \widetilde{\omega}, \mu$)

end for

while KKTError(model)$\leq$ tol **do**

for $k \in \mathcal{K}$ (in parallel) **do**

for $i \in \mathcal{V} \setminus \mathcal{V}'_k$ **do**

setvalue($submodel_k.\boldsymbol{x}[i]$,$model.\boldsymbol{x}[i]$)

end for

for $i \in \mathcal{V} \setminus \mathcal{V}''_k$ **do**

setvalue($submodel_k.\boldsymbol{y}[i]$,$model.\boldsymbol{y}[i]$)

end for

solve($submodel_k$)

for $i \in \mathcal{V}_k$ **do**

setvalue($model.\boldsymbol{x}[i]$,$submodel_k.\boldsymbol{x}[i]$)

end for

for $i \in \mathcal{W}_k$ **do**

setvalue($model.\boldsymbol{y}[i]$,$submodel_k.\boldsymbol{y}[i]$)

end for

end for

end while

Ensure: $model.\boldsymbol{x}, model.\boldsymbol{y}$.

9.2.2 Subproblem-Level Decomposition

This section discusses the implementation of OSM at the subproblem level. Currently, we do not have a working implementation of the subproblem level decomposition. Thus, we only discuss some of the high-level ideas.

The augmented Lagrangian method (ALM) solves NLPs by using an iterative procedure. In this procedure, the augmented Lagrangian subproblems are formulated first. Then in each iteration, the augmented Lagrangian problem is solved, and a dual update is performed based on the constraint violations on the relaxed constraints. Typically the solution of the augmented Lagrangian problem is the most time-consuming step. If the original problem is graph-structured, then the subproblem also preserves such a graph structure. Thus, the OSM (6.17) can be used for solving the augmented Lagrangian subproblem. However, if the original problem is nonlinear, the augmented Lagrangian subproblem is also nonlinear. Thus, with this method, we cannot obtain global convergence.

The sequential quadratic programming (SQP) method solves NLPs by solving a sequence of QP subproblems. QP subproblems are used to find the step direction, and the line-search or trust-region-based strategy is used to find the step size. Solving the QP subproblem is typically the most expensive step within the algorithm. As in the previous case, we can apply OSM to solve the QP subproblems. Since the OSM for QPs is globally convergent if the global regularity is satisfied and the overlap size is sufficiently large, one can enjoy stronger convergence property compared to the problem level decomposition or augmented Lagrangian-based subproblem-level decomposition.

9.2.3 Linear Algebra-Level Decomposition

We now discuss the implementation of OSM at the linear algebra level. In particular, we apply the decomposition within IPM. See Appendix A.2 for a brief introduction of the interior point method. Typically, the solution of the linear system (A.11) is the most computationally intensive step in the IP method. We propose to apply OSM for linear system (8.3) to solve

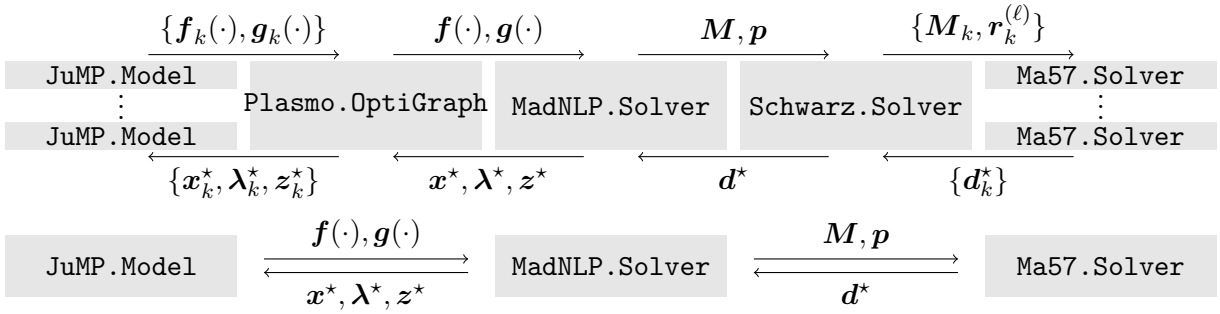

Figure 9.1 Schematics of graph-based modeling and solution (top) and conventional modeling and solution (bottom).

the system (A.11); in particular, this method is implemented in the Schwarz submodule of `MadNLP.jl`.

Related Work: Parallelization of nonlinear programs at linear algebra level, within nonlinear optimization method, has been widely studied in the literature. In typical off-the-shelf solvers, the system in (A.11) is solved using direct linear solvers that are based on block $LDL^\top$ factorization (e.g., HSL routines [92]). Decomposition strategies based on Schur complements [36] and ADMM [136] have been proposed in the literature to parallelize the step computation based on decomposition. Hübner and coworkers have developed a specialized algorithm for the solution of tree-structured KKT system within quasi-Newton method [94]. For dynamic optimization, there has been a wide range of specialized parallelization techniques [48, 109, 122, 123, 164]. The use of iterative solver for general nonlinear optimization problems have been studied in [43]. To the best of our knowledge, the OSM-based decomposition of linear systems within nonlinear optimization method has been first proposed in the work of Gerstner and coworkers [72]; therein, the OSM is used for solving KKT systems that appear within multi-period OPF problems. The solution of (A.11) based on direct block $LDL^\top$ factorization reveals the inertia (the number of positive, zero, negative eigenvalues) of $\boldsymbol{M}$. This inertia information is crucial in determining the acceptability of the computed step and in triggering the regularization of the linear system. However, inertia is not available when using iterative solution algorithms (as proposed in this work). In `MadNLP.jl`, we use

an inertia-free regularization strategy to determine the acceptability of the step [35]. This method performs a simple negative curvature test to trigger regularization.

The `Solver` object of `MadNLP.jl` is created from the `OptiGraph` object of `Plasmo.jl`. The `Solver` object of `MadNLP.jl` uses a line-search filter IP method [161] to solve the problem. The step computation is performed by the linear solver specified by the user. The linear solver can be specified either as a direct solver or as the OSM solver. When the OSM solver is chosen, multiple subproblem solver objects are created by using standard direct solvers (e.g., by using `Ma57` of HSL routines). These subproblem solvers are used for factorization and backsolve for $\boldsymbol{M}_k$ blocks. The OSM (8.6) exploits multi-thread parallelism available within `Julia`. After termination of the IP solution procedure, the primal-dual solutions are sent back to the `OptiGraph` object and `Model` objects from `JuMP.jl` so that the user can query the solution via the interface provided by `Plasmo.jl` and `JuMP.jl`. See Figure 9.1 for a comparison with a conventional implementation. In the `Schwarz` submodule of `MadNLP.jl`, ω is set automatically based on the relative size of $\mathcal{V}_k$, and adaptively adjusted whenever a convergence issue is faced.

Chapter 10

Numerical Experiments

This chapter presents the numerical results that demonstrate OSM. In particular, we demonstrate the implementation of OSM at the problem level and the linear algebra level.

10.1 Problem Level Decomposition

We use four different bechmark problems to demonstrate the OSM; (i) quadrotor motion planning problem (dynamic optimization; Appendix B.2), (ii) stochastic storage control problem (stochastic optimization; Appendix B.3), (iii) thin plate temperature control problem (PDE optimizaiton with Dirichlet boundary condition; Appendix B.4), and (iv) AC PSSE (network optimization; Appendix B.6).

10.1.1 Methods

We conduct the following numerical study for each problem instance. For each problem, we apply Algorithm 3 to solve the given problem, while varying the problem or algorithm parameter. The varied parameters are as follows: the relative size of overlap $\widetilde{\omega}$, the conditioning parameter (η, b), the penalty parameter μ. While we vary one parameter, the other parameters are fixed to the base case, described in the associated appendix sections. Lastly, we compare the performance of OSM with ADMM and Ipopt. A brief introduction of ADMM and a description of our implementation in `SimpleADMM.jl` [3] is provided in Appendix A.1. A brief introduction of IPM (the algorithm of Ipopt) and the implementation in Ipopt [161] and `MadNLP.jl` [1] is described in A.2. The code was run on a server computer equipped with

2 CPUs of Intel Xeon CPU E5-2698 v4 running 2.20GHz (20 core for each), and 34 threads are used for the computation. The results can be reproduced using the scripts provided in https://github.com/zavalab/JuliaBox/tree/master/ShinThesis.

10.1.2 Results

The convergence profiles of OSM with different sizes of overlap are shown in Figure 10.1, 10.2. For all the tested problem instances, we could see that OSM converges faster with larger overlaps. This demonstrates our theoretical development in Theorem 6.5. We do observe the trade-off between the convergence rate and the subproblem complexity. One can see that increasing the size of overlap increases the solution time per iteration; this is due to the increase in the subproblem complexity. However, for the range of the size of overlap we used within this case study, the solution time is always decreased when the size of overlap is increased. Note that convergence issues can be faced when the size of overlap is not adequate, as observed in the stochastic programming case.

The convergence profiles of OSM for different regularization parameters are shown in Figure 10.3, 10.4. Similarly to the case study in Chapter 5, we use the regularization parameter (η, b) to control the regularity of the problem. Note that increasing η strengthens the SSOSC and increasing b strengthens the LICQ. One can see from Figure 10.3 that except for the PDE optimization case, the convergence rate is indeed the fastest when both η and b are sufficiently large. The PDE-based optimization has an almost constant convergence rate for different (η, b) values because the heat diffusion equation itself has a strong dissipative property (thus, the effect of additional regularization is negligible). This verifies our uniform regularity results in Chapter 3 that it is necessary for the smallest eigenvalue of the reduced Hessian and the smallest singular value of the constraint Jacobian to be sufficiently bounded away from zero in order for EDS to be strong enough; due to the inheritance of regularities, the subproblems inherits such regularity, and stronger EDS eventually enables faster convergence of OSM.

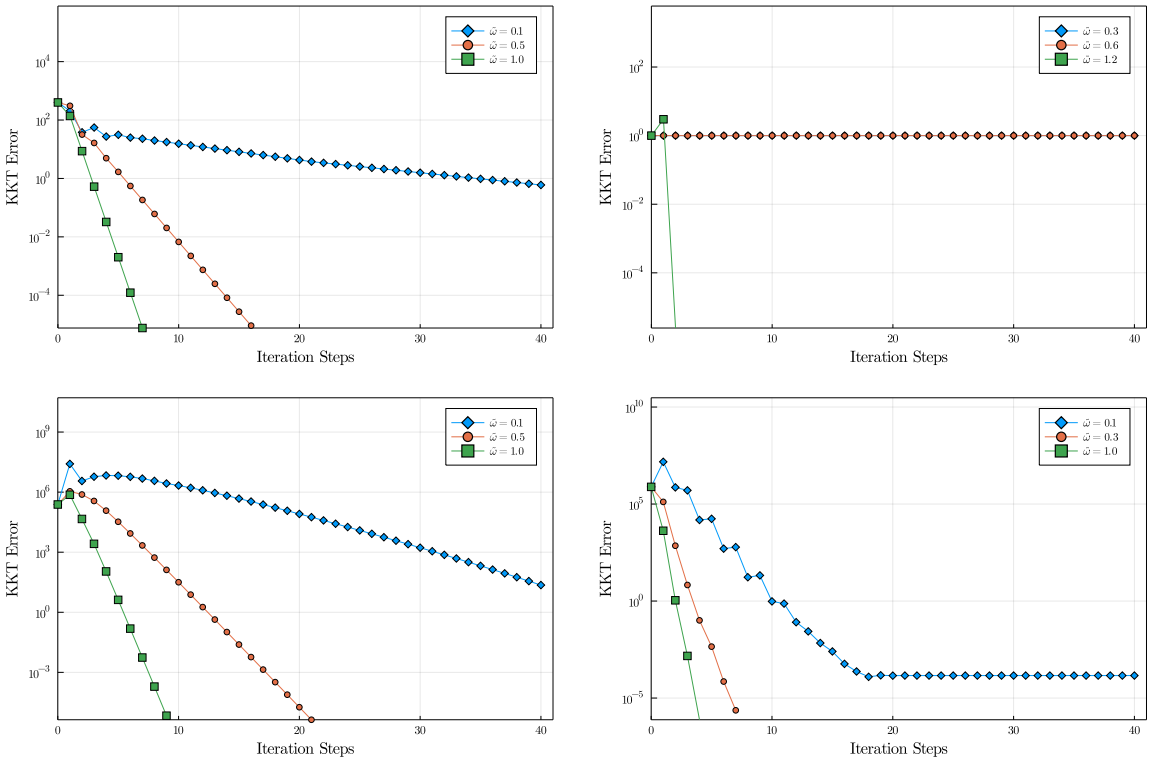

Figure 10.1 Convergence profiles (in iteration steps) of overlapipng Schwarz method for different sizes of overlap. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

In Figure 10.5, 10.6 the convergence profiles of OSM for different penalty parameters are represented. Recall from Theorem 6.5 that it is necessary for the penalty parameter μ to be sufficiently large in order to guarantee the uSSOSC in the subproblems, but its impact on the convergence rate is not strong (it only linearly increases the upper bound of uBLH). From the results in 10.5, we can see that indeed the effect of the penalty parameter on the convergence rate is not strong.

Finally, in Figure 10.7, 10.8, we present the benchmark of OSM with ADMM and Ipopt. First, in Figure 10.7, we can compare the convergence speed of OSM and ADMM in iteration steps. Here, the size of overlap is tuned with appropriate size of overlap ($\widetilde{\omega} = 1$ for dynamic optimization, $\widetilde{\omega} = 1.2$ for stochastic optimization, $\widetilde{\omega} = 2$ for PDE-optimization, $\widetilde{\omega} = 1$ for

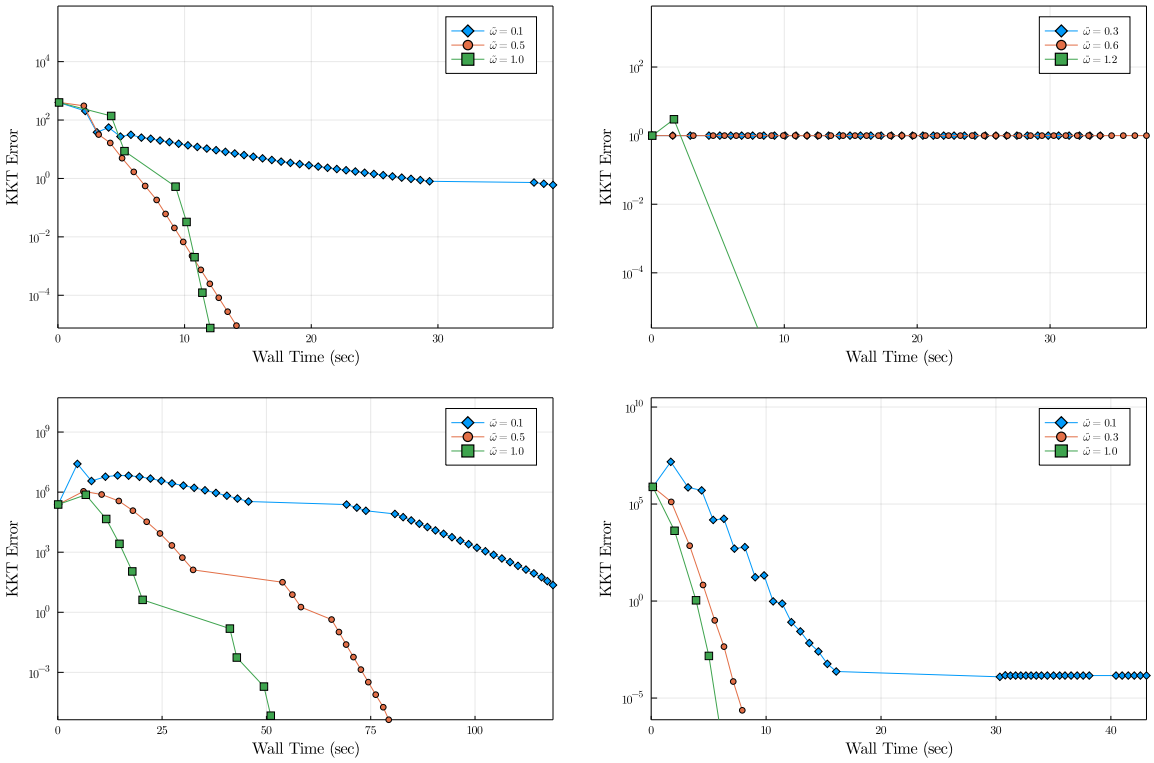

Figure 10.2 Convergence profiles (in wall times) of overlapipng Schwarz method for different sizes of overlap. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

network optimization). One can see that OSM converges significantly faster than ADMM. This is because OSM enjoys exponential improvement in the convergence rate while ADMM has a fixed convergence rate. One can make a similar observation from Figure 10.8, but the benefit of overlap becomes smaller. This is because increasing the size of overlap also increases the subproblem complexity. Thus, each subproblem in OSM is more difficult to solve than the subproblems in ADMM since it contains more variables and constraints. This is manifested when the time per iteration is compared. However, even if they are compared in wall time, one can see that overlapping Schwarz has a much faster convergence speed for all four examples.

