8-2013

Mathematical Optimization for Engineering Design Problems

Brian Dandurand
Clemson University, bdandur@clemson.edu

Mathematical Optimization for Engineering Design Problems

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Mathematical Sciences

---

by
Brian C. Dandurand
August 2013

Accepted by:
Dr. Margaret M. Wiecek, Committee Chair
Dr. Neil Calkin
Dr. Georges Fadel
Dr. Irina Viktorova
Dr. Daniel Warner

Abstract

Applications in engineering design and the material sciences motivate the development of optimization theory in a manner that additionally draws from other branches of mathematics including the functional, complex, and numerical analyses.

The first contribution, motivated by an automotive design application, extends multiobjective optimization theory under the assumption that the problem information is not available in its entirety to a single decision maker as traditionally assumed in the multiobjective optimization literature. Rather, the problem information and the design control are distributed among different decision makers. This requirement appears in the design of an automotive system whose subsystem components themselves correspond to highly involved design subproblems each of whose performance is measured by multiple criteria. This leads to a system/subsystem interaction requiring a coordination whose algorithmic foundation is developed and rigorously examined mathematically.

The second contribution develops and analyzes a parameter estimation approach motivated from a time domain modeling problem in the material sciences. In addition to drawing from the theory of least-squares optimization and numerical analysis, the development of a mathematical foundation for comparing a baseline parameter estimation approach with an alternative parameter estimation approach relies on theory from both the functional and complex analyses.

The application of the developed theory and algorithms associated with both contributions is also discussed.

Dedication

To my wife, Xiaochun (Wendy) Hu, and my son, Ian Dandurand, for their steady support, sacrifice, and patience.

Acknowledgments

I would like to thank Dr. Margaret M. Wiecek, my Ph.D. dissertation advisor, for the abundance of time which she spent in teaching me how to present, how to write papers, and for her meticulous reading of everything that I wrote with her. Additionally, I would also like to thank Dr. Irina Viktorova, for her collaboration, support, and guidance; Dr. Matthew Saltzman, my master's degree advisor, for inspiring my appreciation of Linux, C++, object-oriented programming, and computational mathematics; Dr. Warren Adams, Dr. Neil Calkin, Dr. Daniel Warner, and Dr. Taufiquar Khan for their time, attention, and many helpful suggestions; the Automotive Research Center (ARC), whose grant provided both financial support and also an interesting problem motivating new mathematical questions; Dr. Georges Fadel, Dr. Paolo Guarneri, and the other ARC quad members for their helpful direction and guidance.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The mutual benefit resulting from the interaction between mathematical analysis and engineering application may be described in the following manner. While addressing the needs of engineering design in detail, hidden nuances within mathematical theory may become evident, thus motivating new research. On the other hand, the detailed exploration of mathematical concepts in a holistic manner, even to a degree that is not immediately relevant to the engineering application at hand, frequently provides an unexpected benefit to engineering design. Both of these observations hold true for the two main contributions that are presented in this dissertation. The first contribution addresses multiobjective optimization under the assumption that the problem information is not available in its entirety to a single decision maker, but rather, is distributed among different decision makers. The research presented in this part of the dissertation is motivated by the engineering application of designing an automobile system whose subsystem components themselves correspond to highly involved design subproblems. This leads to system/subsystem interaction whose coordination is to be mathematically analyzed.

The second contribution develops and analyzes a parameter estimation approach motivated by a modeling problem in the material sciences. In addition to drawing from the theory of least-squares optimization and numerical analysis, the analysis of the alternate parameter estimation approach relies on theory from both the functional and complex analyses.

1.1 State of the art

Much has been laid in the way of foundations for the two main contributions that are the subject of this thesis. For each main contribution, different areas of mathematics and engineering that may not seem immediately related to one another are integrated in such a way as to shed light on necessary refinements. This section discusses each foundational area in terms of mathematics or engineering and what additionally is needed for the contributions of this thesis.

1.1.1 Multiobjective, Multidisciplinary Design Optimization

The design of engineering systems often makes use of an optimality concept based on multiple measures of performance corresponding to multiple objective functions. While the optimization of a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, over a set $X \subset \mathbb{R}^n$ is commonly understood in terms of minimization or maximization, the optimization of vector-valued functions $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$ over a set X requires the introduction of a different concept of optimality. Write the vector-valued objective function as $\mathbf{f} = [f_1, \ldots, f_p]$, and assume that each objective function f_i, $i = 1, \ldots, p$ is to be minimized. The concept of minimization as immediately carried over from the single objective case is not well-defined due to the issue of conflict that typically arises in the presence of more than one objective. Conflict occurs where a minimizer for one objective function f_i is not necessarily a minimizer for another objective function f_j, where $i \neq j$. Optimality for a multiobjective optimization problem (MOP) of the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in X \end{aligned} \tag{1.1}$$

for feasible set $X \subset \mathbb{R}^n$ is typically understood in terms of Pareto optimality according to which a partial ordering $\leq$ is assigned to the objective space $\mathbb{R}^p$. Given two points $\mathbf{y}^1, \mathbf{y}^2 \in \mathbb{R}^p$ in the objective space, define $\mathbf{y}^1 \leq \mathbf{y}^2$ whenever $y_i^1 \leq y_i^2$ for each $i = 1, \ldots, p$ and $\mathbf{y}^1 \neq \mathbf{y}^2$. When $\mathbf{y}^1 \leq \mathbf{y}^2$, it is said that $\mathbf{y}^1$ *dominates* $\mathbf{y}^2$ or that $\mathbf{y}^2$ *is dominated* by $\mathbf{y}^1$. Using this preference ordering $\leq$ defined in the objective space $\mathbb{R}^p$, a point $\mathbf{x}^* \in X$ is *efficient* for problem (1.1) if there does not exist a solution $\mathbf{x} \in X$ for which $\mathbf{f}(\mathbf{x}) \leq \mathbf{f}(\mathbf{x}^*)$. The value $\mathbf{y} = \mathbf{f}(\mathbf{x}^*)$ corresponding to an efficient solution $\mathbf{x}^*$ is said to be *nondominated*. The set of solutions $\mathbf{x}^*$ efficient for problem (1.1) is denoted by $E(\mathbf{f}, X)$, and the corresponding image set, denoted by $Y_N = \mathbf{f}(E(\mathbf{f}, X))$ is referred to as the *Pareto*

set.

The definition of a preference ordering $\leq$ on the objective space is not unique, and may be given more generally in terms of a closed cone C, where $\mathbf{y}^1 \leq_C \mathbf{y}^2$ if $(\mathbf{y}^2 - \mathbf{y}^1) \in C$ and $\mathbf{y}^1 \neq \mathbf{y}^2$. For Pareto optimality, $C = \mathbb{R}^p_{\geqq} := \{\mathbf{y} \in \mathbb{R}^p : y_i \geq 0 \text{ for } i = 1, \ldots, p\}$.

While the optimal design for some engineering systems may be modeled with an MOP presented in the form given by (1.1), the optimal design of certain complex engineering systems must be modeled in terms of nonintegrable subproblems having the form

$$\begin{aligned} \min_{\mathbf{x}_i} \quad & \mathbf{f}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) = \mathbf{0} \qquad i = 1, \ldots, m \\ & \mathbf{x}_i \in X_i, \end{aligned} \tag{1.2}$$

where $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ is a partitioning of the solution $\mathbf{x}$ into block coordinates $\mathbf{x}_i \in \mathbb{R}^{n_i}$, $\sum_{i=1}^m n_i = n$; $\mathbf{x}_{\neg i} = [\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_m]$ indicates the vector of block coordinates excluding $\mathbf{x}_i$; and the umlaut notation $\ddot{\mathbf{x}}$ is used to emphasize the role of $\mathbf{x}$ as a fixed variable.

A presentation of an engineering design problem as given by subproblems (1.2) arises naturally where the design of an engineering system is multiobjective and multidisciplinary in the sense that each discipline within a system corresponds to a subsystem whose performance is evaluated with respect to multiple criteria. Distinct disciplines within a system originate from various science and engineering areas, such as fluid dynamics, thermodynamics, structures, etc., that interact with each other within the design process. These subsystems must be designed in a coordinated way due to the likely systemwide influences present in each subsystem; hence the presence of a coordinating constraints $\mathbf{h}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) = \mathbf{0}$.

Underlying the nonintegrable subproblems (1.2) is an implicitly available all-in-one (AiO) problem of the form (1.1) whose efficient points are to be computed. However, the explicit presentation of problem (1.1) is assumed to be unknown to the multiple decision makers, and so the efficient points for (1.1) cannot be computed using existing methods from multiobjective optimization theory. The efficient points for problem (1.1) must somehow be obtained from subproblem-based computations on subproblems (1.2).