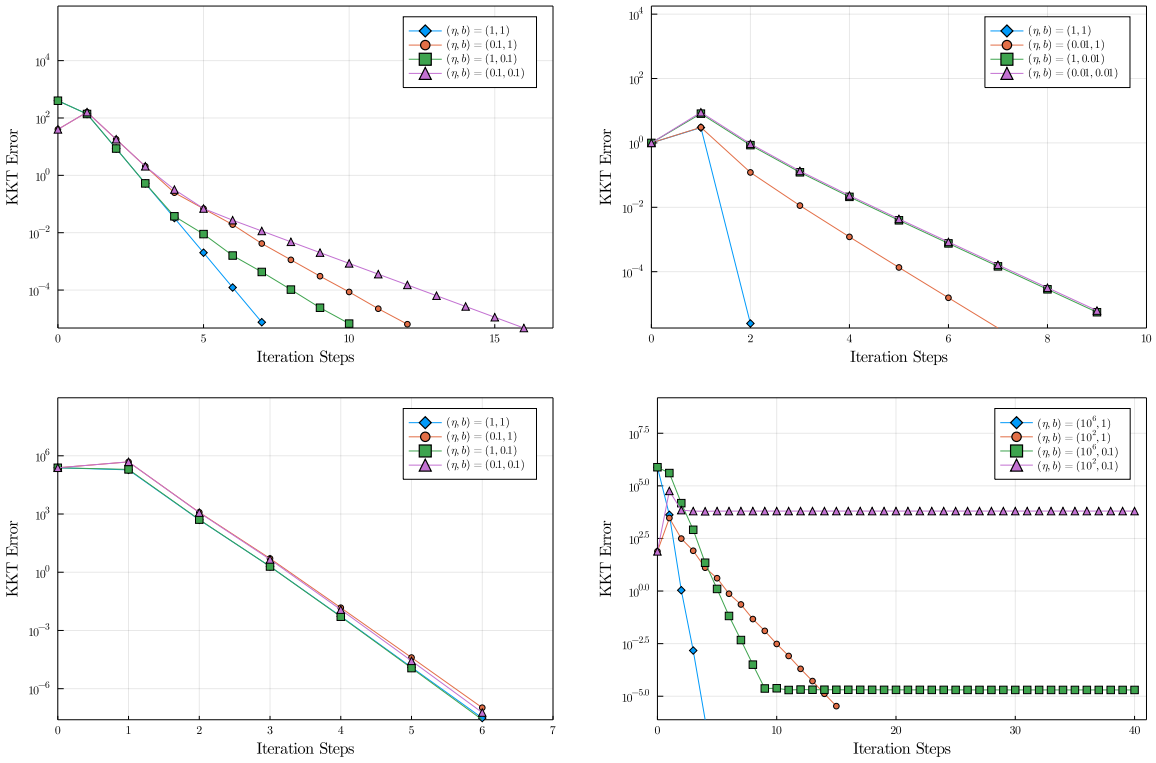

Figure 10.3 Convergence profiles (in iteration steps) of overlapipng Schwarz method for different regularizations. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

In Figure 10.8, we also have represented the solution time for Ipopt to solve the full problem, so that we can compare the performance with the centralized method. We observe that both OSM and ADMM are not as fast as solving the centralized problems except for the PDE optimization case. However, we can see that for OSM, most of the time is spent in the first iteration. This is because of the absence of a good initial guess. One can see that after the first iteration step, the convergence of OSM is very fast, and it is comparable to the centralized solver. Thus, if a good initial guess is available (e.g, as in receding horizon control/estimation problems), the OSM can be indeed effective.

We also emphasize that the advantage of decomposition is expected to be greater for larger problems. One can see from Table 10.1 that the numbers of variables in the tested

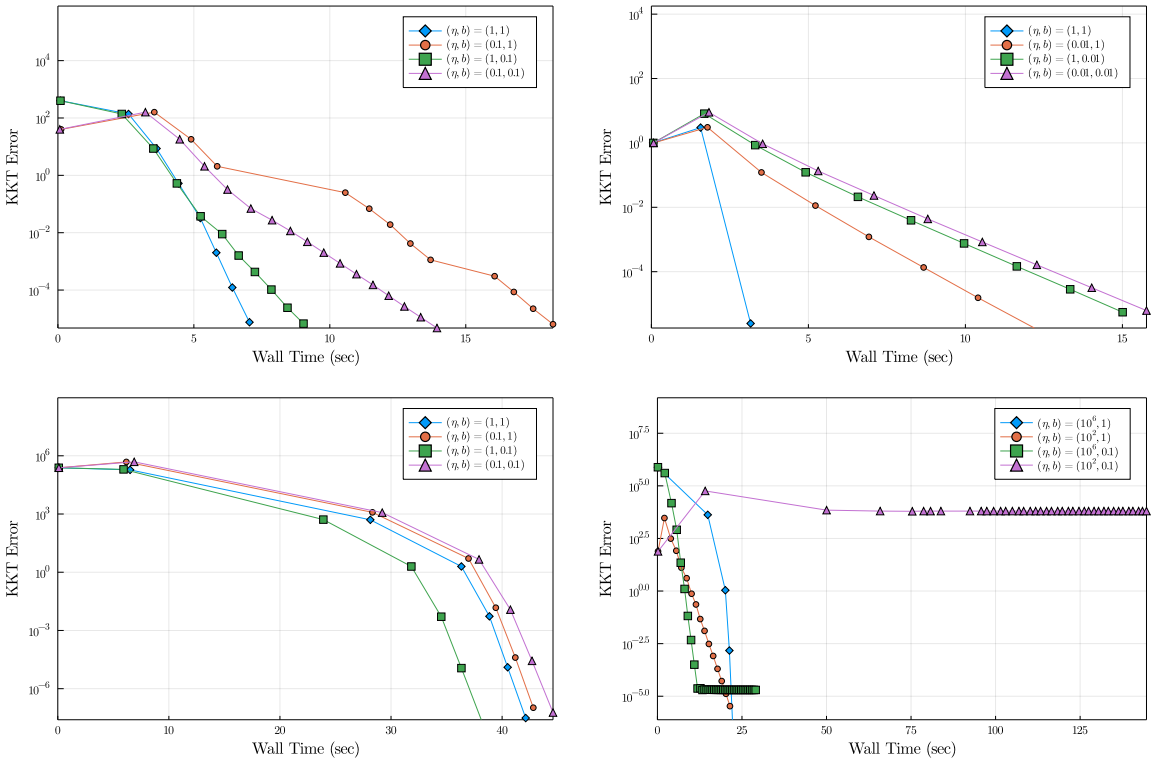

Figure 10.4 Convergence profiles (in wall times) of overlapipng Schwarz method for different regularizations. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

problems are around 10^5. As the solution time of the centralized solvers stiffly increase when the number of variables is greater than 10^5, we may see more benefits of decomposition when the problem size is larger. For such large-scale problems, even storing the problem data in a single computer can be challenging. However, our current implementation of OSM uses multi-thread parallelism. Thus, our implementation of OSM also suffers from such limitations in the memory. However, we emphasize that OSM can in principle, be implemented for distributed memories and run on clusters (see [153, Section V]). Some of these capabilities will be demonstrated in the next section. Therefore, with our current implementation, it was difficult to see that overlapping Schwarz is clearly faster than the

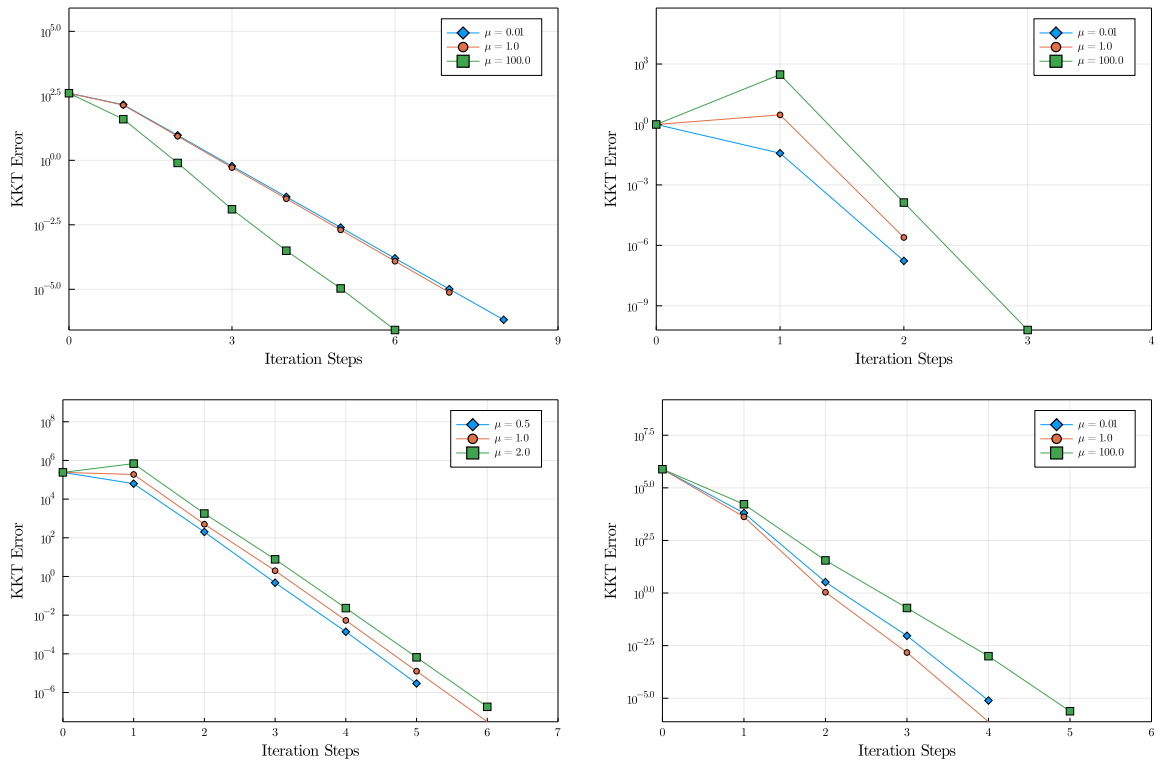

Figure 10.5 Convergence profiles (in iteration steps) of overlapipng Schwarz method for different penalty parameters. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

Table 10.1 Statistics for the problems

Problem	# Variables	# Constraints	# Nonzeros in Jac.	# Nonzeros in Hess.
Dynamic Opt.	156,009	108,009	516,009	276,009
Stochastic Opt.	531,440	265,720	797,158	531,440
PDE Opt.	320,400	160,000	958,800	320,000
Network Opt.	120,000	60,000	461,864	290,932

centralized methods, but we have seen that if we have a reasonably good initial guess, the OSM can indeed be effective.

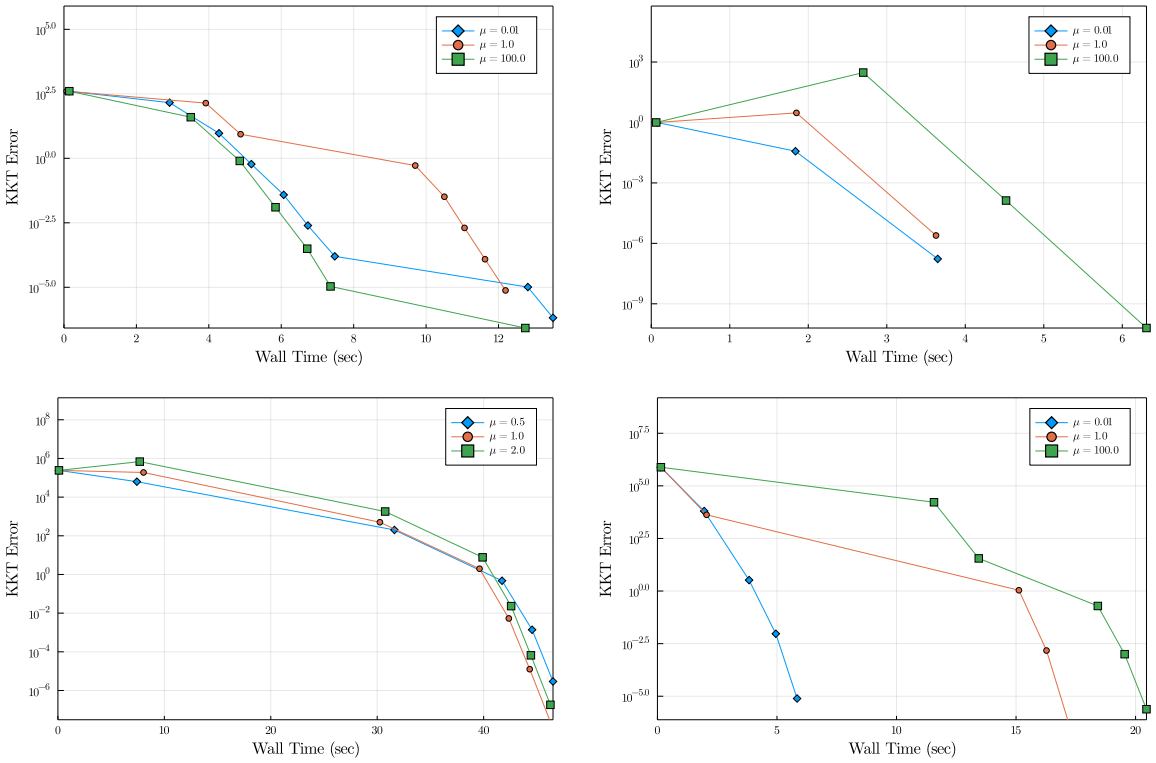

Figure 10.6 Convergence profiles (in wall times) of overlapipng Schwarz method for different penalty parameters. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

10.1.3 Additional Results: DC Power System State Estimtaion

Methods: In this case study, we apply the OSM to solve a DC PSSE problem (Appendix B.5), which is formulated as an unconstrained quadratic program (i.e., the solution of the positive definite linear system). OSM (8.3) is used as a spatial decomposition method for solving such a problem. In this implementation, the inverse of the block is explicitly computed and used for the subproblem solution. Our implementation of overlapping Schwarz uses the popular `MPICH` MPI library and the basic linear algebra subprograms package for matrix computation. The program was run on a cluster (four nodes and one CPU core Intel Xeon Processor E5-2695v4 per node). Compared to the multi-thread implementation in the

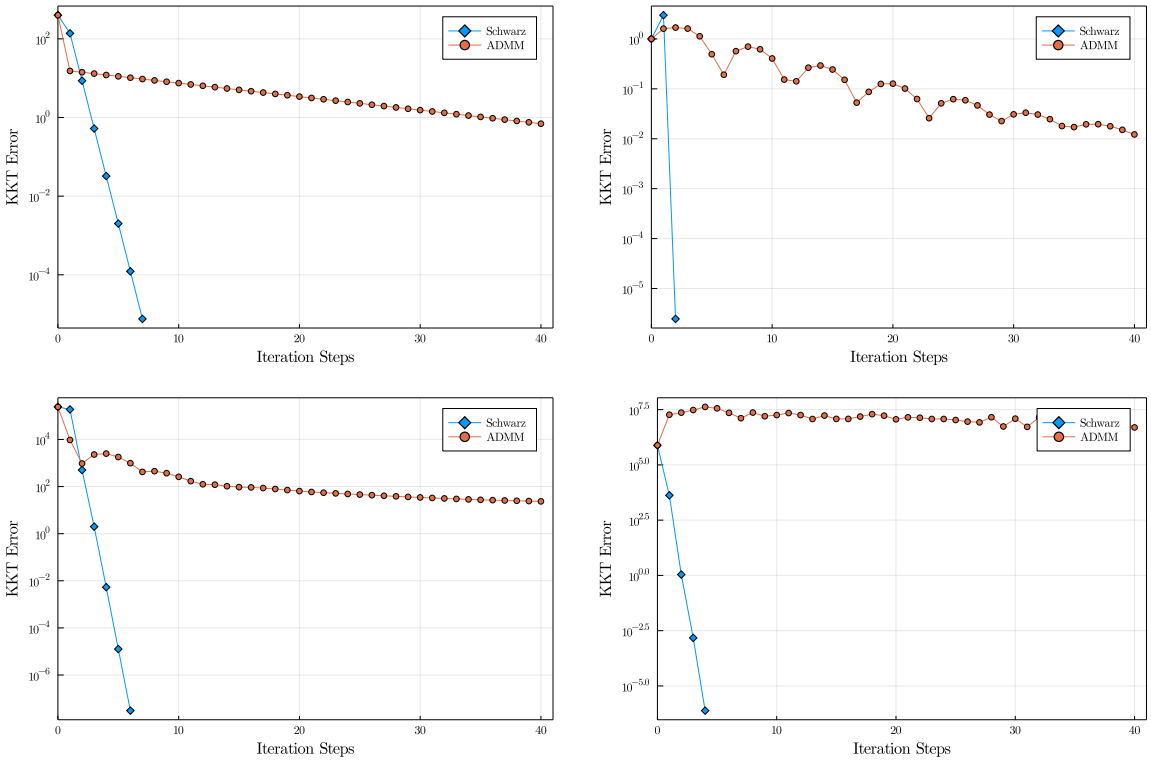

Figure 10.7 Convergence Profiles (in iteration steps) of overlapipng Schwarz Method and ADMM. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

previous section, the MPI-based implementation may cause more communication time, but it can run flexibly on any distributed architecture.

Results: In Figure 10.9, the residual is plotted with different sizes of the overlap and the regularization parameter. The results confirm that the solution converges linearly to the solution and that it converges faster as the size of overlap increases and as the conditioning of the problem improves. We also observe that the increase in the overlap reduces the number of iterations but not necessarily the solution time. The reason is that the increase in the overlap also increases the computation and communication cost. The overall computing cost increases with the size of the block (i.e., $|\mathcal{V}_k'|$), and the communication cost also tends to increase with the size of boundary (i.e., $|\mathcal{N}[\mathcal{V}_k'] \setminus \mathcal{V}_k'|$). We can also observe that the size

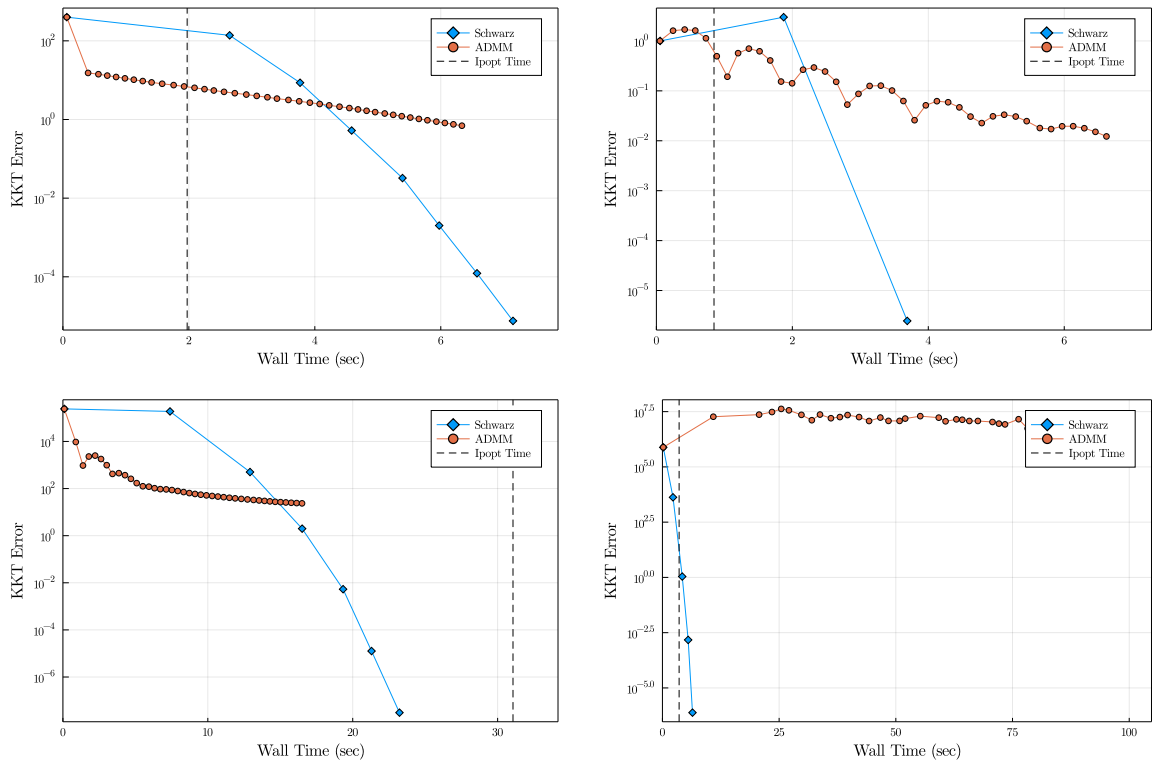

Figure 10.8 Convergence profiles (in wall times) of overlapipng Schwarz method and ADMM. Quadrotor motion planning (top left); stochastic storage control problem (top right); thin plate temperature control problem (bottom left); tracking optimal power flow problem (bottom right).

of overlapping blocks and the size of boundary increase with ω (Table 10.2). Such effects ultimately are manifested in the overall CPU time per iteration: 0.0963 sec/iter when $\omega = 1$, 0.140 sec/iter when $\omega = 2$, and 0.256 sec/iter when $\omega = 3$. These results again illustrate the trade-offs in convergence and computational performance. This case study demonstrates that overlapping Schwarz can be implemented for distributed memory settings, and can run on clusters.

10.2 Linear Algebra-Level Decomposition

In this section, we demonstrate the linear algebra-level decomposition method with two energy system optimization problems: transient gas network operation (Appendix (B.9)) and

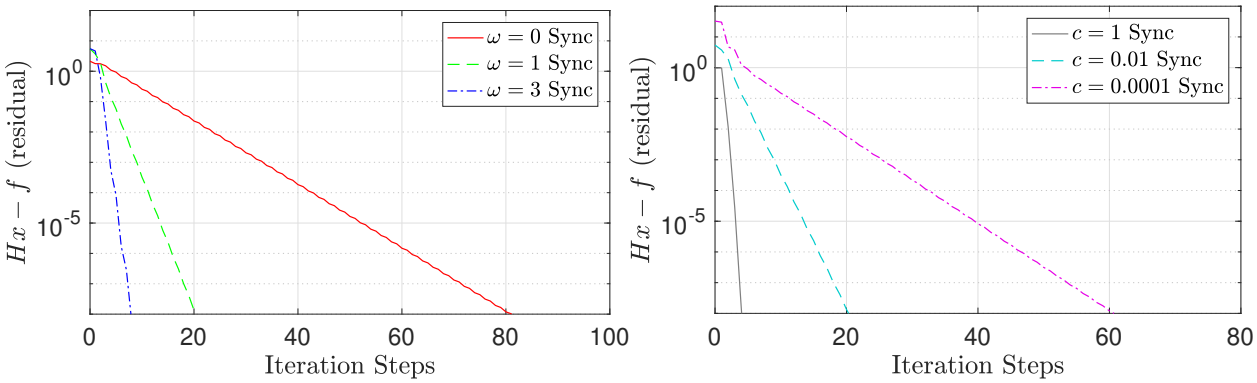

Figure 10.9 Left: Residual over iteration steps ($c = 0.1$); Right: Residual over iteration steps $\omega = 1$.

Table 10.2 Statistics for the subsystems

k	1	2	3	4	total.
$\lvert V_k \rvert$	2,324	2,366	2,278	2,273	9,241
$\lvert \mathcal{N}_{\mathcal{G}}^1[\mathcal{V}_k] \rvert$	2,361	2,398	2,291	2,304	9,354
$\lvert \mathcal{N}_{\mathcal{G}}^2[\mathcal{V}_k] \rvert$	2,452	2,469	2,322	2,376	9,619
$\lvert \mathcal{N}_{\mathcal{G}}^3[\mathcal{V}_k] \rvert$	2,570	2,558	2,380	2,506	10,014
$\lvert \mathcal{N}_{\mathcal{G}}^4[\mathcal{V}_k] \rvert$	2,744	2,680	2,485	2,727	10,636
$\lvert \mathcal{N}_{\mathcal{G}}^1[\mathcal{V}_k] \setminus \mathcal{V}_k \rvert$	37	32	13	31	113
$\lvert \mathcal{N}_{\mathcal{G}}^2[\mathcal{V}_k] \setminus \mathcal{N}_{\mathcal{G}}^1[\mathcal{V}_k] \rvert$	91	71	31	72	265
$\lvert \mathcal{N}_{\mathcal{G}}^3[\mathcal{V}_k] \setminus \mathcal{N}_{\mathcal{G}}^2[\mathcal{V}_k] \rvert$	118	89	58	130	395
$\lvert \mathcal{N}_{\mathcal{G}}^4[\mathcal{V}_k] \setminus \mathcal{N}_{\mathcal{G}}^3[\mathcal{V}_k] \rvert$	174	122	105	221	622

multi-period AC OPF (Appendix (B.8). Both of the problems are formulated as a network optimization problem over a time period (dynamic optimization). We use our graph-based modeling interface `Plasmo.jl`, interior point nonlinear optimization solver `MadNLP.jl`, and the OSM linear solver `Schwarz` submodule within `MadNLP.jl` to demonstrate the method.

10.2.1 Methods

We compare the proposed method (`MadNLP.jl` interfaced with `Plasmo.jl` and `Schwarz` linear solver), with the conventional method (`MadNLP.jl` interfaced with serial/parallel direct solvers `Ma57` or `MKL-Pardiso` along with non-graph based algebraic modeling language `JuMP.jl`). The conventional methods are referred to as JuMP-Ma57 and JuMP-PardisoMKL, and the proposed method is referred to as Plasmo-Schwarz/Ma57. Furthermore, the mix of proposed/conventional approaches (JuMP-Schwarz/Ma57, Plasmo-Ma57, and Plasmo-PardisoMKL) is also tested together. For JuMP-Schwarz/Ma57, the graph partitioning tool `METIS` was used to partition the primal-dual coupling graph (discussed in Remark 1.2). A Richardson scheme was used as an iterator for the RAS scheme. When the solver is interfaced with `Plasmo.jl`, the NLP function evaluations (derivatives, Jacobians, Hessians) are also parallelized. The study was performed by solving the gas (B.12) and power (B.11) problems while varying the size of the problems (by increasing the length of the prediction horizon). The code was run on a server computer equipped with 2 CPUs of Intel Xeon CPU E5-2698 v4 running 2.20GHz (20 core for each), and 20 threads are used for the computation. Code to reproduce the results can be found in https://github.com/zavalab/JuliaBox/tree/master/AdchemCaseStudy

10.2.2 Results

For both problems, we found that the graph-based approach can significantly accelerate the solution (see Figure 10.10). In particular, comparing JuMP-Ma57 and Plasmo-Schwarz/Ma57, Plasmo-Schwarz/Ma57 becomes faster than JuMP-Ma57 when the prediction horizon is 3 days or more. Function evaluations are always faster in `Plasmo.jl` compared to `JuMP.jl` because the computational savings from function evaluations directly reduce the total solution time (parallelizing the function evaluation itself has no impact on the other part of the algorithm). On the other hand, one can see that the speed-up from parallel linear algebra is only observed when the problem size is sufficiently large (3 days in the gas network and 60 days in the power network) because the reduction in the problem size also reduces

the overlap size. In our implementation, we set the size of overlap using the relative size of the block (the size of overlap is reduced if the overall problem size is reduced). As a result, the OSM scheme (8.6) may become slow, and the number of required factorization/backsolve steps increases. This indicates that the use of OSM is beneficial only when the problem size is sufficiently large. For the gas problems, the acceleration of linear algebra computations was more pronounced. In contrast, for the power problems, the acceleration of function evaluations was more pronounced because the AC power flow formulation has a large number of nonlinear expressions. By comparing the linear solver time for JuMP-Schwarz/Ma57 and Plasmo-Schwarz/Ma57, we see the advantage of using a graph-based modeling language for obtaining the partitions. We recall that for JuMP-Schwarz/Ma57, the `Metis` graph partitioning routine is directly applied to the primal-dual connectivity graph while Plasmo-Schwarz/Ma57 uses the user-provided problem graph. One can observe that, in general, the linear solver time is shorter for PlasmoNLP-Schwarz/Ma57. This indicates that the user-provided graph information can be leveraged for obtaining high-quality partitions.

From the results in this section, we can see that OSM can indeed be computationally effective than centralized solvers because the OSM at the linear algebra level enjoys global convergence. We could see that problem-level decompositions can also be effective for certain cases, but it needs to be used with cautions since the nonlinearity in the problem can cause convergence issues.