Approaching the solution of the AiO problem (1.1) through references to subproblems (1.2) addresses the reality where the design of distinct subsystems is typically assigned to independent

engineering teams with complementary background and expertise. Each team has a limited understanding of the other disciplines, and the subproblems possess disparate domains and require different solution algorithms. Thus, there is no single designer with knowledge or control of the system-wide design problem data in its entirety. The incomplete flow of information and the fragmentation of design control across the disciplines is characteristic of multidisciplinary design optimization (MDO), where an engineering design problem is never solved in its entirety, but is solved through a coordination of subproblem solutions.

Within a multiobjective optimization setting, the following engineering papers address MDO: Makinen et al. 1999 [79]; Peri and Campana 2003 [91]; Jilla and Miller 2004 [66]; Dondelinger et al. 2006 [30]; Cristello and Kim 2007 [22]; Lee et al. 2007 [74]; and Dellino et al. 2007 [28]. Specific methodologies such as Multiobjective Collaborative Optimization are addressed in Tappeta and Renaud 1997 [107] and in Rabeau et al. 2007 [95]. Multiobjective Concurrent Subspace Optimization (CSSO) is addressed in Huang 2003 [61], Gunawan et al. 2003a [50], Huang and Bloebaum 2004a, 2004b, 2004c [63, 64, 62] and in Huang et al. 2007 [60]. Another direction of research involves the development of genetic algorithms for multiobjective MDO. Gunawan et al. (2003a, 2003b, 2004) [51, 50, 52] develop genetic algorithms for MDO optimization problems with global and local variables and demonstrate their applicability to engineering design problems. Parashar and Bloebaum (2006) [88] propose a genetic algorithm for multiobjective CSSO.

Liu, Hoyle, and Chen [77] outline a technique for applying decomposition to a multidisciplinary problem where each discipline's problem is multiobjective. The approach is framed with ATC and Compromise Programming (CP) for generating Pareto sets. The algorithm starts out by generating Pareto sets for each subproblem. These Pareto sets are interpolated by a function that is used to form constraints in the system level. These constraints aid in the system level computation of reasonable subproblem targets. The targets are meant to convey coordinating information to the subproblems. Efficient points computed at the subsystem level are based on scalarizing and target information determined at the system level.

Honda, Ciucci, and Yang [59, 20] present an information-passing strategy for achieving Pareto optimality in the design of complex systems that applies a game-theoretic approach to coordinating the designs of complex systems. Information is passed in the form of constraints enforcing the requirement that no other systems' objective is worsened as a result of improving the current objective. These constraint are formulated from approximations of subsystem objective functions.

Efficient designs are computed in a nearly non-cooperative context, where information that is shared is minimal.

The above methods are intended to address the issues resulting from the computation of efficient points in a multiobjective, multidisciplinary setting. However, there is a need to gain a better understanding mathematically, either with respect to the convergence of proposed algorithms, or with respect to describing the coupling relationships between subproblem efficient sets and the AiO efficient set.

The mathematical relationships between system-wide efficient sets and subsystem efficient sets under various decomposition schemes have already been examined under a number of assumptions regarding the coupling of subsystems. Tarvainen and Haimes [108] derive necessary conditions for AiO efficiency in terms of marginal trade-off information between objective values—both within each subproblem and between the subproblems—based on the existence of some unknown utility function. Multiobjective decomposition is stated in terms of the above necessary conditions applied to existing single objective decompositions. Li and Haimes [76] develop a multiobjective duality theory, based on an envelope analysis, that is used to propose a coordination algorithm for the decomposition of a convex hierarchical MOP. Kopsidas [70] develops a two level approach for Lagrange decomposable multiobjective optimization problem under an optimality concept of equilibrium points that is more general than Pareto optimality. At one level, the subproblems (subcommittees) decide on an optimal solution on its subproblems. At another level, there is coordination. Engau and Wiecek [37] examine the relationship between the AiO Pareto set with the observed Pareto sets for the pairwise biobjective subproblems over a unique feasible domain, but not for disparate solution domains. Gardenghi and Wiecek [45] theoretically study the relationships between necessary conditions for quasiseparable AiO MOP problem and the coordinated, separated problem, but do not provide any computational methods. (Problem (1.1) is *quasiseparable* when it can be written in the special form

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & [\mathbf{f}_1(\mathbf{x}_1),\ldots,\mathbf{f}_m(\mathbf{x}_m)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1,\ldots,\mathbf{x}_m) = \mathbf{0} \\ & \mathbf{x}_i \in X_i \quad \text{for } i = 1,\ldots,m, \end{aligned} \tag{1.3}$$

where $\mathbf{f}_i : \mathbb{R}^{n_i} \to \mathbb{R}^{p_i}$ denotes the block i vector of objective functions, and the set $X_i \subseteq \mathbb{R}^{n_i}$ is the

local feasible set for block i. Furthermore, $\sum_{i=1}^{m} p_i = p$ and $\sum_{i=1}^{m} n_i = n$.

The mathematical theory described above characterizes various types of relationships between AiO efficient sets and the efficient sets of the coordinated quasiseparable problem in terms of equivalence of necessary conditions and containment relationships, but the issues arising in multiobjective MDO related to the disparate solution domains appearing in the subproblem formulations (1.2) are not addressed.

In view of this review, it appears that the algorithmic developments on the engineering side can benefit from an improvement in the mathematical foundation for multiobjective MDO. This foundation is needed to provide algorithms with an accompanying convergence analysis including the development of conditions under which the limit points of iterate sequences computed with the algorithms are either locally or globally efficient for the AiO problem (1.1), and possibly a rate-of-convergence result. In examining the existing work describing the mathematical relationships between AiO problem efficient sets and coordinated decomposed problem efficient sets, it becomes clear that the issues of fragmented availability of problem data and fragmented control over decision variables that are specific to multiobjective MDO have yet to be addressed theoretically. On the computational side, the existing approaches to generating solutions for multiobjective problems assume a fully integrated problem of the form (1.1) and these need to be adapted for the requirements of MDO.

1.1.2 Solution generating approaches for multiobjective optimization

Approaches to generating solution sets for fully integrated MOPs of the form (1.1) are well-studied and generally fall into two categories: scalarization methods and nonscalarizing methods [34, 36]. The scalarizing methods convert an MOP into a single objective problem (SOP) by replacing the vector-valued objective $\mathbf{f}$ with a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, and, possibly, by adding additional constraints. Under certain assumptions, the optimal solutions of these new problems yield efficient solutions of the original MOP. Each SOP instance typically yields one solution of the MOP, and so multiple SOPs are formulated for a single MOP, and the optimal solutions of the SOPs correspond to a subset of the efficient solutions of the MOP.

SOP reformulations of MOPs such as the weighted-sum method [46], weighted-t^{th} power method [119, 75], and weighted-quadratic method [112] introduce scalarizing functions f and do not introduce additional constraints. The weighted-quadratic method as developed in [112] is motivated

by the dual of a weighted-Chebyshev SOP reformulation of (1.1) given by

$$\min_{x \in X} \max_{1 \leq i \leq m} \{\lambda_i (z_i^* - f_i(x))\},$$

and rewritten as

$$\begin{array}{ll} \min_{\alpha, x} & \alpha \\ \text{s.t.} & \lambda_i f_i(x) - \alpha + \lambda_i z_i^* \leq 0, i = 1, \ldots, m, \\ & x \in X. \end{array}$$

Because this problem may not be computationally practical, the dual problem is considered. The dual problem is shown, under certain conditions, to have the form

$$\max_{x \in X} f(x)^T Q f(x) + \mathbf{q}^T f(x)$$

where symmetric square matrix Q and vector $\mathbf{q}$ are constructed in the proof. That is, both Q and $\mathbf{q}$ depend on optimal dual multipliers from the weighted-Chebyshev problem.

SOP reformulations introducing additional constraints to the original problem include the ε-constraint method [17], the Benson method [8, 18], and the weighted-Chebyshev method [121, 124, 13], and others.