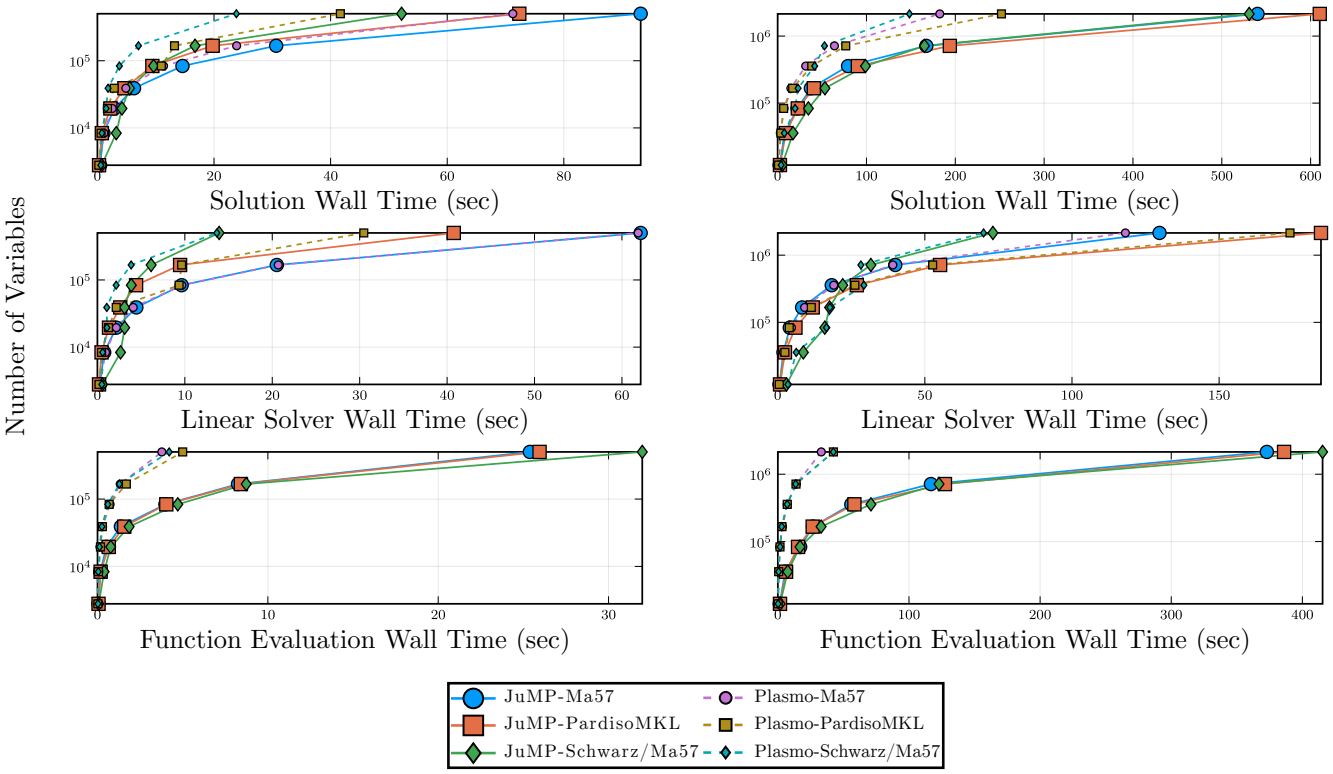

Figure 10.10 Solution time (top), linear solver time (middle), function evaluation time (bottom) for transient gas network (left) and multi-period AC optimal power flow (right) problems.

Chapter 11

Conclusions and Future Work

In this work, we have studied the properties and algorithms for graph-structured nonlinear programs. We have introduced the notion of graph-structured nonlinear programming, which is a generalized abstraction for a diverse class of structured optimization problems (dynamic optimization, stochastic optimization, PDE optimization, and network optimization). The formulation of graph-structured nonlinear programming is highly flexible, so a variety of practical optimization problems can be perceived as graph-structured nonlinear programs and studied under this unifying abstraction. Studying the problem under this abstraction is advantageous in that it allows elucidating the fundamental properties that graph-structured nonlinear programs share in common.

We have established a fundamental property of graph-structured nonlinear programs that we call exponential decay of sensitivity. This property states that under the regularity condition given by the strong second sufficiency condition and linear independence constraint qualifications, the impact of nodal parametric perturbation on the nodal solution decay exponentially with the distance from the perturbation point on the graph. This property explains how the impact of data perturbation propagates along with the graph and addresses lots of practical questions on the solution stability. Furthermore, we observe that the sensitivity decay rate is governed by the singular values of the primal-dual Lagrangian Hessian matrix (also commonly referred to as the KKT matrix). Moreover, we have found that those singular values can be uniformly bounded under the uniform regularity, which consists of uniformly bounded Lagrangian Hessian, uniform second-order sufficiency, linear

independence constraint qualifications. With numerical examples on dynamic, stochastic, PDE, and network optimization problems, we have demonstrated our theoretical results on the exponential decay of sensitivity.

We have found that the uniformly regular graph-structured nonlinear programs can be efficiently solved with the overlapping Schwarz decomposition method, which exploits the exponential decay of sensitivity by design. The overlapping Schwarz method has been traditionally used for solving discretized partial differential equations and sparse linear systems. We have generalized the overlapping Schwarz method so that it can be used for the distributed solution of graph-structured nonlinear programs. We have shown that the overlapping Schwarz method is locally convergent and the convergence rate exponentially improves with the size of overlap for mesh-like graph-structured nonlinear programs. Furthermore, for quadratic programming and linear system settings, we have shown global convergence under strengthened regularity conditions. We have discussed various levels at which the overlapping Schwarz method can be implemented: problem-level, sub-problem-level (within Augmented Lagrangian method and sequential quadratic programming method), and linear algebra level (within interior point method). We have numerically demonstrated that for a number of instances, the overlapping Schwarz method at the problem level can solve the problem faster than the existing decomposition method ADMM. Furthermore, the overlapping Schwarz decomposition at the linear algebra level can solve problems significantly faster than the state-of-the-art nonlinear programming solver, Ipopt. We expect that the benefit of the decomposition will be even greater for larger problem instances.

We believe this dissertation has shed light on a small portion of the science concerning graph-structured nonlinear programming, and there remains a wide range of unaddressed important research questions that might have theoretical and practical significance. In what follows, a few high-level, open research questions and future research plans are discussed.

EDS in Multi-Stage Stochastic Programs: Our EDS result is based on the assumption that the smallest eigenvalue of the reduced Hessian is uniformly bounded away from zero (uSSOSC). In the context of multi-stage stochastic optimization, this assumption may be

violated if the width (the number of nodes per stage) of the scenario tree is unbounded. The reason is that the probability is multiplied by the nodal objective function term, and the probability decays to zero as we move down the tree. Thus, our method in Chapter 2-3 is not directly applicable to multistage stochastic optimization problems with unbounded width (in particular, the uniform regularity condition cannot be established based on the composability principles in the current form). In future work, we will address this issue and establish the EDS for multistage stochastic programs. Specifically, we will first formally investigate whether multistage stochastic optimization problems with bounded width (but unbounded depth) exhibit EDS. Such problems arise from stochastic control problems with robust horizon assumptions [115]. To prove this, we first partition the problem by a non-growing part (where branching occurs) and multiple growing parts (where branching does not occur). One can observe that each growing part reduces to a deterministic dynamic optimization trajectory. The uniform regularity conditions for the growing parts can be proved by using the results in Chapter 4 (controllability and observability imply uSSOSC and uLICQ [152]). One can then use the composability argument to show the uniform regularity conditions for the full problem. We will try to generalize this result to show the EDS for trees with unbounded width. We expect that showing LICQ from our existing composability principles will be straightforward, but we need a more sophisticated technique to show uSSOSC (or a comparable regularity condition).

EDS in Infinite Graph-Structured Nonlinear Programs: Extending the current EDS results to infinite-graph problems is of interest. Note that we have shown EDS for problems with arbitrarily large graphs, but the graph itself was always finite. One of the particularly interesting examples is infinite-horizon dynamic optimization problems. Showing EDS for infinite-horizon problems will allow characterizing the closed-loop regret of finite-horizon dynamic programs and analyzing the infinite horizon value function. For instance, it has been recently shown that the regret (the difference between the closed-loop objective cost of finite-horizon control policies and the infinite horizon objective) of finite-horizon linear-quadratic problems decays exponentially with the horizon length [173]; with the EDS result

for infinite-horizon dynamic programs, we may establish a comparable result for nonlinear systems. For LQR, we expect that extending our result will be straightforward since the effect of truncation can be easily cast as a bounded parametric perturbation (e.g., by specifying terminal cost as the adjoint solution of the infinite-horizon problem). For the nonlinear case, however, the extension may not be straightforward. The reason is that for nonlinear cases, the sensitivity result [146] holds only within the neighborhood of the reference data, but the size of such neighborhood is not theoretically characterized. In particular, to establish the desired result, one must show that the radius of the neighborhood is uniformly bounded below. We hypothesize that we can show the uniform boundedness in the size of the neighborhood and use this result to show the EDS for infinite-horizon dynamic programs.

EDS in Continuous-Domain Optimization Problems: Many graph-structured optimization problems are obtained as a result of the discretization of continuous-domain optimization problems; examples include DAE and PDE-constrained optimization problems. Recently, the exponential decay of sensitivity was studied in the context of continuous-time (infinite-dimensional) optimization problems [82–84]. In the future, it is of interest to connect this result with our notion of EDS for gsNLPs. We will first investigate if there is a limiting behavior of EDS under the mesh refinement setting. This will require properly redefining the EDS and the magnitudes of the perturbations. From such a limiting behavior, we will make connections with the continuous-time optimization problems.

Overlapping Schwarz within ALM and SQP methods: In Chapter 9, we have briefly discussed the possibility of applying the overlapping Schwarz method within the sequential quadratic programming method. However, we have not presented any numerical results. In the future, it is of interest to numerically demonstrate this idea and compare it against problem-level and linear algebra-level decomposition methods.

Implementation of Distributed Overlapping Schwarz: The numerical results in Section 10.2 reveals that in order to solve the huge nonlinear programs, where decomposition is computationally more favorable than centralized method, the implementation of overlapping Schwarz based on distributed parallelism is necessary. This can be implemented via either

MPI (the one we used for the DC PSSE problem) or native distributed parallelism in Julia (`Distributed.jl`). In the future, we will implement the distributed overlapping Schwarz solver to facilitate solving such large-scale problems.

Regularity Maximization: The discussion in Chapter 3 reveals that the singular value of the constraint Jacobian of the gsNLP plays a key role in determining *flexibility* of the system. In particular, we have shown that if the smallest singular value of the constraint Jacobian is close to zero, the system may exhibit extreme sensitivity to disturbances, and the disturbances may have a far-reaching impact within the network. Thus, it is important to design the system in a way that the smallest singular value in the constraint Jacobian is sufficiently bounded away from zero. This motivates studying the optimal design problem that maximizes the smallest singular value of the constraint Jacobian while jointly considering the cost of installations. In such a design problem, multiple candidates of actuators are considered (in the context of energy infrastructure, these can either be energy storages or generators); and the problem seeks for an optimal combination of such actuators that maximizes the flexibility (to be more specific, the smallest singular value of the constraint Jacobian is maximized). Our preliminary result suggests that this problem reduces to a standard mixed-integer semidefinite program [70] if the system is linear. On the other hand, for nonlinear cases, we expect that more sophisticated algorithms and relaxation techniques are required [155]. In the future, we will study the formulations and algorithms for these problems and apply the developed methods to critical infrastructure design problems.

LIST OF REFERENCES

[1] MadNLP.jl. https://github.com/zavalab/madnlp.jl, 2021.

[2] Plasmo.jl. https://github.com/zavalab/plasmo.jl, 2021.

[3] SimpleADMM.jl. https://github.com/sshin23/simpleadmm.jl, 2021.

[4] SimpleNL.jl. https://github.com/sshin23/simplenl.jl, 2021.

[5] SimpleSchwarz.jl. https://github.com/sshin23/simpleschwarz.jl, 2021.

[6] Ali Abur and Antonio Gomez Exposito. *Power system state estimation: theory and implementation*. CRC press, 2004.

[7] David P Ahlfeld, John M Mulvey, Ron S Dembo, and Stavros A Zenios. Nonlinear programming on generalized networks. *ACM Transactions on Mathematical Software (TOMS)*, 13(4):350–367, 1987.

[8] Ravindra K Ahuja, Thomas L Magnanti, James B Orlin, and MR Reddy. Applications of network optimization. *Handbooks in Operations Research and Management Science*, 7:1–83, 1995.

[9] Andrew Allman, Wentao Tang, and Prodromos Daoutidis. DeCODe: a community-based algorithm for generating high-quality decompositions of optimization problems. *Optimization and Engineering*, 20(4):1067–1084, 2019.

[10] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[11] Sogol Babaeinejadsarookolaee, Adam Birchfield, Richard D Christie, Carleton Coffrin, Christopher DeMarco, Ruisheng Diao, Michael Ferris, Stephane Fliscounakis, Scott Greene, Renke Huang, et al. The power grid library for benchmarking AC optimal power flow algorithms. *arXiv preprint arXiv:1908.02788*, 2019.

[12] Satish Balay, Shrirang Abhyankar, Mark Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Alp Dener, Victor Eijkhout, W Gropp, et al. PETSc users manual. 2019.

[13] Gulay Barbarosoğlu and Yasemin Arda. A two-stage stochastic programming framework for transportation planning in disaster response. *Journal of the operational research society*, 55(1):43–53, 2004.

[14] Russell Bent, Kaarthik Sundar, and David Fobes. GasModels.jl. https://github.com/lanl-ansi/GasModels.jl, 2020.

[15] Dennis S Bernstein. *Matrix mathematics: theory, facts, and formulas*. Princeton university press, 2009.

[16] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[17] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

[18] John T Betts. *Practical methods for optimal control and estimation using nonlinear programming*, volume 19. Siam, 2010.

[19] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, 59(1):65–98, 2017.

[20] Lorenz T Biegler. *Nonlinear programming: concepts, algorithms, and applications to chemical processes*, volume 10. Siam, 2010.

[21] Lorenz T Biegler. A survey on sensitivity-based nonlinear model predictive control. *IFAC Proceedings Volumes*, 46(32):499–510, 2013.

[22] Lorenz T Biegler, Omar Ghattas, Matthias Heinkenschloss, David Keyes, and Bart van Bloemen Waanders. *Real-time PDE-constrained Optimization*. SIAM, 2007.

[23] Lorenz T Biegler, Omar Ghattas, Matthias Heinkenschloss, and Bart van Bloemen Waanders. Large-scale PDE-constrained optimization: an introduction. In *Large-Scale PDE-Constrained Optimization*, pages 3–13. Springer, 2003.

[24] Adam B Birchfield, Ti Xu, Kathleen M Gegner, Komal S Shetye, and Thomas J Overbye. Grid structural characteristics as validation criteria for synthetic networks. *IEEE Transactions on power systems*, 32(4):3258–3265, 2016.

[25] John R Birge and Francois V Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988.

[26] J Frédéric Bonnans and Alexander Shapiro. *Perturbation analysis of optimization problems*. Springer Science & Business Media, 2013.

[27] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[28] Xiao-Chuan Cai, Charbel Farhat, and Marcus Sarkis. A minimum overlap restricted additive schwarz preconditioner and applications in 3D flow simulations. *Contemporary mathematics*, 218:479–485, 1998.

[29] Xiao-Chuan Cai and Yousef Saad. Overlapping domain decomposition algorithms for general sparse matrices. *Numerical linear algebra with applications*, 3(3):221–237, 1996.

[30] Xiao-Chuan Cai and Marcus Sarkis. A restricted additive schwarz preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21(2):792–797, 1999.

[31] Xiao-Chuan Cai and Olof B Widlund. Domain decomposition algorithms for indefinite elliptic problems. *SIAM Journal on Scientific and Statistical Computing*, 13(1):243–258, 1992.

[32] Tony F Chan and Barry F Smith. Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes. *Electronic Transactions on Numerical Analysis*, 2:171–182, 1994.