Another notable scalarization method that introduces additional constraints is associated with a multiobjective concept of optimality known as *equitable* optimality. The equitable concept of optimality is a refinement of Pareto optimality that may be considered when the objectives are comparable (e.g., same underlying phenomena being measured, same units), and distribution of objective values is significant in addition to the scalar assessment of the objective values. For example, it may be desirable to not only to maximize some measure of benefit for multiple clients, but to make sure that this benefit is as evenly distributed as possible. Under the equitable concept of optimality, the objective space points $\mathbf{y}^1$ and $\mathbf{y}^2$ that are noncomparable (i.e., $\mathbf{y}^1 \not\leq \mathbf{y}^2$ and $\mathbf{y}^2 \not\leq \mathbf{y}^1$) with respect to Pareto optimality may become comparable with respect to equitable optimality. These concepts are developed and analyzed in Kostreva et al. [71, 72] and in the Ph.D. dissertation of Singh [100]. The latter two works state MOP reformulations whose efficient points are equitable efficient for the original MOP. Important auxiliary concepts are also developed in Ogryczak and Tamir [117] and Pan et al. [87]. Baatar and Wiecek [3] and Mut and Wiecek [82] provide additional

analysis under the assumption of the general use of convex preference cones.

The equitable concept of optimality has been previously applied for financial allocation problems and decision problems where an evenly distributed allocation of resources is desirable. However, the application of the equitable concept of optimality is largely unexplored in engineering design, which may provide various application contexts. In particular, the evaluation of certain engineering subsystems, such as the lithium-ion battery, are well-suited for the equitable optimality concept. Even when there is no special reason to prefer equitable efficient designs to, more generally, efficient designs, engineering design can benefit from the introduction of equitable optimality as a means to reduce the efficient set to a subset.

Nonscalarizing methods for computing efficient solutions include approaches using optimality concepts other than Pareto, (such as the lexicographic methods and max-ordering methods [33]), descent methods transferred from nonlinear programming [39, 31, 38], and set-oriented methods [41, 43, 42, 35, 99]. Lexicographic methods apply single objective minimization in a recursive manner along some ordering of objective functions reflecting their importance. In the steepest descent methods, a search direction is derived from the gradient and/or Hessian information of the objective functions of the MOP. These methods include variants for the constrained and unconstrained case. Set-oriented methods, in contrast to all previously presented approaches, find a solution set of the MOP without using scalarizing functions or other optimality concepts.

The vast literature on the computation of efficient points does not provide a method for computing these points in a distributed fashion, that is, when the MOP is available as a collection of nonintegrable subproblems as given in (1.2). In particular, the role of scalarized SOP reformulations of MOPs is well-examined in the setting where the MOP is presented as a fully integrated problem of the form (1.1). This is not so where the MOP is given in the non-integrated fashion (1.2) arising from the decomposition along multidisciplinary boundaries. In the multidisciplinary setting, the interaction between the scalarization and the decomposition and coordination algorithm used to compute efficient points needs to be examined carefully. Due to the lack of understanding regarding this interaction, the use of simple scalarization methods such as the weighted-sum is usually employed in a multidisciplinary setting. Although this is adequate for convex multiobjective MDO problems, for nonconvex problems, this leaves much to be desired.

In this dissertation, the computation of efficient points for multiobjective nonconvex MDO is of special interest. Once a suitable SOP reformulation is established that respects the requirements of

MDO, then the adaptation of the tools of Gauss-Seidel decomposition and Lagrangian coordination—as developed in the single objective setting—is addressed for use in MDO.

1.1.3 Alternating Direction Method of Multipliers (ADMM), Block Coordinate Descent (BCD) method, and Method of Multipliers

Gauss-Seidel (GS) decomposition is originally developed as a means to solve large-scale linear systems $A\,\mathbf{x} = \mathbf{b}$ in a decomposed manner where the solution space is partitioned $\mathbb{R}^n = \prod_{i=1}^m \mathbb{R}^{n_i}$ and the corresponding solutions $\mathbf{x} \in \mathbb{R}^n$ are partitioned into block coordinates $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ with $\mathbf{x}_i \in \mathbb{R}^{n_i}$ for each $i = 1, \ldots m$. With Gauss-Seidel methods, a sequence $\mathbf{x}^k$ of solution approximations to a system of equations is computed by solving the system for one block $\mathbf{x}_i$ at a time, using the optimal solution $\mathbf{x}_i^*$ as the update for that block, and fixing the values of the other blocks with their most recent updates. This idea is extended for solving nonlinear systems of equalities/inequalities whose solutions can correspond to 1st order necessary conditions for optimality. In the setting of nonlinear optimization where the problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in X \end{aligned} \tag{1.4}$$

is to be solved and the solution space is partitioned as above, the Gauss-Seidel method may be stated as the generation of a sequence $\left\{\mathbf{x}^k\right\}$ of approximations to optimal solutions for problem (1.4) with the elements $\mathbf{x}^k$ computed by

$$\begin{aligned} &\mathbf{x}_i^k = \text{argmin}_{\mathbf{x}_i} f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}^{k-1}) \text{ s.t. } (\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) \in X, \\ &\mathbf{x}_{\neg i}^k = \mathbf{x}_{\neg i}^{k-1}. \end{aligned} \tag{1.5}$$

The convergence analysis for the GS method is based on certain assumptions of the problem or system, such as the decomposability of feasible set $X = \prod_{i=1}^m X_i$ where the sets $X_i \subseteq \mathbb{R}^{n_i}$ are closed and convex for $i = 1, \ldots, m$.

A related decomposition approach is the Jacobi method, likewise developed in the context of solving large linear systems and extended for solving nonlinear systems and nonlinear optimization problems. Like the GS method, the Jacobi method generates a sequence $\left\{\mathbf{x}^k\right\}$ of solution approximations by computing single block $\mathbf{x}_i$ updates. The difference, as stated in the context of nonlinear

optimization, is that while the GS method computes a new sequence element $\mathbf{x}^k$ after a single block $\mathbf{x}_i$ update, the Jacobi method computes each sequence element $\mathbf{x}^k$ only after a full cycle $i = 1, \ldots, m$ of block $\mathbf{x}_i$ computations, as stated below.

$$\begin{aligned} &\widetilde{\mathbf{x}}_i^k = \operatorname{argmin}_{\mathbf{x}_i} f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}^{k-1}) \quad \text{s.t. } (\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}^{k-1}) \in X, \quad \text{for } i = 1, \ldots, m \\ &\mathbf{x}^k = \left(\widetilde{\mathbf{x}}_1^k, \ldots, \widetilde{\mathbf{x}}_m^k\right). \end{aligned} \tag{1.6}$$

Variations exist where $\mathbf{x}^k = u(\widetilde{\mathbf{x}}_1^k, \ldots, \widetilde{\mathbf{x}}_m^k, \mathbf{x}_1^{k-1}, \ldots, \mathbf{x}_m^{k-1})$ for some function

$$u : \prod_{i=1}^{m} \mathbb{R}^{n_i} \times \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \prod_{i=1}^{m} \mathbb{R}^{n_i}.$$

One such example of u is given by

$$u(\widetilde{\mathbf{x}}_1^k, \ldots, \widetilde{\mathbf{x}}_m^k, \mathbf{x}_1^{k-1}, \ldots, \mathbf{x}_m^{k-1}) := (\mathbf{x}_1^{k-1}, \ldots, \mathbf{x}_{i-1}^{k-1}, \widetilde{\mathbf{x}}_i^k, \mathbf{x}_{i+1}^{k-1}, \ldots, \mathbf{x}_m^{k-1})$$

where the block coordinate $\widetilde{\mathbf{x}}_i^k$ on the right-hand side corresponds to the block coordinate $\mathbf{x}_i$ update that affects the most improvement for the objective value. The nonlinear Jacobi method is applied to the distributed computation of solutions for convex, quasiseparable multi-commodity network flow problems in the 1995 dissertation of Zakarian [123]. A detailed overview of both the GS method and the Jacobi method may be found in the textbook of Bertsekas and Tsitsiklis [9].

GS and Jacobi methods are sometimes combined with Lagrangian coordination methods for solving optimization problems of the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X \end{aligned} \tag{1.7}$$

where the constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ has been relaxed and added as a Lagrange term and a penalty term in the objective function. Such augmented Lagrangian problems take the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}) + \frac{\mu}{2} \left\|\mathbf{h}(\mathbf{x})\right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x} \in X \end{aligned} \tag{1.8}$$

Lassiter et al. (2005) [73] provides a mathematically rigorous examination of the application of the subgradient method [4] for solving single objective, quasiseparable optimization problems of the form (1.3) within the framework of ATC decomposition. Tosserams et al. (2006) [113] and Li et al. (2008) [120] introduce the application of the GS methods such as the alternating direction method of multipliers (ADMM) and the Block Coordinate Descent (BCD) method as tools for applying decomposition in a single objective setting. In the case where problem functions are continuously differentiable, the method of multipliers is integrated with BCD for the purpose of Lagrangian coordination.