[33] Tony F Chan and Jun Zou. Additive schwarz domain decomposition methods for elliptic problems on unstructured meshes. *Numerical Algorithms*, 8(2):329–346, 1994.

[34] Chi-Tsong Chen and Chi-Tsong Chen. *Linear system theory and design*, volume 301. Holt, Rinehart and Winston New York, 1984.

[35] Nai-Yuan Chiang and Victor M Zavala. An inertia-free filter line-search algorithm for large-scale nonlinear programming. *Computational Optimization and Applications*, 64(2):327–354, 2016.

[36] Naiyuan Chiang, Cosmin G Petra, and Victor M Zavala. Structured nonconvex optimization of large-scale energy systems using PIPS-NLP. In *2014 Power Systems Computation Conference*, pages 1–7. IEEE, 2014.

[37] Jaquelin Cochran, Mackay Miller, Owen Zinaman, Michael Milligan, Doug Arent, Bryan Palmintier, Mark O'Malley, Simon Mueller, Eamonn Lannoye, Aidan Tuohy, et al. Flexibility in 21st century power systems. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2014.

[38] Carleton Coffrin, Russell Bent, Kaarthik Sundar, Yeesian Ng, and Miles Lubin. PowerModels. jl: An open-source framework for exploring power flow formulations. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–8. IEEE, 2018.

[39] Carleton Coffrin, Hassan L Hijazi, and Pascal Van Hentenryck. The qc relaxation: A theoretical and computational study on optimal power flow. *IEEE Transactions on Power Systems*, 31(4):3008–3018, 2015.

[40] Antonio J. Conejo, Enrique Castillo, Roberto Mínguez, and Raquel García-Bertrand. *Decomposition techniques in mathematical programming: Engineering and science applications*. 2006.

[41] Christian Conte, Tyler Summers, Melanie N Zeilinger, Manfred Morari, and Colin N Jones. Computational aspects of distributed optimization in model predictive control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6819–6824. IEEE, 2012.

[42] GAMS Development Corporation. General Algebraic Modeling System (GAMS) Release 24.2.1. Washington, DC, USA, 2013.

[43] Frank E Curtis, Johannes Huber, Olaf Schenk, and Andreas Wächter. A note on the implementation of an interior-point algorithm for nonlinear optimization with inexact step computations. *Mathematical programming*, 136(1):209–227, 2012.

[44] Prodromos Daoutidis, Wentao Tang, and Sujit S Jogwar. Decomposing complex plants for distributed control: Perspectives from network theory. *Computers & Chemical Engineering*, 114:43–51, 2018.

[45] Juan Carlos De los Reyes. *Numerical PDE-constrained optimization*. Springer, 2015.

[46] Ron S Dembo, John M Mulvey, and Stavros A Zenios. OR practice-large-scale nonlinear network models and their application. *Operations Research*, 37(3):353–372, 1989.

[47] Stephen Demko, William F Moss, and Philip W Smith. Decay rates for inverses of band matrices. *Mathematics of computation*, 43(168):491–499, 1984.

[48] Haoyang Deng and Toshiyuki Ohtsuka. A parallel newton-type method for nonlinear model predictive control. *Automatica*, 109:108560, 2019.

[49] Moritz Diehl, H Georg Bock, Johannes P Schlöder, Rolf Findeisen, Zoltan Nagy, and Frank Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[50] Victorita Dolean, Pierre Jolivet, and Frâdâric Nataf. *An introduction to domain decomposition methods: algorithms, theory, and parallel implementation*, volume 144. SIAM, 2015.

[51] Asen L Dontchev and R Tyrrell Rockafellar. *Implicit functions and solution mappings*, volume 543. Springer, 2009.

[52] Maksymilian Dryja and Olof Widlund. *An additive variant of the Schwarz alternating method for the case of many subregions*. Ultracomputer Research Laboratory, Univ., Courant Inst. of Mathematical Sciences, Division of Computer Science, 1987.

[53] Maksymilian Dryja and Olof B Widlund. Domain decomposition algorithms with small overlap. *SIAM Journal on Scientific Computing*, 15(3):604–620, 1994.

[54] Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[55] Evridiki Efstathiou and Martin J Gander. Why restricted additive schwarz converges faster than additive schwarz. *BIT Numerical Mathematics*, 43(5):945–959, 2003.

[56] Alexander Engelmann, Yuning Jiang, Boris Houska, and Timm Faulwasser. Decomposition of non-convex optimization via bi-level distributed ALADIN. *IEEE Transactions on Control of Network Systems*, 2020.

[57] Alexander Engelmann, Yuning Jiang, Tillmann Mühlpfordt, Boris Houska, and Timm Faulwasser. Toward distributed opf using ALADIN. *IEEE Transactions on Power Systems*, 34(1):584–594, 2018.

[58] Tomaso Erseghe. Distributed optimal power flow using ADMM. *IEEE transactions on power systems*, 29(5):2370–2380, 2014.

[59] Alessandro Falsone, Ivano Notarnicola, Giuseppe Notarstefano, and Maria Prandini. Tracking-ADMM for distributed constraint-coupled optimization. *Automatica*, 117:108962, 2020.

[60] Timm Faulwasser, Lars Grüne, Matthias A Müller, et al. *Economic nonlinear model predictive control*. Now Foundations and Trends, 2018.

[61] Timm Faulwasser, Milan Korda, Colin N Jones, and Dominique Bonvin. On turnpike and dissipativity properties of continuous-time optimal control problems. *Automatica*, 81:297–304, 2017.

[62] Timm Faulwasser and Mario Zanon. Asymptotic stability of economic NMPC: The importance of adjoints. *IFAC-PapersOnLine*, 51(20):157–168, 2018.

[63] Anthony V Fiacco. Sensitivity analysis for nonlinear programming using penalty methods. *Mathematical programming*, 10(1):287–311, 1976.

[64] Anthony V Fiacco and Yo Ishizuka. Sensitivity and stability analysis for nonlinear programming. *Annals of Operations Research*, 27(1):215–235, 1990.

[65] Stéphane Fliscounakis, Patrick Panciatici, Florin Capitanescu, and Louis Wehenkel. Contingency ranking with respect to overloads in very large power systems taking into account uncertainty, preventive, and corrective actions. *IEEE Transactions on Power Systems*, 28(4):4909–4917, 2013.

[66] Robert Fourer, David M Gay, and Brian W Kernighan. *AMPL: A mathematical programming language*. AT & T Bell Laboratories Murray Hill, NJ, 1987.

[67] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. Optimal power flow: a bibliographic survey I. *Energy Systems*, 3(3):221–258, 2012.

[68] Andreas Frommer and Daniel B Szyld. Weighted max norms, splittings, and overlapping additive schwarz iterations. *Numerische Mathematik*, 83(2):259–278, 1999.

[69] Andreas Frommer and Daniel B Szyld. An algebraic convergence theory for restricted additive schwarz methods using weighted max norms. *SIAM journal on numerical analysis*, 39(2):463–479, 2001.

[70] Tristan Gally, Marc E Pfetsch, and Stefan Ulbrich. A framework for solving mixed-integer semidefinite programs. *Optimization Methods and Software*, 33(3):594–632,.

[71] Martin J Gander and Soheil Hajian. Block jacobi for discontinuous galerkin discretizations: no ordinary schwarz methods. In *Domain Decomposition Methods in Science and Engineering XXI*, pages 305–313. Springer, 2014.

[72] Philipp Gerstner, Michael Schick, Vincent Heuveline, Nico Meyer-Hübner, Michael Suriyah, Thomas Leibfried, Viktor Slednev, Wolf Fichtner, and Valentin Valentin Bertsch. A domain decomposition approach for solving dynamic optimal power flow problems in parallel with application to the german transmission grid. *Preprint Series of the Engineering Mathematics and Computing Lab*, (1), 2016.

[73] Frederik Geth, Carleton Coffrin, and David M Fobes. A flexible storage model for power network optimization. *arXiv preprint arXiv:2004.14768*, 2020.

[74] Pontus Giselsson and Anders Rantzer. Distributed model predictive control with suboptimality and stability guarantees. In *49th IEEE Conference on Decision and Control (CDC)*, pages 7272–7277. IEEE, 2010.

[75] Pontus Giselsson and Anders Rantzer. On feasibility, stability and performance in distributed model predictive control. *IEEE Transactions on Automatic Control*, 59(4):1031–1036, 2013.

[76] Pontus Giselsson and Anders Rantzer. Generalized accelerated gradient methods for distributed MPC based on dual decomposition. In *Distributed Model Predictive Control Made Easy*, pages 309–325. Springer, 2014.

[77] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2012.

[78] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[79] Ignacio E. Grossmann. Advances in mathematical programming models for enterprise-wide optimization. *Computers and Chemical Engineering*, 47:2–18, 2012.

[80] Ignacio E Grossmann and M Morari. Operability, resiliency, and flexibility: Process design objectives for a changing world. 1983.

[81] Lars Grüne and Matthias A Müller. On the relation between strict dissipativity and turnpike properties. *Systems & Control Letters*, 90:45–53, 2016.

[82] Lars Grüne, Manuel Schaller, and Anton Schiela. Sensitivity analysis of optimal control for a class of parabolic PDEs motivated by model predictive control. *SIAM Journal on Control and Optimization*, 57(4):2753–2774, 2019.

[83] Lars Grüne, Manuel Schaller, and Anton Schiela. Abstract nonlinear sensitivity and turnpike analysis and an application to semilinear parabolic PDEs. *arXiv preprint arXiv:2008.13001*, 2020.

[84] Lars Grüne, Manuel Schaller, and Anton Schiela. Exponential sensitivity and turnpike analysis for linear quadratic optimal control of general evolution equations. *Journal of Differential Equations*, 268(12):7311–7341, 2020.

[85] William E. Hart, Carl D. Laird, Jean-Paul Watson, David L. Woodruff, Gabriel A. Hackebeil, Bethany L. Nicholson, and John D. Siirola. *Pyomo–optimization modeling in python*, volume 67. Springer Science & Business Media, second edition, 2017.

[86] Markus Hehn and Raffaello D'Andrea. A flying inverted pendulum. In *2011 IEEE International Conference on Robotics and Automation*, pages 763–770. IEEE, 2011.

[87] Roland Herzog and Karl Kunisch. Algorithms for PDE-constrained optimization. *GAMM-Mitteilungen*, 33(2):163–176, 2010.

[88] Michael Hinze, René Pinnau, Michael Ulbrich, and Stefan Ulbrich. *Optimization with PDE constraints*, volume 23. Springer Science & Business Media, 2008.

[89] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

[90] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[91] Boris Houska, Janick Frasch, and Moritz Diehl. An augmented lagrangian based algorithm for distributed nonconvex optimization. *SIAM Journal on Optimization*, 26(2):1101–1127, 2016.

[92] A HSL. collection of fortran codes for large-scale scientific computation. *See http://www. hsl. rl. ac. uk*, 2007.

[93] Kai Huang and Shabbir Ahmed. The value of multistage stochastic programming in capacity planning under uncertainty. *Operations Research*, 57(4):893–904, 2009.

[94] Jens Hübner, Martin Schmidt, and Marc C Steinbach. Optimization techniques for tree-structured nonlinear problems. *Computational Management Science*, 17(3):409–436, 2020.

[95] M Huneault and FD Galiana. A survey of the optimal power flow literature. *IEEE transactions on Power Systems*, 6(2):762–770, 1991.

[96] Jennifer R Jackson and Ignacio E Grossmann. Temporal decomposition scheme for nonlinear multisite production planning and distribution models. *Industrial & engineering chemistry research*, 42(13):3045–3055, 2003.

[97] Jordan Jalving, Yankai Cao, and Victor M Zavala. Graph-based modeling and simulation of complex systems. *Computers & Chemical Engineering*, 125:134–154, 2019.

[98] Jordan Jalving, Sungho Shin, and Victor M Zavala. A graph-based modeling abstraction for optimization: Concepts and implementation in Plasmo.jl. *arXiv preprint arXiv:2006.05378*, 2020.

[99] Jordan H Jalving. *Graph-Based Modeling and Simulation of Cyber-Physical Systems*. The University of Wisconsin-Madison, 2020.

[100] Yuning Jiang, Dimitris Kouzoupis, Haoyu Yin, Moritz Diehl, and Boris Houska. Decentralized optimization over tree graphs. *Journal of Optimization Theory and Applications*, pages 1–24, 2021.

[101] Mark T Jones and Daniel B Szyld. Two-stage multisplitting methods with overlapping blocks. *Numerical linear algebra with applications*, 3(2):113–124, 1996.

[102] Jia Kang, Yankai Cao, Daniel P. Word, and C. D. Laird. An interior-point method for efficient solution of block-structured NLP problems using an implicit Schur-complement decomposition. *Computers and Chemical Engineering*, 2014.

[103] George Karypis and Vipin Kumar. METIS–unstructured graph partitioning and sparse matrix ordering system, version 2.0. 1995.

[104] Masoumeh Kazemi Zanjani, Mustapha Nourelfath, and Daoud Ait-Kadi. A multi-stage stochastic programming approach for production planning with uncertainty in the quality of raw materials and demand. *International Journal of Production Research*, 48(16):4701–4723, 2010.

[105] Youngdae Kim and Mihai Anitescu. A real-time optimization with warm-start of multiperiod AC optimal power flows. *Electric Power Systems Research*, 189:106721, 2020.

[106] Dimitris Kouzoupis, Rien Quirynen, Boris Houska, and Moritz Diehl. A block based ALADIN scheme for highly parallelizable direct optimal control. In *2016 American Control Conference (ACC)*, pages 1124–1129. IEEE, 2016.

[107] Attila Kozma, Christian Conte, and Moritz Diehl. Benchmarking large-scale distributed convex quadratic programming algorithms. *Optimization Methods and Software*, 30(1):191–214, 2015.

[108] Ranjeet Kumar, Michael J Wenzel, Matthew J Ellis, Mohammad N ElBsat, Kirk H Drees, and Victor M Zavala. A stochastic model predictive control framework for stationary battery systems. *IEEE Transactions on Power Systems*, 33(4):4397–4406, 2018.

[109] Forrest Laine and Claire Tomlin. Parallelizing lqr computation through endpoint-explicit riccati recursion. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 1395–1402. IEEE, 2019.

[110] Eamonn Lannoye, Damian Flynn, and Mark O'Malley. Evaluation of power system flexibility. *IEEE Transactions on Power Systems*, 27(2):922–931, 2012.

[111] Claude Lemaréchal. Lagrangian relaxation. In *Computational combinatorial optimization*, pages 112–156. Springer, 2001.

[112] Yingying Li, Xin Chen, and Na Li. Online optimal control with linear dynamics and predictions: Algorithms and regret analysis. In *Advances in Neural Information Processing Systems*, pages 14887–14899, 2019.

[113] Yingying Li and Na Li. Online learning for markov decision processes in nonstationary environments: A dynamic regret analysis. In *2019 American Control Conference (ACC)*, pages 1232–1237. IEEE, 2019.

[114] YP Li, GH Huang, and SL Nie. An interval-parameter multi-stage stochastic programming model for water resources management under uncertainty. *Advances in Water Resources*, 29(5):776–789, 2006.

[115] Sergio Lucia, Tiago Finkler, and Sebastian Engell. Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of process control*, 23(9):1306–1319, 2013.

[116] Tarek Mathew. *Domain decomposition methods for the numerical solution of partial differential equations*, volume 61. Springer Science & Business Media, 2008.

[117] MathWorks. Nonlinear heat transfer in thin plate. https://www.mathworks.com/help/pde/ug/nonlinear-heat-transfer-in-a-thin-plate.html.