The Gauss-Seidel and Lagrangian coordination algorithm given by ADMM as stated in the classical setting of two block decomposition originates in the mid-1970s from Glowinski and Marrocco (1975) [47] and Gabay and Mercier (1976) [40]. ADMM generates a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ of approximations to an optimizer $\mathbf{x}^*$ and a multiplier $\mathbf{v}^*$ associated with constraints $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ for optimization problems of the form

$$\begin{aligned} \min_{\mathbf{x}_1, \mathbf{x}_2} \quad & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0} \\ & \mathbf{x}_1 \in X_1, \quad \mathbf{x}_2 \in X_2 \end{aligned} \tag{1.9}$$

where the functions $f_1 : \mathbb{R}^{n_1} \to \mathbb{R}$ and $f_2 : \mathbb{R}^{n_2} \to \mathbb{R}$ are continuous and convex (but not necessarily differentiable) over the convex sets $X_1 \subseteq \mathbb{R}^{n_1}$ and $X_2 \subseteq \mathbb{R}^{n_2}$, respectively; and the constraint function $\mathbf{h} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^q$ is linear. When the constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ is relaxed and added to the objective as a Lagrange term and a penalty term, the following augmented Lagrangian problem is given by

$$\begin{aligned} \min_{\mathbf{x}_1, \mathbf{x}_2} \quad & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) + \frac{\mu}{2} \left\|\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2)\right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_1 \in X_1, \ \mathbf{x}_2 \in X_2, \end{aligned} \tag{1.10}$$

where $\mathbf{v}$ is a multiplier vector associated with the constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$, and $\mu > 0$ is a fixed penalty constant. The ADMM sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ is generated through the following sequence of updates

starting with $\mathbf{x}^0$, $\mathbf{v}^0$.

$$\mathbf{x}_1^k \leftarrow \arg\min_{\mathbf{x}_1 \in X_1} f_1(\mathbf{x}_1) + \mathbf{v}^{k-1} \cdot \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2^{k-1}) + \frac{\mu}{2} \left\| \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2^{k-1}) \right\|_2^2 \tag{1.11}$$

$$\mathbf{x}_2^k \leftarrow \arg\min_{\mathbf{x}_2 \in X_2} f_2(\mathbf{x}_2) + \mathbf{v}^{k-1} \cdot \mathbf{h}(\mathbf{x}_1^k, \mathbf{x}_2) + \frac{\mu}{2} \left\| \mathbf{h}(\mathbf{x}_1^k, \mathbf{x}_2) \right\|_2^2 \tag{1.12}$$

$$\mathbf{v}^k \leftarrow \mathbf{v}^{k-1} + \mu\, \mathbf{h}(\mathbf{x}_1^k, \mathbf{x}_2^k) \tag{1.13}$$

The convergence analysis of ADMM is frequently examined under the more general setting of proximal point algorithms and splitting methods. For example, Eckstein and Bertsekas [32] discuss a convergence analysis for ADMM under the assumption of summably inexact updates as a special case of a generalized Douglas-Rachford splitting method. Kontogiorgis et al. [69] provide a convergence analysis that does not assume unique minimization in computing the block updates of $\mathbf{x}_i$. A recent overview of ADMM and its history for the two-block case may be found in Boyd et al. [14].

In recent years, research interest in ADMM for the general case of $m > 2$ block decomposition is evident. No known convergence analysis is available under the traditional convexity assumptions on the objective function, although empirical observation in applications [105, 90] suggests that such convergence may yet carry over for $m > 2$. Han and Yuan [53] establish convergence under the assumptions of strong convexity on each term f_i over set X_i of the objective function. A convergence analysis is also established for predictor-corrector variants of ADMM in He et al. [54, 55].

A block nonlinear Gauss-Seidel iterative approach known as the Block Coordinate Descent (BCD) method is applied for the distributed computation of critical points of optimization problems having the form

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & f(\mathbf{x}_1, \ldots, \mathbf{x}_m) \\ \text{s.t.} \quad & \mathbf{x}_1 \in X_1, \ldots, \mathbf{x}_m \in X_m, \end{aligned} \tag{1.14}$$

where sets X_i, $i = 1, \ldots, m$, are closed and convex and the function $f : \prod_{i=1}^m X_i \to \mathbb{R}$ is continuously differentiable and not required to be of the separable form assumed for ADMM. BCD generates a sequence $\{\mathbf{x}^k\}$ of approximations to a local minimizer $\mathbf{x}^*$, where each approximation update $\mathbf{x}^k$ results from the update of a single block coordinate $\mathbf{x}_i$ obtained by computing the block-wise optimal solution $\mathbf{x}_i^* \in X_i$ of $f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i})$; each block coordinate $\mathbf{x}_j$, $j \neq i$, is fixed at its most recently updated

value denoted by $\ddot{\mathbf{x}}_j$. The sequence computed with the BCD method does not always possess limit points, and limit points do not necessarily satisfy necessary conditions for optimality. Powell [94] provides example problems where the sequence generated by BCD cycles and never approaches any points satisfying necessary conditions for optimality. For each problem, the cause of this is either nondifferentiability or nonconvexity of the function f.

Some early works related to BCD include the 1957 work of Hildreth [57] and the 1963 study of Warga [118]. Convergence analysis for BCD under the assumption that function f is convex and has bounded level sets is provided in a number of nonlinear optimization textbooks, such as [11] and [9]. Convergence analysis under generalized convexity conditions on function f is provided by Grippo and Sciandrone [49] and Tseng [114]. The latter also analyzes convergence under certain relaxations on the differentiability assumption of the function f.

Unlike ADMM, the BCD method has no built-in mechanism for addressing the issue of Lagrangian coordination. The Lagrangian coordination occurs independently of BCD and so is discussed separately. For the general setting where problem data such as the objective function f and the constraint function $\mathbf{h}$ are continuous but not necessarily differentiable, the subgradient method [4, 11] may be applied. When additional assumptions such as differentiability of the objective function f and the constraint function $\mathbf{h}$ are added, then the method of multipliers may be applied.

The method of multipliers has its origins in Hestenes [56] and Powell [93]. Under the assumption that the objective function f and the constraint function $\mathbf{h}$ in problem (1.7) are continuously differentiable, the method of multipliers generates a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ with a starting point $\mathbf{v}^0$ and a starting penalty weight $\mu^0 > 0$ where

1. the solution $\mathbf{x}^k$ is a (locally) optimal solution for the augmented Lagrangian problem (1.8) with $\mathbf{v} = \mathbf{v}^k$ and $\mu = \mu^k$,

2. the multiplier vector $\mathbf{v}^{k+1} = \mathbf{v}^k + \mu^k \mathbf{h}(\mathbf{x}^k)$,

3. the penalty coefficients satisfy $0 < \mu^k \leq \mu^{k+1}$ and $\lim_{k\to\infty} \mu^k = \infty$.

The properties of the limit points $(\mathbf{x}^*, \mathbf{v}^*)$, when they exist, are given under various differentiability conditions in the textbooks of Bertsekas [10] and [11]. For example, when the feasible set $X = \mathbb{R}^n$ and the functions f and $\mathbf{h}$ are twice continuously differentiable, then conditions are given for local linear convergence to $(\mathbf{x}^*, \mathbf{v}^*)$, which in this case correspond to a local optimum point $\mathbf{x}^*$ and a

corresponding Lagrange multiplier $\mathbf{v}^*$ associated with the constraints $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ for problem (1.7). The use of inexact updates $\mathbf{x}^k$ is also addressed in the setting where feasible set $X = \mathbb{R}^n$.

For the purposes of applying the method of multipliers to MDO, an additional analysis of the method of multipliers under the setting of the inexact computation of the updates $\mathbf{x}^k$ and the presence of the feasible set $X \subset \mathbb{R}^n$ is required. A foundation for this development may be found in Proposition 4.2.1 of [11]. This proposition claims that any limit point $\mathbf{x}^*$ of a sequence $\{\mathbf{x}^k\}$ of globally optimal solutions for problem (1.8) with $\mathbf{v} = \mathbf{v}^k$, $\mu = \mu^k$ is globally optimal for problem (1.7), where $\mathbf{v}^k$ is any arbitrary bounded sequence and $\{\mu^k\}$ is a nondecreasing sequence of positive penalty coefficients growing without bound. Note that the only other assumption on problem (1.7) in Proposition 4.2.1 [11] is that of continuity of f and $\mathbf{h}$. This result suggests an algorithmic approach that is a generalized form of the penalty method that requires $\mu^k \to \infty$. The ill-conditioning that arises from the need to let $\mu^k \to \infty$ may be circumvented with a judicious construction of the sequence $\{\mathbf{v}^k\}$, and so the next addition to the foundation of Proposition 4.2.1 [11] found in Proposition 1 of [120] addresses this.

Proposition 1 of [120] uses the same assumptions as in Proposition 4.2.1 [11] and the additional assumptions on problem (1.7) that X is closed and convex, and f and $\mathbf{h}$ are continuously differentiable. Under these additional assumptions, a sequence $\{\widetilde{\mathbf{v}}^k\}$ is given where $\widetilde{\mathbf{v}}^k = \mathbf{v}^k + \mu^k \mathbf{h}(\mathbf{x}^k)$, and whose limit points $\mathbf{v}^*$, together with $\mathbf{x}^*$ satisfy the necessary conditions of optimality for problem (1.7) given by

$$\begin{gathered}\left[\nabla f(\mathbf{x}^*) + (\mathbf{v}^*)^T \nabla \mathbf{h}(\mathbf{x}^*)\right](\mathbf{x} - \mathbf{x}^*) \geq 0 \quad \text{for all } \mathbf{x} \in X \\ \mathbf{h}(\mathbf{x}^*) = \mathbf{0}.\end{gathered} \tag{1.15}$$

Algorithmically, if it becomes clear that the sequence $\{\widetilde{\mathbf{v}}^k\}$ is converging, this sequence may be used in place of the original sequence $\{\mathbf{v}^k\}$ in the subsequent computations of $\mathbf{x}^k$. Because $\{\mathbf{v}^k\}$ is converging to $\mathbf{v}^*$ such that necessary conditions (1.15) are satisfied, the need to let μ^k go to infinity and the resulting ill-conditioning may be dispensed with.

Like Proposition 4.2.1 [11], Proposition 1 of [120] requires $\mathbf{x}^k$ to be a globally optimal solution for problem (1.8). The computation of method of multiplier sequence elements $\mathbf{x}^k$ using limit points from the sequences generated with BCD under generalized convexity conditions motivates the generalization of Proposition 1 [120] to address the the assumption where optimal solutions $\mathbf{x}^k$

for problem (1.8) may be 1) nonglobally optimal and 2) inexact. Furthermore, rate-of-convergence analysis is desirable, including the questions of when the sequences $(\mathbf{x}^k, \widetilde{\mathbf{v}}^k)$ generated in the manner of Proposition 1 [120] actually have limit points.

Lagrangian coordination techniques related to method of multipliers are also developed in the context where the sequence elements $\mathbf{x}^k$ are computed inexactly in the manner resulting from the use (quasi)-Newton step updates. Tapia [106] introduces diagonalized multiplier methods, where, given the augmented Lagrangian $L(\mathbf{x}, \mathbf{v}, \mu) := f(\mathbf{x}) + \mathbf{v}^T \mathbf{h}(x) + \frac{\mu}{2} \|\mathbf{h}(\mathbf{x})\|_2^2$, a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ is generated with the update schemes

$$\begin{aligned} \mathbf{v}^{k+1} &= U(\mathbf{x}^k, \mathbf{v}^k, \mu^k \,:\, H_k) \\ \mathbf{x}^{k+1} &= \mathbf{x}^k - H_k \nabla_{\mathbf{x}} L(\mathbf{x}^k, \mathbf{v}^{k+1}, \mu^k) \\ H_{k+1} &= \mathcal{H}(\mathbf{x}^{k+1}, \mathbf{x}^k, \mathbf{v}^{k+1}, \mu^{k+1}, H_k), \end{aligned}$$

where $\{H_k\}$ is a sequence of positive-definite approximations to the Hessian $\nabla^2_{\mathbf{x},\mathbf{x}} L$. Various updating formulas U for multiplier vector $\mathbf{v}$ are stated and analyzed. Bertsekas [11] also addresses the case of $\mathbf{x}^k$ obtained with a quasi-Newton descent step where step-sizes for computing $\mathbf{x}^k$ are obtained with the Armijo rule. The approaches based on quasi-Newton steps with Armijo rule step-sizes assume that the augmented Lagrange problem is not decomposed. A similar analysis, in the setting of BCD decomposition under generalized convexity assumptions and nonseparability of the objective functions, is highly desirable from a theoretical and practical perspective, and to our knowledge has not been carried out.

The application of Gauss-Seidel methods as tools in single objective MDO motivates additional research in the convergence analysis for ADMM and for the integration of BCD with the method of multipliers. As noted earlier, the convergence of ADMM in the multi-block case where $m > 2$ is an active area of research whose results have benefits for MDO. Convergence analysis for the method of multipliers has been extensively developed under various assumptions for problems of the form (1.9), but these assumptions do not adequately address the requirements of MDO. In particular, convergence of the method of multipliers is not well-examined under the assumptions of 1) proper containment for feasible set $X \subset \mathbb{R}^n$ 2) that each update $\mathbf{x}^k$ is computed inexactly due to its computation by the BCD method, and 3) that updates $\mathbf{x}^k$ are computed with non-globally optimal solutions for (1.8).

Furthermore, the application of these methods to the distributed and coordinated computation of efficient points for MOPs requires an SOP reformulation. Even more challenging, in the case of multiobjective MDO, this SOP reformulation needs to be compatible with the nonintegrable reality *as reflected in the requirements of ADMM and BCD*. For example, the use of ϵ-constraint method may lead to the introduction of global, nonconvex constraints, thus violating the requirements for applying ADMM and BCD. SOP reformulations that do not add additional constraints, such as the weighted-sum method and a quadratic scalarization, become an appealing option but have to be implemented in such a manner so that the boundaries delimiting inter-subproblem knowledge are respected. In conclusion, the application of SOP reformulations to MOPs for use in MDO needs to be examined in the context of the Gauss-Seidel and Lagrangian coordination techniques used to address the requirements of MDO.

1.1.4 Parameter estimation

The modeling of time-dependent properties of viscoelastic materials such as stress, and the resulting deformation referred to as strain, is well-developed mathematically [116, 96, 111, 122].

Viscoelastic materials have memory in the sense that stress applied in the past can affect strain at the present moment. The introduction of time dependence or memory effect leads to the analysis of Volterra's equation of second type [116, 96] that models the dependence of stress as a functional of strain

$$\varphi(\varepsilon(t)) = \sigma(t) + \int_0^t K(\mathbf{p}, t-\tau)\sigma(\tau)d\tau, \tag{1.16}$$

where the response functional $\varphi(\varepsilon) = E\varepsilon$ is linear in ε; the passing of time in hours is denoted by t; the applied stress in megapascals (MPa) at time t is denoted by $\sigma(t)$; and the function $K(\mathbf{p}, \cdot) : \mathbb{R} \to \mathbb{R}$ is a kernel parameterized by a vector of parameters $\mathbf{p}$. In practice, (1.16) models the relations between time, stress, and strain successfully for a wide range of materials such as polymers, metals, and composites [96, 111, 122].

One of the most effective and universal kernels $K(\mathbf{p}, t)$ is based on the exponential of an arbitrary order function [96]

$$K(\mathbf{p}, t) := \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n \ t^{n(1-\alpha)}}{\Gamma\left[(1-\alpha)(n+1)\right]}, \tag{1.17}$$

where $\mathbf{p} = [\alpha, \beta, \lambda]$ is the vector of parameters. The exponential of arbitrary order operators combine several important features [96, 15]:

- The initial moment singularity at time $t = 0$ is integratable;
- The asymptotic exponential behavior with time $t \to \infty$;
- The resolvent operator is the same type of exponential of arbitrary order with a different set of defining parameters.

Using the kernel given in (1.17) together with the assumption that the stress function $\sigma(t) := \sigma$ is a fixed known constant, then the integral in (1.16) can be evaluated, and so equation (1.16) becomes

$$\varphi(\varepsilon(t)) = \sigma \left[1 + \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n \, t^{(1-\alpha)(n+1)}}{\Gamma\left[(1-\alpha)(n+1)+1\right]} \right]. \tag{1.18}$$

Thus, equation (1.18) is used to model the relation between time t, strain ε, and load stress σ as parameterized by material-specific values for parameters $\mathbf{p} = [\alpha, \beta, \lambda]$. The parameters $\mathbf{p}$ that are not known beforehand are estimated with least-squares minimization techniques. Two least-squares problems (LSP) are considered. Denoting the right-hand side of equation (1.18) by the model function $m(\mathbf{p}, t)$, the time domain least-squares problem is given by

$$\min_{\mathbf{p} \in P} \sum_{i=1}^{N} \left(m(\mathbf{p}, t_i) - \varphi_i \right)^2 \tag{1.19}$$

where points (t_i, φ_i), $i = 1, \ldots, N$, are experimental observations of responses φ_i taken at time t_i from tested materials.