[118] Alcir Monticelli. Electric power system state estimation. *Proceedings of the IEEE*, 88(2):262–282, 2000.

[119] Sen Na and Mihai Anitescu. Exponential decay in the sensitivity analysis of nonlinear dynamic programming. *To appear in SIAM Journal on Optimization*, 2019.

[120] Sen Na and Mihai Anitescu. Superconvergence of online optimization for model predictive control. *arXiv preprint arXiv:2001.03707*, 2020.

[121] Sen Na, Sungho Shin, Mihai Anitescu, and Victor M. Zavala. Overlapping schwarz decomposition for nonlinear optimal control. 2020. Under Review.

[122] Isak Nielsen and Daniel Axehill. An $O(\log N)$ parallel algorithm for newton step computation in model predictive control. *IFAC Proceedings Volumes*, 47(3):10505–10511, 2014.

[123] Isak Nielsen and Daniel Axehill. A parallel structure exploiting factorization algorithm with applications to model predictive control. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 3932–3938. IEEE, 2015.

[124] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[125] Hussam Nosair and Franois Bouffard. Flexibility envelopes for power system operational planning. *IEEE Transactions on Sustainable Energy*, 6(3):800–809, 2015.

[126] Mario VF Pereira and Leontina MVG Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical programming*, 52(1-3):359–375, 1991.

[127] Alfio Quarteroni and Alberto Valli. Domain decomposition methods for partial differential equations numerical mathematics and scientific computation. *Quarteroni, A. Valli–New York: Oxford University Press.–1999*, 1999.

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

[129] Christopher V Rao, James B Rawlings, and Jay H Lee. Constrained linear state estimation—a moving horizon approach. *Automatica*, 37(10):1619–1628, 2001.

[130] James Blake Rawlings, David Q Mayne, and Moritz Diehl. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI, 2017.

[131] Felix Rey, Peter Hokayem, and John Lygeros. ADMM for exploiting structure in MPC problems. *IEEE Transactions on Automatic Control*, 2020.

[132] Stephen M Robinson. Perturbed Kuhn-Tucker points and rates of convergence for a class of nonlinear-programming algorithms. *Mathematical programming*, 7(1):1–16, 1974.

[133] Stephen M Robinson. Generalized equations and their solutions, Part I: Basic theory. In *Point-to-Set Maps and Mathematical Programming*, pages 128–141. Springer, 1979.

[134] Stephen M Robinson. Strongly regular generalized equations. *Mathematics of Operations Research*, 5(1):43–62, 1980.

[135] Stephen M Robinson. Generalized equations and their solutions, Part II: Applications to nonlinear programming. In *Optimality and Stability in Mathematical Programming*, pages 200–221. Springer, 1982.

[136] Jose S Rodriguez, Carl D Laird, and Victor M Zavala. Scalable preconditioning of block-structured linear algebra systems using ADMM. *Computers & Chemical Engineering*, 133:106478, 2020.

[137] Jose S Rodriguez, Bethany Nicholson, Carl Laird, and Victor M Zavala. Benchmarking ADMM in nonconvex NLPs. *Computers & Chemical Engineering*, 119:315–325, 2018.

[138] Youcef Saad and Martin H Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869, 1986.

[139] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[140] Hermann Amandus Schwarz. *Ueber einen Grenzübergang durch alternirendes Verfahren*. Zürcher u. Furrer, 1870.

[141] Fred C Schweppe and J Wildes. Power system static-state estimation, Part I: Exact model. *IEEE Transactions on Power Apparatus and systems*, (1):120–125, 1970.

[142] James Fairbanks Seth Bromberger and other contributors. JuliaGraphs/LightGraphs.jl: an optimized graphs package for the Julia programming language, 2017.

[143] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

[144] S. Shin, T. Faulwasser, M. Zanon, and V. M. Zavala. A parallel decomposition scheme for solving long-horizon optimal control problems. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 5264–5271, 2019.

[145] Sungho Shin, Mihai Anitescu, and Victor M Zavala. Overlapping schwarz decomposition for constrained quadratic programs. *arXiv preprint arXiv:2003.07502*, 2020.

[146] Sungho Shin, Mihai Anitescu, and Victor M. Zavala. Exponential decay of sensitivity in graph-structured nonlinear programs. *arXiv preprint arXiv:2101.03067v1*, 2021.

[147] Sungho Shin, Carleton Coffrin, Kaarthik Sundar, and Victor M Zavala. Graph-based modeling and decomposition of energy infrastructures. *arXiv preprint arXiv:2010.02404*, 2020.

[148] Sungho Shin, Philip Hart, Thomas Jahns, and Victor M Zavala. A hierarchical optimization architecture for large-scale power networks. *IEEE Transactions on Control of Network Systems*, 2019.

[149] Sungho Shin, Ophelia S Venturelli, and Victor M Zavala. Scalable nonlinear programming framework for parameter estimation in dynamic biological system models. *PLoS computational biology*, 15(3):e1006828, 2019.

[150] Sungho Shin and Victor M Zavala. Multi-grid schemes for multi-scale coordination of energy systems. In *Energy Markets and Responsive Grids*, pages 195–222. Springer, 2018.

[151] Sungho Shin and Victor M Zavala. Diffusing-horizon model predictive control. *arXiv preprint arXiv:2002.08556*, 2020.

[152] Sungho Shin and Victor M Zavala. Controllability and observability imply exponential decay of sensitivity in dynamic optimization. *arXiv preprint arXiv:2101.06350*, 2021.

[153] Sungho Shin, Victor M Zavala, and Mihai Anitescu. Decentralized schemes with overlap for solving graph-structured optimization problems. *IEEE Transactions on Control of Network Systems*, 2020.

[154] Amik St-Cyr, Martin J Gander, and Stephen J Thomas. Optimized multiplicative, additive, and restricted additive schwarz preconditioning. *SIAM Journal on Scientific Computing*, 29(6):2402–2425, 2007.

[155] Defeng Sun, Jie Sun, and Liwei Zhang. The rate of convergence of the augmented Lagrangian method for nonlinear semidefinite programming. *Mathematical Programming*, 114(2):349–391, 2008.

[156] Junjie Sun and Leigh Tesfatsion. DC optimal power flow formulation and solution using quadprogj. 2010.

[157] Kaarthik Sundar and Anatoly Zlotnik. State and parameter estimation for natural gas pipeline networks using transient state data. *IEEE Transactions on Control Systems Technology*, 27(5):2110–2124, 2018.

[158] Ross Edward Swaney and Ignacio E Grossmann. An index for operational flexibility in chemical process design. Part I: Formulation and theory. *AIChE Journal*, 31(4):621–630, 1985.

[159] Wentao Tang, Andrew Allman, Davood Babaei Pourkargar, and Prodromos Daoutidis. Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection. *Computers & Chemical Engineering*, 111:43–54, 2018.

[160] Andrea Toselli and Olof Widlund. *Domain decomposition methods-algorithms and theory*, volume 34. Springer Science & Business Media, 2006.

[161] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[162] Lipo Wang. *Support vector machines: theory and applications*, volume 177. Springer Science & Business Media, 2005.

[163] Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78(1):29–63, 2019.

[164] Stephen J Wright. Solution of discrete-time optimal control problems on parallel computers. *Parallel Computing*, 16(2-3):221–237, 1990.

[165] Wanting Xu and Mihai Anitescu. Exponentially accurate temporal decomposition for long-horizon linear-quadratic dynamic optimization. *SIAM Journal on Optimization*, 28(3):2541–2573, 2018.

[166] Wanting Xu and Mihai Anitescu. Exponentially convergent receding horizon strategy for constrained optimal control. *Vietnam Journal of Mathematics*, 47(4):897–929, 2019.

[167] Zheng Xu, Soham De, Mario Figueiredo, Christoph Studer, and Tom Goldstein. An empirical study of ADMM for nonconvex problems, 2016.

[168] Mario Zanon and Timm Faulwasser. Economic MPC without terminal constraints: Gradient-correcting end penalties enforce asymptotic stability. *Journal of Process Control*, 63:1–14, 2018.

[169] Victor M Zavala. Stability analysis of an approximate scheme for moving horizon estimation. *Computers & Chemical Engineering*, 34(10):1662–1670, 2010.

[170] Victor M Zavala. New architectures for hierarchical predictive control. *IFAC-PapersOnLine*, 49(7):43–48, 2016.

[171] Victor M Zavala, Mihai Anitescu, and Theodore Krause. On the optimal on-line management of photovoltaic-hydrogen hybrid energy systems. In *Computer Aided Chemical Engineering*, volume 27, pages 1953–1958. Elsevier, 2009.

[172] Victor M Zavala, Carl D Laird, and Lorenz T Biegler. Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems. *Chemical Engineering Science*, 63(19):4834–4845, 2008.

[173] Runyu Zhang, Yingying Li, and Na Li. On the regret analysis of online LQR control with predictions. *arXiv preprint arXiv:2102.01309*, 2021.

[174] Anatoly Zlotnik, Michael Chertkov, and Scott Backhaus. Optimal control of transient flow in natural gas networks. In *2015 54th IEEE conference on decision and control (CDC)*, pages 4563–4570. IEEE, 2015.

[175] Anatoly Zlotnik, Line Roald, Scott Backhaus, Michael Chertkov, and Göran Andersson. Coordinated scheduling for interdependent electric power and natural gas infrastructures. *IEEE Transactions on Power Systems*, 32(1):600–610, 2016.

[176] T Zolezzi. On stability analysis in mathematical programming. In *Sensitivity, Stability and Parametric Analysis*, pages 227–242. Springer, 1984.

APPENDIX
Benchmark Algorithms

In this appendix, we briefly discuss two standard optimization algorithms: ADMM and IPM. Here, we do not intend to provide a comprehensive review of these methods. Rather, these will provide high-level, introductory explanations, which facilitate our discussion in the main text. The interested readers are referred to the following articles: [27, ADMM]; [124, IPM].

A.1 Alternating Direction Method of Multipliers

Consider a nonlinear program of the following form:

$$\min_{\boldsymbol{x}} \ \boldsymbol{f}(\boldsymbol{x}) \tag{A.1}$$

$$\text{s.t. } \boldsymbol{c}(\boldsymbol{x}) = 0 \tag{A.2}$$

$$\boldsymbol{x} \geq 0. \tag{A.3}$$

Typically, ADMM has been used for solving convex problems, and the convergence proof also relies on the convexity assumptions. However, for a number of instances, it has been demonstrated that ADMM converges reasonably well for nonconvex problems (nonconvex $\boldsymbol{f}(\cdot)$ and nonlinear $\boldsymbol{c}(\cdot)$) [136, 167], and the convergence is analyzed under a few particular nonconvex settings [89, 163].

First, we assume that problem (A.1) has a certain decomposable structure (e.g., as in (1.1)), and thus one can reformulate the problem as follows:

$$\min_{\{\boldsymbol{x}_k\}_{k\in\mathcal{K}},\boldsymbol{z}} \ \sum_{k\in\mathcal{K}} \boldsymbol{f}_k(\boldsymbol{x}_k) \tag{A.4a}$$

$$\text{s.t. } \boldsymbol{c}_k(\boldsymbol{x}_k) = 0 \tag{A.4b}$$

$$\boldsymbol{x}_k \geq 0 \tag{A.4c}$$

$$A_k\boldsymbol{x}_k + B_k\boldsymbol{z} = 0 \tag{A.4d}$$

This can be done as follows. First, we introduce duplicate variables for coupled variables for each k (they are contained in $\boldsymbol{x}_k$), and also introduce the global coupled variables $\boldsymbol{z}$. Then, we enforce the equality constraint (A.4d). This procedure is often called *lifting*. Now, one can observe that if we fix variable $\boldsymbol{z}$, then the problem becomes separable for each $k \in \mathcal{K}$.

We now are ready to define the algorithm. ADMM algorithm is essentially an inexact variation of method of multipliers [16, 124]. Instead of exactly solving the primal minimization problem, it performs one sweep of Gauss-Seidel iteration on the augmented Lagrangian, and then perform the dual update. The ADMM algorithm can be stated as follows: for each iteration $\ell = 0, 1, \cdots,$

$$\begin{aligned}\boldsymbol{x}_k^{(\ell+1)} = \underset{\boldsymbol{x}_k}{\text{argmin}}\ & \boldsymbol{f}_k(\boldsymbol{x}_k) + (\boldsymbol{y}_k^{(\ell)})^\top(A_k\boldsymbol{x}_k + B_k\boldsymbol{z}^{(\ell)}) + (\mu/2)\|A_k\boldsymbol{x}_k + B_k\boldsymbol{z}^{(\ell)}\|^2,\ k \in \mathcal{K} \\ & \text{s.t. } \boldsymbol{c}_k(\boldsymbol{x}_k) = 0 \\ & \quad \boldsymbol{x}_k \geq 0\end{aligned} \tag{A.5a}$$

$$\boldsymbol{z}^{(\ell+1)} = \underset{\boldsymbol{z}}{\text{argmin}} \sum_{k\in\mathcal{K}} (\boldsymbol{y}_k^{(\ell)})^\top(A_k\boldsymbol{x}_k^{(\ell+1)} + B_k\boldsymbol{z}) + (\mu/2)\|A_k\boldsymbol{x}_k^{(\ell+1)} + B_k\boldsymbol{z}\|^2 \tag{A.5b}$$

$$\boldsymbol{y}_k^{(\ell+1)} = \boldsymbol{y}_k^{(\ell)} + \mu(A_k\boldsymbol{x}_k^{(\ell+1)} + B_k\boldsymbol{z}_k^{(\ell+1)}),\ k \in \mathcal{K}. \tag{A.5c}$$

Here, the partial augmented Lagrangian is minimized for each $\boldsymbol{x}_k$ for $k \in \mathcal{K}$ and $\boldsymbol{z}$ (sequentially). Then the dual for the relaxed constraints are updated as in standard method of multipliers. This iterative scheme allows exploiting the parallel computaions. Note that the most expensive step (A.5a) is parallelizable for each $k \in \mathcal{K}$. Other parts are not computationally expensive; one can see that solving (A.5b) is just solving an unconstrained QP and (A.5c) only involves a matrix-vector multiplications and vector additions. Thus, the computation in (A.5b)-(A.5c) is much cheaper than (A.5a). We have implemented multi-threading parallelism-based ADMM in `SimpleADMM.jl` [3]

A.2 Interior Point Method

We consider a nonlinear program of the form in (A.1). The IPM finds the solution of NLPs by solving a sequence of *barrier subproblems*:

$$\min \varphi(\boldsymbol{x}) := \boldsymbol{f}(\boldsymbol{x}) - \mu\, \boldsymbol{e}^T \log(\boldsymbol{x}) \tag{A.6}$$

$$\text{s.t. } \boldsymbol{c}(\boldsymbol{x}) = 0. \tag{A.7}$$

with a decreasing sequence for parameter μ. Here, $\boldsymbol{e}$ is a vector of ones. Note that the inequality constraint in (A.1) is replaced by a smooth log-barrier function. Treating the inequalities as log-barrier function allows eliminating the combinatorial complexity of the inequality constrained NLPs.