If a regression $r(t)$ is applied to the observations (t_i, φ_i), $i = 1, \ldots, N$, then, substituting the regression $r(t)$ for the response functional $\varphi(\varepsilon(t))$ in equation (1.18) and taking the Laplace transform of both sides (1.18) results in the following least-squares problem. Denoting $R(s)$ and $M(\mathbf{p}, s)$ as the Laplace transforms of time domain functions $r(t)$ and $m(\mathbf{p}, t)$, respectively, then the Laplace domain least-squares problem is given by

$$\min_{\mathbf{p} \in P} \sum_{s \in S_N} \left| M(\mathbf{p}, s) - R(s) \right|^2 \Delta(s). \tag{1.20}$$

The use of the least-squares problem (1.20) in place of (1.19) is introduced in [96, 122] with the motivation that the transformed model $M(\mathbf{p}, s)$ is of a simpler form than the time domain model

$m(\mathbf{p}, t)$. In particular, the series in the time domain model $m(\mathbf{p}, t)$ becomes a geometric series in the Laplace domain model $M(\mathbf{p}, s)$ and so it may be written in a closed-form.

The Laplace domain least-squares problem for obtaining optimal parameter estimates is applied in [122] using a sample of positive real-valued s_i. In general, the Laplace variable s takes its value from the open half-plane $H = \{s \in \mathbb{C} : Re(s) > 0\}$, and the use of the Euclidean 2-norm in the statement of problem (1.20) make it clear that problem (1.20) makes sense as a minimization problem over real-valued decision variables. Thus, the use of complex-valued s in formulating the problem (1.20) merits examination.

Furthermore, the relationship between the time domain LSP and the Laplace domain LSP is unclear, and so a mathematical foundation is needed to describe the relationship between the time domain LSP and the Laplace domain LSP. This has the practical implication of evaluating the use of the Laplace domain LSP as a tool for obtaining optimal parameter estimates for the time domain model.

1.2 Research Goal

In light of the state of art described in Section 1.1, the following research goals are stated in the two areas: multiobjective, multidisciplinary optimization; and Laplace domain estimation of optimal parameter estimates for time domain models.

1.2.1 Multidisciplinary, multiobjective optimization algorithms

As was seen in the last section, much has been accomplished mathematically and algorithmically in multiobjective optimization, and in the coordination and distribution of single objective optimization. But these results have yet to be adapted and integrated to meet the needs of multiobjective MDO. Therefore, the research goals for multiobjective MDO are stated as follows:

1. Refine the theory of convergence for the coordinated, Gauss-Seidel computations applied in a single objective MDO setting. In particular, the convergence analysis of an integration of BCD decomposition and Lagrangian coordination using the the method of multipliers requires an extension of the convergence analysis found in [10, 11, 120].

2. Adapt single objective reformulation of MOPs for application to nonconvex, multiobjective

MDO, and examine necessary and/or sufficient conditions for the optimal solutions to these scalarized SOPs to be efficient for the AiO MOP.

3. Integrate the above adaption of scalarization with the above Lagrangian coordination/Gauss-Seidel refinement. Analyze sequences generated by algorithms using theory from ADMM, BCD, the method of multipliers, and scalarized SOP reformulations for MOPs. Discuss presence of limit points, and conditions under which various necessary conditions for efficiency are satisfied by limit points. Discuss the role of this theory against the issue of limited communication and distribution of design control within an algorithmic multiobjective, multidisciplinary optimization setting.

4. Based on the above integrations, develop algorithms for the distributed computation of efficient points for MOPs, and tie these algorithms to previously developed theory relating efficient sets of AiO MOPs to efficient sets of decomposed MOPs.

5. Apply the proposed algorithms to mathematical examples and to real-world engineering design problems. One such application is to the multiobjective, multidisciplinary design of an automobile using a lithium-ion battery. The design of such an automobile highlights the engineering challenges of optimally packaging components within a system where the component shapes are allowed to vary in order to optimize the functionality of each component. Both the optimal placement of components, and the optimal design of the components themselves, are measured by multiple criteria. The simultaneous placement of components and the design of components are conceptually distinct, require distinct algorithmic concepts, and yet have an unavoidable interaction, all of which fit the paradigm of multiobjective MDO.

   The component whose design receives the most attention is the lithium-ion battery. The optimal functioning of the lithium-ion battery requires the maintenance of a temperature distribution that is not only as on target as possible, but that is also as evenly distributed as possible. These conditions depend on the internal layout of the lithium-ion battery, which in turn determines its shape and size as it is placed in the underhood of the automobile. Thus, the apt application of the equitable concept of optimality to the design of the lithium-ion battery is also to be explored.

1.2.2 Parameter estimation problem

For the parameter estimation problem, equation (1.18) was stated in Section 1.1.4 describing the time-dependent relationship between stress and strain where the memory effect is modeled with the Rabotnov kernel. A Laplace domain LSP for estimating parameters $\mathbf{p}$ for the time domain model $m(\mathbf{p}, t)$ was developed, whose relationship with the original time domain LSP is unclear. Furthermore, it is initially unclear how best to formulation a Laplace domain LSP. Thus, the following goals become evident:

1. Establish a mathematical foundation using the tools of functional and complex analysis by which to compare the time domain and Laplace domain least-squares parameter estimation approaches. This foundation is to be established by viewing each LSP as a minimization of distance as defined in some normed function space. Thus, the goal is to state a normed space for each LSP, and use the tools of functional analysis to describe the relationship between the two normed spaces.

2. Formulate a Laplace domain least-squares problem (1.20) that is 1) computationally stable, and 2) whose optimal solutions are parameter estimates that yield a good model-to-data fit. The main issue for addressing the first part of this goal is to develop a nonnegative-valued regression functions $r : [a, b] \to \mathbb{R}$ for the experimental data that has a closed-form expression for its Laplace transform $R : H \to \mathbb{C}$ where

   (a) the regression function r fits the data well,

   (b) evaluations of the transformed regression function R at each $s \in H$ are obtained in a computationally stable manner,

   (c) the end-behavior of the regression function r as time $t \to \infty$ is bounded from above by a linear function of t, that is, $r(t) \in O(t)$.

   Addressing the second goal depends on the choice of the finite sample set $S_N \subset H$. Thus, the question of choosing good sample sets S_N is addressed.

3. Apply the above formulated Laplace domain LSP to test data for various composite materials under different loading conditions.

1.3 Research Contribution

The following contributions have been realized in answer to the research goals in the two stated areas: multiobjective, multidisciplinary optimization; and Laplace domain estimation of optimal parameter estimates for time domain models.

1.3.1 Contributions to multidisciplinary, multiobjective optimization

The contributions in answer to the stated goals for multidisciplinary, multiobjective optimization are as follows:

1. Reprisals are given for some of the BCD proofs from [49, 114] with the goal of clarifying pertinent issues related to the integration of BCD with the method of multipliers. From a mathematical point of view, these rewritten proofs also provide a substantially different viewpoint on the key proof ideas from the proof given in [114]. Furthermore, these rewritten proofs lead to a proof of a new BCD convergence condition. This new convergence condition is furthermore integrated with some existing BCD convergence conditions yielding yet another new convergence condition.

2. An adaptation of a method of multiplier convergence result is presented for integration with BCD. This adaptation extends Proposition 1 of [120], which, in turn was an extension of Proposition 4.2.1 of [11]. The extension of the former result provides an analogous convergence result for the method of multipliers under a relaxation of the assumption that each computation of update $\mathbf{x}^k$ needs to be computed with a globally optimal solution for problem (1.8). Furthermore, this result is easily adapted to provide the same convergence result in the presence of inexact computations of updates $\mathbf{x}^k$ under the assumption that the magnitude of inexactness vanishes in the limit.

3. Sufficient conditions are formulated and proven for the optimal solutions of a certain SOP reformulation of an MOP to be efficient for the MOP. The SOP objective function is obtained from the quadratic scalarization $f_q : \mathbb{R}^p \to \mathbb{R}$ having the form

$$f_q(\mathbf{f}) = \frac{1}{2}(\mathbf{f} - \mathbf{y}^r)^T Q(\mathbf{f} - \mathbf{y}^r) + (\mathbf{f} - \mathbf{y}^r)^T \mathbf{q},$$

where Q is a positive semidefinite $p \times p$ matrix, $\mathbf{q}$ is a length p vector, and $\mathbf{y}^r$ is a reference point in the objective space. Both local and global efficiency are considered.

4. A specific type of quadratic scalarization f_q is suggested, based on desired properties of function f_q that are easily encoded into the construction of a matrix Q, a vector $\mathbf{q}$, and a reference point $\mathbf{y}^r$.

5. An **O**bjective **S**pace **D**ecomposition **A**lgorithm (OSDA) is proposed based on an integration of ADMM and the quadratic scalarization. OSDA computes efficient points for an AiO multiobjective MDO problem based on a quasiseparable decomposition originating in the objective space in the sense that the decomposition is stated in terms of the partition $\mathbf{f}(\mathbf{x}) = [\mathbf{f}_1(\mathbf{x}_1), \ldots, \mathbf{f}_m(\mathbf{x}_m)]$ of the vector of objective functions. In OSDA, scalarization is applied in two stages. In the first stage, the quadratic scalarization is applied to each subproblem (i.e., intra-subproblem scalarizations), so that each reformulated subproblem is single objective. The intra-subproblem scalarization furthermore induces a reformulation of the underlying AiO MOP. In the second stage, the weighted-sum scalarization is applied to the reformulated AiO problem (i.e., an inter-subproblem scalarization). Direct knowledge of the weighted-sum weights associated with the second stage of scalarization may be hidden from the subproblems by encoding this information into the values of the multiplier $\mathbf{v}$ and the penalty coefficient μ that are passed to each subproblem.

6. A **D**ecision **S**pace **D**ecomposition **A**lgorithm (DSDA) is proposed based on the integration of BCD, the method of multipliers, and the quadratic scalarization. DSDA computes efficient points for an AiO multiobjective MDO problem whose decomposition is based on a partition $\mathbb{R}^n = \prod_{i=1}^m \mathbb{R}^{n_i}$ of the solution space. When the AiO decomposable MOP (1.1) is also quasiseparable, then the application of the quadratic scalarization only requires the exchange of objective function values under a distributed optimization approach based on a subproblem decomposition.

7. Convergence analyses are given for OSDA and DSDA based on the theory developed for their constituent parts. This analysis is provided under the following conditions for OSDA and DSDA as shown in the following table.

Assumptions for OSDA and DSDA		
	OSDA	DSDA
X_i	closed, convex	closed, convex
$\mathbf{f}$	separable	nonseparable
f_q	continuous, convex	continuously differentiable, generalized convexity
$\mathbf{h}$	linear	continuously differentiable

8. The quadratic scalarization has been applied to compute efficient solutions for a mathematical example problem taking the form of a nonconvex MOP having four objective functions. The use of the quadratic scalarization is shown to be nontrivial for this example in the sense that many efficient solutions are computed that are not computable using the weighted-sum method.

9. This same MOP example is then decomposed into two nonintegrable biobjective subproblems. OSDA and DSDA are implemented and applied for solving the resulting multiobjective MDO problem. Where necessary, the generated solutions are tested for satisfying the developed quadratic scalarization sufficient conditions of efficiency for the AiO MOP. Plots of the efficient solutions verify that coordination between subproblem copies of the variables has been achieved, and that the region in the solution space in which the coordinated solutions lie corresponds to the region in the solution space of the efficient solutions priorly computed in the AiO (nondecomposed) setting.

10. DSDA has been applied to a bilevel automotive design problem whose decomposition is presented as two nonintegrable subproblems. One subproblem aims to optimally package the components of the automobile's underhood using a genetic algorithm, while the other subproblem addresses the optimal design of one component in particular, the lithium-ion battery, in terms of the equitable concept of optimality. Treating the design of the lithium-ion battery as a separate design problem results in the morphing of its shape and size as it is being placed in the underhood; hence the resulting coupling between the two subproblems as they are being independently solved is coordinated with DSDA. The development of such solution approaches for packaging problems where the components are subject to morphing in the design process is an important contribution in engineering design.

11. Introduced and proved the usefulness of the equitable optimality concept in engineering design. In addition to being important for modeling the optimal functioning of the lithium-ion battery, the application of the equitable preference reduces the efficient set to a subset, thus making bilevel design easier.

1.3.2 Parameter estimation problem

The contributions in answer to the stated goals for the parameter estimation problem are as follows:

1. Mathematical foundation is established by describing normed function spaces corresponding to each of the two least-squares problems:

   (a) The time domain least-squares problem is associated with the L^2 topology on the space of continuous functions $C[a, b]$ defined on a real interval $[a, b]$ equipped with the standard L^2 norm denoted by $\|\cdot\|_2$.

   (b) The Laplace domain problem is associated with a topology on the same space $C[a, b]$ but equipped with an alternative norm denoted $\|\cdot\|_S$. Since $\|\cdot\|_S$ is not a standard norm, its norm properties are verified.

   Once each least-squares problem is understood as minimization of a norm, then equivalence between the least-squares problems is identified with equivalence between the norms as defined in [92]. This norm equivalence is characterized by the existence of fixed bounding coefficients ℓ, u, $0 < \ell < u < \infty$ where the inequalities $\ell \|f\|_2 \leq \|f\|_S \leq u \|f\|_2$ are satisfied for all functions $f \in C[a, b]$.

2. It is shown that an upper bound coefficient u exists, and the absence of the lower bound coefficient $\ell > 0$ is shown through a counterexample. The implications of the existence of an upper bound coefficient $u < \infty$ but nonexistence of a lower bound ℓ are briefly discussed.

3. Problem (1.20) is formulated in such a manner so that 1) the sample set S_N, and 2) the collection of least-squares weighting coefficients $\Delta(s)$ are motivated by the development of the mathematical foundation. This formulation is applied to the test data described in [115].

1.4 The content of the disseration

Chapters 2 through 6 present work that has either been published, is under review, or is in the final stages before submission for publication. The multiobjective MDO contributions as outlined in Section 1.3 are included in Chapters 2 through 4, while the contributions related to the Laplace domain estimation of parameters are found in Chapters 5 and 6.

Chapter 2 contains the main mathematical contributions for multidisciplinary, multiobjective optimization. The content of this chapter starts with a mathematical formulation for a multiobjective problem decomposed along multidisciplinary lines, and continues with a theoretical refinement and integration of pre-existing tools to address the generation of efficient points in an MDO setting. Algorithms based on this theory are developed and applied to mathematical examples, one of which is a nonconvex multiobjective MDO problem.

The multiobjective, multidisciplinary automotive design problem motivating the developments of Chapter 2 is addressed in Chapters 3 and 4. Recall that this automotive design problem features a component packaging problem where the shape and size of the components are also varied; hence the role of multidisciplinary optimization. The shape and size of components vary as they are part of a design process whose goal is an optimal functionality that is measured with multiple criteria. Chapter 3 begins the presentation of this automotive application by addressing the optimal design of one component in particular, the lithium-ion battery independent of its placement within the underhood of the automobile. As explained in Section 1.1, the equitable optimality concept is well-suited to the requirements for the optimal functioning of the lithium-ion battery, and so this concept is introduced and applied to engineering design. Chapter 4 completes this automotive application by integrating the design process of the lithium-ion battery with the automotive packaging problem using the algorithms developed in Chapter 2. Both Chapters 3 and 4 state and discuss computational results from an engineering perspective and from a mathematical perspective.

In presenting the contributions relating to Laplace domain parameter estimation, Chapter 5 develops a mathematical foundation for comparing the two least-squares parameter estimation approaches under consideration: the time domain and the Laplace domain. With this foundation, statements regarding this comparison are formulated and proven, and an improvement in the application of the Laplace domain least-squares problem is obtained and applied to the test data.

The paper presented in Chapter 6 explores an application of the time domain least-squares

problem from a numerical perspective, and may be viewed as motivating the developments of the preceding chapter. In particular, the potential for catastrophic cancellation in the series term of the model function $m(\mathbf{p}, t)$ for certain values of parameter vector $\mathbf{p}$, and the typical ill-conditioning of the time domain least-squares problem are explored.

Following the content chapters, Appendix A, Appendix B, and Appendix C record the latest version of the code that is used to generate results and implement the algorithms provided in chapters 2, 4, and 5 of this dissertation, respectively.

1.5 Conclusion and Future Research

The contents of this dissertation make clear the need for refinement of currently available mathematical theories, and their adaptation for the needs of engineering design. The mathematical potential for such theoretical and practical refinements is realized in the following pages of this dissertation in the areas of Lagrangian coordination, Gauss-Seidel decomposition, multiobjective scalarization, and parameter estimation techniques.

On each of these fronts, the contributions of this dissertation, though nontrivial, are only a beginning. In the way of Lagrangian coordination, the (not so) simple rewriting and repackaging of the many results developed and/or presented in the textbooks [10, 11] may lead to a more integrated framework that allows additional insight. The results presented along this vein in Chapter 2 may be extended to address other necessary conditions for optimality that are met by the method of multiplier limit points. More can also be said on any constraint qualification conditions that imply the actual presence of limit points that are currently assumed to be present.