The KKT conditions for (A.6) give rise to the nonlinear equations:

$$\nabla \boldsymbol{f}(\boldsymbol{x}) + A^\top \boldsymbol{y} - \boldsymbol{z} = 0 \tag{A.8}$$

$$\boldsymbol{c}(\boldsymbol{x}) = 0 \tag{A.9}$$

$$\boldsymbol{X}\boldsymbol{Z}\mathbf{e} - \mu\, \mathbf{e} = 0, \tag{A.10}$$

where $A := \nabla \boldsymbol{c}(\boldsymbol{x})$, $\boldsymbol{X} := \text{diag}(\boldsymbol{x})$, and $\boldsymbol{Z} := \text{diag}(\boldsymbol{z})$; note that one can obtain the standard form KKT conditions upon the elimination of $\boldsymbol{z}$. A solution of KKT system (A.8) can be obtained by computing primal-dual Newton steps $\boldsymbol{d}^\star$ from:

$$\underbrace{\begin{bmatrix} W + \Sigma + \delta_w I & A^\top \\ A & -\delta_c I \end{bmatrix}}_{\boldsymbol{M}} \underbrace{\begin{bmatrix} \boldsymbol{d}^x \\ \boldsymbol{d}^y \end{bmatrix}}_{\boldsymbol{d}} = - \underbrace{\begin{bmatrix} \nabla_{\boldsymbol{x}} \varphi(\boldsymbol{x}) + A^\top \boldsymbol{y} \\ \boldsymbol{c}(\boldsymbol{x}). \end{bmatrix}}_{\boldsymbol{p}}, \tag{A.11}$$

where $W := \nabla^2_{\boldsymbol{x}\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$, $\Sigma := \boldsymbol{X}^{-1}\boldsymbol{Z}$, and $\delta_w, \delta_c > 0$ are regularization parameters. The regularization parameter is selected typically based on the intertia information obtained from block $LDL^\top$ factorization. The step $\boldsymbol{d}^\star$ computed from (A.11) is safeguarded by a line-search filter procedure to induce global convergence [161]. The decrease in the penalty parameter is triggered by checking the residual to the KKT conditions (A.11) for the barrier problem. The algorithm is also typically safeguarded by the so-called restoration phase, which tries to

minimize the constraint violation when a convergence issue is faced. The readers are referred to [161] for more details of the implementation.

APPENDIX
Problem Formulations

B.1 Storage Control

Consider a dynamic optimization problem for energy storage:

$$\min_{\{s_i,u_i,v_i\}_{i=1}^k} \sum_{i=1}^{k} \frac{1}{2}\eta(s_i)^2 + \frac{1}{2}(u_i)^2 + \pi_i u_i \tag{B.1a}$$

$$\text{s.t. } s_1 = \overline{s} \tag{B.1b}$$

$$s_i = s_{i-1} + bu_{i-1} + w_i,\ i \in \mathbb{I}_{[2,k]}. \tag{B.1c}$$

Here, $s_i \in \mathbb{R}$ is the stored energy (state) at time i; $u_i \in \mathbb{R}$ is the charge/discharge of energy (control); $v_i \in \mathbb{R}$ are the transactions with the grid; $\overline{s} = w_1$ is the initial storage; $\pi_i \in \mathbb{R}$ is the energy price forecast; d_i is the energy demand forecast; and w_i is the disturbance forecast. We consider $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as a linear graph that represents time domain, $\mathcal{V} := \mathbb{I}_{[1,N]}$ and $\mathcal{E} := \{\{i, i+1\}\}_{i=1}^{n-1}$. Practical problems have inequality constraints for s_i, u_i, but here we neglect them for simplicity, and incorporate the regularizations in the objective function as in (B.1b).

B.2 Quadrotor Motion Planning

We consider a quadrotor motion planning problem [86]:

$$\min_{x(\cdot),u(\cdot)} \int_0^T \left(\frac{1}{2}(s(t) - \overline{s}(t))^\top Q(s(t) - \overline{s}(t)) + \frac{1}{2}u(t)^\top Ru(t)\right) dt \tag{B.2a}$$
$$+ \frac{1}{2}(x(T) - \overline{s}(T))^\top Q_f(x(T) - \overline{s}(T))$$

$$\text{s.t. } \frac{d^2X}{dt^2} = a(\cos\gamma \sin\beta \cos\alpha + \sin\gamma \sin\alpha),\ t \in [0, T] \tag{B.2b}$$

$$\frac{d^2Y}{dt^2} = a(\cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha),\ t \in [0,T] \tag{B.2c}$$

$$\frac{d^2Z}{dt^2} = a\cos\gamma\cos\beta - g,\ t \in [0,T] \tag{B.2d}$$

$$\frac{d\gamma}{dt} = (b\omega_X\cos\gamma + \omega_Y\sin\gamma)/\cos\beta,\ t \in [0,T] \tag{B.2e}$$

$$\frac{d\beta}{dt} = -b\omega_X\sin\gamma + \omega_Y\cos\gamma,\ t \in [0,T] \tag{B.2f}$$

$$\frac{d\alpha}{dt} = b\omega_X\cos\gamma\tan\beta + \omega_Y\sin\gamma\tan\beta\ + \omega_Z,\ t \in [0,T], \tag{B.2g}$$

where X, Y, Z are the coordinates in the 3-D space; α, β, γ are the yaw, pitch, and roll angles; a is the thrust; $\omega_X, \omega_Y, \omega_Z$ are the rotational rates; g is the gravitational acceleration; T is the prediction horizon length; $Q := \text{diag}(1,1,1,\eta,\eta,\eta,1,1,1)$, $R, Q_f := I$; $s(\cdot)$ is the state variables, defined as: $x := (X, \dot{X}, Y, \dot{Y}, Z, \dot{Z}, \gamma, \beta, \alpha)$; $u(\cdot)$ is the control variable, defined as $(a, \omega_X, \omega_Y, \omega_Z)$; and $\overline{s}(\cdot)$ is the set point trajectory. The problem is formulated as a tracking model predictive control (MPC) problem that seeks to minimize the deviation of the predicted trajectory from the given reference trajectory while minimally using the input, over the specified prediction horizon $[0, T]$. See Figure B.1 for the schematic illustration of the physical system. While the problem in (B.2) is formulated as a continuos time ODE-constrained optimization problem, it can be reformulated as a finite dimensional NLP by applying the discretization techniques. The discretization can be performed either by directly embedding the discretized ODE as algebraic equations or by interfacing the numerical ODE solvers with the problem [18,20]. For either case, the discretized problem reduces to a gsNLP. We use the explicit Euler scheme to formulate the problem as a gsNLP. For testing OSM, we used the following as the base parameter sets: $\widetilde{\omega} = 1$, $\overline{\mu} = 1$, $\eta = 1$, and $b = 1$. For OSM, we decomposed the problem into 20 subproblems.

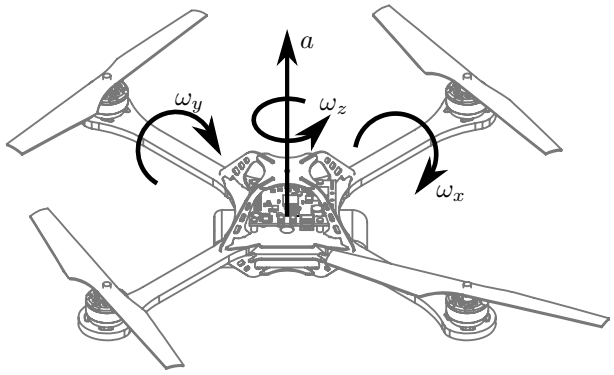

Figure B.1 Schematic of quadrotor.

B.3 Stochastic Storage Control

Consider a stochastic program for the energy storage control problem:

$$\min_{s(0),u(0)} \frac{1}{2}\eta s(0)^2 + \frac{1}{2}u(0)^2 + \pi(0)u(0) + \mathbb{E}\left[\min_{s(1),u(1)} \frac{1}{2}s(1)^2 + \frac{1}{2}u(1)^2 + \pi(1)u(1) + \mathbb{E}\left[\cdots\right]\right] \tag{B.3a}$$

$$\text{s.t. } s(0) = \overline{s} \tag{B.3b}$$

$$s(t+1) = s(t) + bu(t) + w(t),\ t \in \mathbb{I}_{[0,N]}, \tag{B.3c}$$

where t denotes the time stages, $s(t)$ is the state of charge, $u(t)$ is the charge/discharge rate, $w(t)$ and $\pi(t)$ are the time-varying uncertain system load and the electricity price, $\overline{s}$ is the initial state of the charge, and $\mathbb{E}[\cdot]$ detnotes the expected value. The battery system serves the electricity loads of the buildings while transacting with the power grid by reponding to the frequency regulation price signals. The operational decision of the battery is made by solving the optimization problem in (B.3) in real-time. The problem aims to maximize the expected profit of battery system operation by exploiting the time-varying nature of electricity price and demand. The control decision is implemented in a receding-horizon fashion. A schematic of the battery system is illustrated in Figure B.2.

The problem in (B.3) can be reformulated as a multi-stage stochastic program:

$$\min_{\{s_i,u_i\}_{i\in\mathcal{V}}} \sum_{i\in\mathcal{V}} p_i \cdot (\frac{1}{2}\eta s_i^2 + \frac{1}{2}u_i^2 + \pi_i u_i) \tag{B.4a}$$

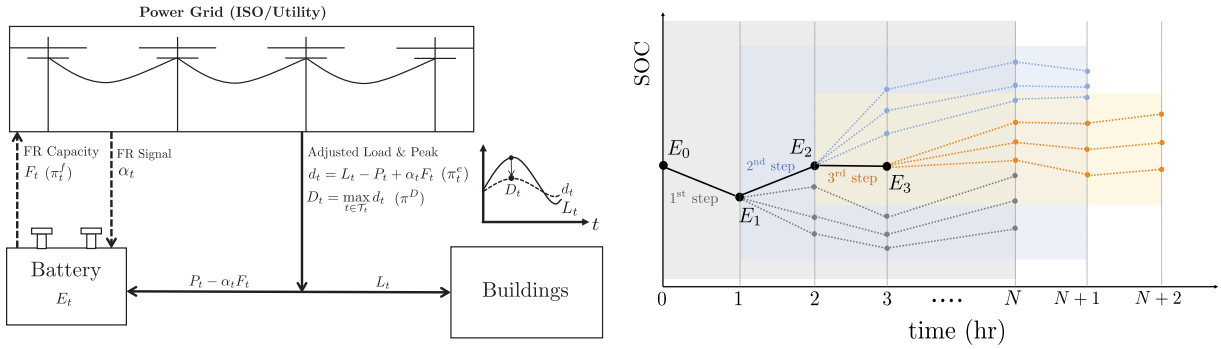

Figure B.2 Schematic of battery system under study (left) and the scenario tree (right).

$$\text{s.t. } s_1 = \overline{s}, \tag{B.4b}$$

$$s_i = s_{an(i)} + bu_{an(i)} + w_{an(i)},\ i \in \mathcal{V} \setminus \{1\}. \tag{B.4c}$$

Here $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents the scenario tree (see Figure B.2); $1 \in \mathcal{V}$ is the root node; $an(i) \in \mathcal{N}_{\mathcal{G}}[i]$ denotes the parent node; $p_i \in \mathbb{R}_{\geq 0}$ denotes the probability of node i; $s_i \in \mathbb{R}$ is the stored energy at node i; $u_i \in \mathbb{R}$ is the charge/discharge of energy at node i; $\overline{s} = w_1$ is the initial energy storage; $\pi_i \in \mathbb{R}$ is the forecasted energy price at node i; w_i is the disturbance forecast at node i. One can observe that the problem is formulated as a gsNLP, and the problem is structured by the scenario tree $\mathcal{G}$. For testing OSM, we used the following as the base parameter sets: $\widetilde{\omega} = 1.2$, $\overline{\mu} = 1$, $\eta = 1$, and $b = 1$. For OSM, we decomposed the problem into 28 subproblems.

B.4 Thin Plate Temperature Control

We consider a steady-state temperature control problem over 2D space [117]:

$$\min_{s(\cdot),u(\cdot)} \int_{w \in \Omega} \frac{1}{2}\eta(s(w) - \overline{s}(w))^2 + \frac{1}{2}u(w)^2 dw \tag{B.5a}$$

$$\text{s.t. } \Delta s(w) = \frac{2h_c}{\kappa t_z}(s(w) - \overline{T}) + \frac{2\epsilon\sigma}{\kappa t_z}(s(w)^4 - \overline{T}^4) - \frac{1}{\kappa t_z}(bu(w) + d(w)),\ w \in \Omega \tag{B.5b}$$

$$\nabla s(w) \cdot \hat{\boldsymbol{n}}(w) = 0,\ w \in \partial\Omega_{(k)}, \tag{B.5c}$$

where $\Omega = [0, L] \times [0, L] \subseteq \mathbb{R}^2$ is the 2-dimensional domain of interest; $\partial\Omega$ is the boundary of Ω; $s : \Omega \to \mathbb{R}$ is the temperature; $u : \Omega \to \mathbb{R}$ is the control; $d : \Omega \to \mathbb{R}$ is the disturbance; Δ is

the Laplacian operator; $\hat{\boldsymbol{n}}$ is the unit normal vector; $\cdot$ is the inner product; (B.5b) is the heat equation whose right-hand-side consists of convection, radiation, and forcing terms by control and disturbance; (B.5c) is the Neumann boundary condition (i.e., insulated); $\overline{s} : \Omega \to \mathbb{R}$ is the desired temperature; $\kappa = 400$, $t_z = .01$, $h_c = 1$, $\epsilon = .5$, $\sigma = 5.67 \times 10^{-8}$, and $\overline{T} = 300$ are the constant parameters. A variant problem can be formulated by replacing the Neumann boundary condition to the Dirichlet boudnary condition (the boundary temperature is fixed):

$$s(w) = T_a, \ w \in \partial\Omega.$$

For testing OSM, we used the following as the base parameter sets: $\widetilde{\omega} = 2$, $\overline{\mu} = 1$, $\eta = 1$, and $b = 0.1$. For OSM, we decomposed the problem into 25 subproblems.

B.5 DC Power System State Estimation

We consider a DC power system state estimation problem. We assume that the network is primarily inductive, the voltage amplitudes are fixed to one, and the voltage angle differences between the neighboring nodes are small enough to apply a DC approximation. The power flow P_{ij} on edge $\{i, j\} \in \mathcal{E}$ can be expressed by $P_{ij} = y_{ij}(\delta_i - \delta_j)$ (assume that a direction is assigned to each edge). We assume that the power flow is measured and the measurement is performed based on a statistical model $P_{ij}^m = P_{ij} + \xi_{P_{ij}}$, where $\xi_{P_{ij}}$ is a random variable whose distribution is $\xi_{P_{ij}} \sim N(0, \sigma_{P_{ij}}^2)$. By incorporating the prior on $\delta_i \sim N(\delta_i^m, \sigma_{\delta_i}^2)$ for $i \in \mathcal{V}$, one can derive the following maximum a posteriori problem.

$$\min_{\delta, P} \ \sum_{i \in \mathcal{V}} \left(\frac{\delta_i - \delta_i^m}{\sigma_{\delta_i}} \right)^2 + \sum_{\{i,j\} \in \mathcal{E}} \left(\frac{P_{ij} - P_{ij}^m}{\sigma_{P_{ij}}} \right)^2 \tag{B.6a}$$

$$\text{s.t. } P_{ij} = y_{ij}(\delta_i - \delta_j), \quad \{i, j\} \in \mathcal{E} \tag{B.6b}$$

In our estimation setting, we assume that only a subset of flows can be measured and the rest need to be inferred from data.

Accordingly, we assume $\sigma_{P_{ij}} = y_{ij}$ for the measured flows (about half of the edges are randomly selected) and assume much weaker prior for the rest of the edges by $\sigma_{P_{ij}} = \sqrt{10} y_{ij}$.

The prior weight $c \in \mathbb{R}_{>0}$ on the unmeasured voltage angles is assumed to be uniform, that is, $\sigma_{\delta_i} = \frac{1}{c}, \forall i \in \mathcal{V}$. The estimation problem can be written in vector form as

$$\min_{\delta,P} \ (\delta - \delta^m)^T \Sigma_\delta (\delta - \delta^m) + (P - P^m)^T \Sigma_P (P - P^m) \tag{B.7a}$$

$$\text{s.t. } P = Y\delta, \tag{B.7b}$$

where $Y \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{V}|}$ and $\Sigma_\delta, \Sigma_P \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. This problem can be reduced to an unconstrained QP:

$$\min_{\delta} \ \delta^T \left(\Sigma_\delta + Y^T \Sigma_P Y\right) \delta - 2(Y^T \Sigma_P P^m + \Sigma_\delta \delta^m)^T \delta. \tag{B.8}$$

where $H := \Sigma_\delta + Y^T \Sigma_P Y$, and $f := Y^T \Sigma_P P^m + \Sigma_\delta \delta^m$.

We used data from the Pegase project [65] to derive the power system model (available at `pglib-opf` v18.08 [11, 24]). We apply graph partitioning based on a multilevel k-way partitioning method using `METIS` [103] to identify the partition $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$.

B.6 AC Optimal Power Flow

We consider the alternating current (AC) optimal power flow problem:

$$\min_{\substack{\{v_i\in\mathbb{C}\}_{i\in\mathcal{V}} \\ \{s^g_k\in\mathbb{C}\}_{k\in\mathcal{W}} \\ \{s_{ij}\in\mathbb{C}\}_{i,j\in\mathcal{V}}}} \ \eta\Big(\sum_{i\in\mathcal{V}}(|v_i| - v_{\text{ref}})^2 + \sum_{\{i,j\}\in\mathcal{E}} (\angle v_i v_j^*)^2\Big) + \sum_{k\in\mathcal{W}} c^1_{(k)} \Re(s^g_{(k)}) + c^2_{(k)} \Re(s^g_{(k)})^2 \tag{B.9a}$$

$$\text{s.t. } \angle v_i = 0, \ i \in \mathcal{V}_{\text{ref}} \tag{B.9b}$$

$$s^{gL}_{(k)} - b(1+\sqrt{-1}) \le s^g_{(k)} \le s^{gU}_{(k)} + b(1+\sqrt{-1}), \ k \in \mathcal{W} \tag{B.9c}$$

$$\sum_{k\in\mathcal{W}_i} s^g_k - s^d_i = \sum_{j\in N_{\mathcal{G}}[i]} s_{ij}, \ v^L_i \le |v_i| \le v^U_i, \ i \in \mathcal{V} \tag{B.9d}$$

$$s_{ij} = (Y_{ij} + Y^c_{ij})^* \frac{|v_i|^2}{|T_{ij}|} v_i^* - Y_{ij} \frac{v_i v_j^*}{T_{ij}}, \ |s_{ij}| \le s^U_{ij}, \ i,j \in \mathcal{V} \tag{B.9e}$$

$$\theta^{\Delta L}_{ij} \le \angle v_i v_j^* \le \theta^{\Delta U}_{ij}, \ \{i,j\} \in \mathcal{E}. \tag{B.9f}$$

Here, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents the power network, $\mathbb{C}$ denotes the set of complex numbers; $\Re(\cdot)$ and $\Im(\cdot)$ denotes the real and imaginary part of the argument; $(\cdot)^*$ denotes the complex

conjugate of the argument; $p \geq q \iff \Re(p) \geq \Re(q)$ and $\Im(p) \geq \Im(q)$ for $p, q \in \mathbb{C}$; $\mathcal{W}_i$ is the set of generators connected to node i; $\mathcal{W} := \bigcup_{i \in \mathcal{V}} \mathcal{W}_i$; $\mathcal{V}_{\text{ref}}$ is the set of reference nodes; $v_i \in \mathbb{C}$ is the voltage at node i; $s^g_{(k)} \in \mathbb{C}$ is the power generation at generator k; $\{v_i^L, v_i^U, s_i^d \in \mathbb{C}\}_{i \in \mathcal{V}}$,$\{\theta_{ij}^{\Delta,L}, \theta_{ij}^{\Delta,U} \in \mathbb{R}\}_{\{i,j\} \in \mathcal{E}}$, $\{s_{ij}^U, Y_{ij}, Y_{ij}^c, T_{ij} \in \mathbb{C}\}_{i,j \in \mathcal{V}}$, $\{c^1_{(k)}, c^2_{(k)} \in \mathbb{R}, s^{gL}_{(k)}, s^{gU}_{(k)} \in \mathbb{C}\}_{k \in \mathcal{W}}$ are the data. The readers are pointed to the documentation of `PowerModels.jl` [38] for the details of Problem (B.9). Here we modified the problem by adding the regularization term (the first term in (B.9a)) and by introducing the additional terms in constraint (B.9c) to examine the effect of positive curvature and flexibility in the constraints; the problem reduces to the original problem when $(\eta, b) = 0$. We treat the edge variables, constraints, and the objective terms by treating them as node terms for one of the connected node (in particular, the one with lower node index), as explained in Remark 1.1; note that this manipulation only alters indexing and does not change the problem. We set z_i as all the primal/dual variable that are associated with node $i \in \mathcal{V}$ (including generator and edge variables/constraints), and we set $p_i = [\Re(s_i^d), \Im(s_i^d)]$. We use test case `pglib_opf_case500_tamu` available at `pglib-opf` v18.08 [11, 24] (the problem data $c^1_{(k)}$, $c^2_{(k)}$, V_i^L, etc are available therein). The problem is modeled using modeling library `PowerModels.jl`.

B.7 AC Power System State Estimation

We consider an AC PSSE problem [118]:

$$\min_{\substack{\{v_i \in \mathbb{C}\}_{i \in \mathcal{V}} \\ \{s_k^g \in \mathbb{C}\}_{k \in \mathcal{W}}}} \quad \sum_{i \in \mathcal{V}} \left(\eta |v_i - v_i^m|^2 + \sum_{k \in \mathcal{W}_i} |s_k^g - s_k^{g,m}|^2 \right) \tag{B.10a}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{N}_{\mathcal{G}}[i]} (Y_{ij} + Y_{ij}^c)^* \frac{|v_i|^2}{|T_{ij}|} v_i^* - Y_{ij} \frac{v_i v_j^*}{T_{ij}} = \sum_{k \in \mathcal{W}_i} b s_k^g - s_i^d \tag{B.10b}$$

The problem form is similar to the standard AC OPF problem (B.9), but there is a key difference that the objective function is formulated as the quadratic penalty of the deviation from the measurement. The measurement consists of the voltage and the active/reactive power injections. The power flow equation is enforced as equality constraints. As discussed in [118], other types of measurement (e.g., voltage angle difference, active/reactive power flow, current

magnitude) can be incorporated, and other types of constraints (maximum/minimum power generation) can be included in the problem. We use test case `pglib_opf_case30000_goc` available at `pglib-opf` v20.07 [11, 24] to formulate the problem. Randomly generated data are used for the measurement. The graph partitioning technique based on a multilevel k-way partitioning method in `METIS` [103] is used to identify the partition $\{\mathcal{V}_k\}_{k\in\mathcal{K}}$. For testing OSM, we used the following as the base parameter sets: $\widetilde{\omega} = 1$, $\overline{\mu} = 1$, $\eta = 10^6$, and $b = 1$. For OSM, we decomposed the problem into 20 subproblems.

B.8 Multi-Period AC OPF

We consider a multi-period AC power flow problem with storage [73] of the form:

$$\min_{\substack{v,s,s^g,s^s\in\mathbb{C}\\ e,sc,sd,sqc\in\mathbb{R}}} \sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{G}} c^0_{kt} + c^1_{kt}\Re(s^g_{kt}) + c^2_{kt}\Re(s^g_{kt})^2 \tag{B.11a}$$

$$\text{s.t. } v^L_i \le |v_{it}| \le v^U_i,\ i\in\mathcal{N}, t\in\mathcal{T} \tag{B.11b}$$

$$\sum_{k\in\mathcal{G}_i} s^g_{kt} - \sum_{k\in L_i} s^d_{kt} + \sum_{k\in S_i} s^s_{kt} = \sum_{j\in\mathcal{N}_\mathcal{G}[i]} s_{ijt},\ i\in\mathcal{N}, t\in\mathcal{T} \tag{B.11c}$$

$$\begin{aligned} s_{ijt} &= (Y_{ij}+Y^c_{ij})^*\frac{|v_{it}|^2}{|T_{ij}|^2} - Y^*_{ij}\frac{v_{it}v^*_{jt}}{T_{ij}}, (i,j)\in\mathcal{E},\ t\in\mathcal{T}\\ s_{ijt} &= (Y_{ij}+Y^c_{ji})^*|v_{jt}|^2 - Y^*_{ij}\frac{v^*_{it}v_{jt}}{T^*_{ij}},\ (i,j)\in\mathcal{E}^R,\ t\in\mathcal{T} \end{aligned} \tag{B.11d}$$

$$|s_{ijt}| \le s^U_{ij},\ (i,j)\in\mathcal{E}\cup\mathcal{E}^R,\ t\in\mathcal{T} \tag{B.11e}$$

$$\theta^{\Delta L}_{ij} \le \angle(v_{it}v^*_{jt}) \le \theta^{\Delta U}_{ij},\ (i,j)\in\mathcal{E},\ t\in\mathcal{T} \tag{B.11f}$$

$$s^{gL}_k \le s^g_{kt} \le s^{gU}_k,\ k\in\mathcal{G},\ t\in\mathcal{T} \tag{B.11g}$$

$$e_{kt} - e_{kt-1} = (\eta^c sc_t - sd_t/\eta^d)\Delta t, k\in\mathcal{S}, t\in\mathcal{T}\setminus\{T\} \tag{B.11h}$$

$$s^s_{kt} + (sc_{kt} - sd_{kt}) = \sqrt{-1}sqc_{kt} + s^{\text{loss}}_k,\ k\in\mathcal{S}, t\in\mathcal{T} \tag{B.11i}$$

$$|s^s_{kt}| \le s^u_k, 0\le e_{kt}\le e^u_k\ k\in\mathcal{S}, t\in\mathcal{T} \tag{B.11j}$$

$$0\le sc_{kt}\le sc^u_k, 0\le sd_{kt}\le sc^u_k,\ k\in\mathcal{S}, t\in\mathcal{T}, \tag{B.11k}$$

Here, $\mathcal{G}$ is the set of generators; $\mathcal{N}$ is the set of buses; $\mathcal{E}$ is the set of (directed) branches; $\mathcal{E}^R$ is the set of branches with inverted directions; $\mathcal{S}$ is the set of storage; $\mathcal{T}$ is the time index

set; $v \in \mathbb{C}$ is the voltage; $e \in \mathbb{R}$ is the state of charge; $s \in \mathbb{C}$ is the power flow; $s^g \in \mathbb{C}$ is the power generation; $s^s \in \mathbb{C}$ is the complex power injected by the storage; $sc \in \mathbb{R}$ is the charging rate; $sd \in \mathbb{R}$ is the discharging rate; $sqc \in \mathbb{R}$ is the reactive power slack; $c^0, c^1, c^2 \in \mathbb{R}$ are the generation costs; $s^d \in \mathbb{C}$ is the power demand; Y is the admittance; T is the branch complex transformation parameter; η is the charging efficiency; s^{loss} is the storage energy loss; Δt is the time interval. Note that (B.11) can be reformulated as an NLP with real variables by separately treating the real and imaginary part of the variables and equations (a polar formulation is used here). The power network under study is a variant of IEEE 14 bus test system; this comprises 14 buses, 5 generators, 1 storage, 1 shunt, and 20 branches. The detailed model is constructed with `PowerModels.jl` [38].

B.9 Transient Gas Network Operation

We consider a transient gas network problem [157] of the form:

$$\min_{\substack{\rho,\varphi^\mu,\varphi^-,\\ \alpha,s,d\in\mathbb{R}}} \sum_{t\in\mathcal{T}} \left(\sum_{(i,j)\in\mathcal{C}} \gamma \mathcal{P}^\mu_{ijt} + \sum_{i\in\mathcal{R}} c_{it} s_{it} - \sum_{i\in D} c_{it} d_{it} \right) \tag{B.12a}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}(i)} f^\mu_{ijt} = \sum_{j\in\mathcal{R}(i)} s_{jt} - \sum_{j\in\mathcal{D}(i)} d_{jt}, i \in \mathcal{N}, t \in \mathcal{T} \tag{B.12b}$$

$$\rho_i^{\min} \le \rho_{it} \le \rho_i^{\max},\ i \in \mathcal{N},\ t \in \mathcal{T} \tag{B.12c}$$

$$\rho^2_{it} - \rho^2_{jt} = -\frac{\lambda L}{D} \varphi^a_{ijt} |\varphi^a_{ijt}|,\ (i,j) \in \mathcal{P}, t \in \mathcal{T} \tag{B.12d}$$

$$\hat{L}(\dot{\rho}_{jt} + \dot{\rho}_{it}) = -4\varphi^-_{ijt},\ (i,j) \in \mathcal{P},\ t \in \mathcal{T} \tag{B.12e}$$

$$f^\mu_{ijt}(\rho_{it} - \rho_{jt}) \le 0,\ (i,j) \in \mathcal{C},\ t \in \mathcal{T} \tag{B.12f}$$

$$\rho_{jt} = \alpha_{ijt}\rho_{it},\ (i,j) \in \mathcal{C},\ t \in \mathcal{T} \tag{B.12g}$$

$$\mathcal{P}^\mu_{ijt} \le P^{\max}_{ij},\ (i,j) \in \mathcal{C},\ t \in \mathcal{T} \tag{B.12h}$$

$$-f^\mu_{ij} \le f^\mu_{ijt} \le f^\mu_{ij},\ (i,j) \in \mathcal{C},\ t \in \mathcal{T}, \tag{B.12i}$$

where $\dot{\rho}_{it} = \frac{\rho_{it} - \rho_{it-1}}{\Delta t}$, $\mathcal{P}^\mu_{ijt} = W_a J_{ij}$, and $f^\mu_{ijt} = J_{ij}\varphi^\mu_{ijt}$. Here, $\mathcal{N}$ is the set of junctions; $\mathcal{P}$ is the set of pipelines; $\mathcal{C}$ is the set of compressors; $\mathcal{R}$ is the set of receipts; $\mathcal{D}$ is the set of

demands; $\mathcal{R}(i)$ is the set of receipts at junction $i \in \mathcal{N}$; $\mathcal{D}(i)$ is the set of demands at junction $i \in \mathcal{N}$; $\mathcal{T}$ is the time index set; ρ is the densitiy; φ^{μ} is the average mass flux; φ^{-} is the negative mass flux; α is the compression ratio; s is the supply; d is the demand; $\dot{\rho}$ is the time derivative of density; $\mathcal{P}^{\mu}$ is the power consumption of compressor; f is the mass flow; c is the gas price; γ is the economic factor; $\lambda, \hat{L}, L, D, A, \Delta t$, and W_a are physical parameters. To implicitly enforce the periodicity, we let $\rho_{i0} = \rho_{iT}$, where T is the end time index. The gas network under study consists of 2 compressors, 6 junctions (35 junctions after discretization), 4 pipelines (32 pipelines after discretization), 1 receiving points and 5 transfer points (which work either as receipt or delivery). The model is constructed using `GasModels.jl` [14].