For Gauss-Seidel decomposition, the convergence of ADMM for the multi-block $m > 2$ decomposition is an active area of research as noted earlier. The integration of BCD and the method of multipliers can be extended to address a variety of manners in which BCD updates and method of multiplier updates are computed inexactly. Finally, the restrictions implied in MDO optimization problems can be motivated by concepts from computer science taking the form of object-oriented programming, and the role of mathematical tools such as Lagrangian coordination and Gauss-Seidel decomposition can be explained along these concepts.

In the area of multiobjective scalarization, comparison of the use of quadratic scalarization with the use of weighted-t^{th} power sum in multiobjective MDO is needed. Furthermore, the

(pseudo)convexifying effect of the quadratic scalarizations warrants investigation. Conditions on the objective functions, and conditions on the quadratic scalarization need to be examined to gain insight.

Although the content of Chapter 2 is motivated by engineering MDO applications, the developments in this chapter address more general types of MOP decomposition that are applicable, for example, in business and military applications that also require the formulation of certain MOPs as a collection of nonintegrable subproblems. For the former, an MOP may model the management of business activities within a large international corporation, where decisions under multiple objectives are made locally in each country so that the corporation performs at its best. For the latter, a collection of MOPs may model military mission planning and execution under partial information due to constraints in the communication bandwidth or due to required communication latencies [84].

Finally, the foundation established in Chapter 5 may be extended to examine special conditions under which the imperfect equivalence between the two least-squares parameter estimation approaches can be remedied. Applications of the Laplace domain least-squares as formulated in Chapter 5 to different models and different types of loading assumptions may also be examined.

Chapter 2

Distributed Computation of Pareto Sets

[The contents of this chapter include material in a paper to be submitted (late June 2013) to the *SIAM Journal on Optimization* titled "Distributed computation of Pareto sets"; the authors are B. Dandurand and M. M. Wiecek. This chapter includes additional material not included in the above paper.]

2.1 Introduction

This chapter addresses the computation of efficient points for multiobjective optimization problems (MOPs) in a setting where the MOP is not available to one solver in an integrated form, but rather is available to multiple solvers in terms of nonintegrable subproblems. This setting is typical, for example, in multidisciplinary optimization (MDO), which addresses engineering design problems whose complexity necessitates specialization in the design process along distinct disciplines composing a system. Many papers present applications of multiobjective MDO in various areas of engineering design [79, 91, 66, 30, 22, 74, 28]. Methodologies such as Multiobjective Collaborative Optimization [107, 95], Multiobjective Concurrent Subspace Optimization (CSSO) [61, 63, 64, 62, 60], and a bilevel method [125] have also been developed. For multilevel systems, an approach based on the use of lower-level efficient designs as targets for upper-level designs and the method of

Analytical Target Cascading is proposed in [77].

Other settings requiring the formulation of an MOP as a collection of nonintegrable subproblems include, for example, business or military applications. For the former, an MOP may model the management of activities in business within a large international corporation, where decisions under multiple objectives are made locally in each country so that the corporation performs at its best. For the latter, a collection of MOPs may model military mission planning and execution under partial information due to constraints in communication bandwidth or due to required communication latencies [84].

2.1.1 Multiobjective optimization background

Let an integrated MOP be given by

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X, \end{aligned} \tag{2.1}$$

where the objective function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$ is vector-valued, $X \subseteq \mathbb{R}^n$, the feasible set is given by $X \cap \{\mathbf{x} \in \mathbb{R}^n : \mathbf{h}(\mathbf{x}) = \mathbf{0}\}$, and the specialized equality constraints $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ are defined through the constraint function $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^q$. The vector space $\mathbb{R}^n$ is referred to as the *decision space*, and the vector space $\mathbb{R}^p$ is referred to as the *objective space*. The vector-valued function $\mathbf{f} = [f_1, \ldots, f_p]$ consists of component functions $f_i : \mathbb{R}^n \to \mathbb{R}$ for $i = 1, \ldots, p$, and the image $Y \subseteq \mathbb{R}^p$ is defined by

$$Y := \mathbf{f}(X) := \{\mathbf{y} \in \mathbb{R}^p : \mathbf{y} = \mathbf{f}(\mathbf{x}) \text{ for some } \mathbf{x} \in X\}.$$

Due to a conflict typically present among the objective functions f_i for $i = 1, \ldots, p$, there may be no solution $\mathbf{x} \in X$ minimizing every objective function f_i simultaneously. Thus, optimality for MOP (2.1) is understood in terms of Pareto optimality [89, 34]. The following vector relations are useful

for the description of Pareto optimality. For vectors $\mathbf{y}^1, \mathbf{y}^2 \in \mathbb{R}^p$, define

$$\begin{aligned}
&\mathbf{y}^1 < \mathbf{y}^2 \text{ if } y_i^1 < y_i^2 \text{ for all } i = 1, \ldots, p; \\
&\mathbf{y}^1 \leqq \mathbf{y}^2 \text{ if } y_i^1 \leq y_i^2 \text{ for all } i = 1, \ldots, p; \\
&\mathbf{y}^1 \leq \mathbf{y}^2 \text{ if } \mathbf{y}^1 \leqq \mathbf{y}^2 \text{ and } \mathbf{y}^1 \neq \mathbf{y}^2.
\end{aligned}$$

From these relations, the following conical sets are defined:

$$\begin{aligned}
\mathbb{R}^p_< &:= \{\mathbf{y} \in \mathbb{R}^p \,:\, \mathbf{y} < \mathbf{0}\} \\
\mathbb{R}^p_{\leqq} &:= \{\mathbf{y} \in \mathbb{R}^p \,:\, \mathbf{y} \leqq \mathbf{0}\} \\
\mathbb{R}^p_{\leq} &:= \{\mathbf{y} \in \mathbb{R}^p \,:\, \mathbf{y} \leq \mathbf{0}\}.
\end{aligned}$$

The relations $>$, $\geqq$, $\geq$ and sets $\mathbb{R}^p_>$, $\mathbb{R}^p_{\geqq}$, $\mathbb{R}^p_{\geq}$ are defined analogously. The negations of the relations $<$, $\leqq$, and $\leq$ are denoted by $\not<$, $\not\leqq$, and $\not\leq$ and are given by

$$\begin{aligned}
&\mathbf{y}^1 \not< \mathbf{y}^2 \text{ if } y_i^1 \geq y_i^2 \text{ for some } i = 1, \ldots, p; \\
&\mathbf{y}^1 \not\leqq \mathbf{y}^2 \text{ if } y_i^1 > y_i^2 \text{ for some } i = 1, \ldots, p; \\
&\mathbf{y}^1 \not\leq \mathbf{y}^2 \text{ if } \mathbf{y}^1 \not\leqq \mathbf{y}^2 \text{ or } \mathbf{y}^1 = \mathbf{y}^2.
\end{aligned}$$

Note that for $p \geq 2$, the relation $\not<$ is *not* equivalent to the relation $\geqq$, and the relation $\not\leqq$ is *not* equivalent to the relation $>$.

A solution $\mathbf{x}^* \in X$ is *weakly efficient* for MOP (2.1) if $\mathbf{f}(\mathbf{x}) \not< \mathbf{f}(\mathbf{x}^*)$ for all $\mathbf{x} \in X$. A solution $\mathbf{x}^* \in X$ is *efficient* for MOP (2.1) if $\mathbf{f}(\mathbf{x}) \not\leqq \mathbf{f}(\mathbf{x}^*)$ for all $\mathbf{x} \in X$. (This strengthens the notion of weak efficiency by adding the requirement that if any improvement in one objective's value $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ is obtained by the substitution of solution $\mathbf{x}^*$ with solution $\mathbf{x}$, then deterioration of at least one other objective's value $f_j(\mathbf{x}) > f_j(\mathbf{x}^*)$ must also occur.) A solution $\mathbf{x}^* \in X$ is *locally efficient* if there exists a neighborhood $\mathcal{N}(\mathbf{x}^*)$ of $\mathbf{x}^*$ such that $\mathbf{f}(\mathbf{x}) \not\leqq \mathbf{f}(\mathbf{x}^*)$ for all $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*) \cap X$. For (weakly) efficient solutions $\mathbf{x}^*$, outcome $\mathbf{f}(\mathbf{x}^*)$ is (weak) *Pareto*. The set of (weak) Pareto outcomes is denoted (Y_{wN}) Y_N.

Approaches to generating solution sets of MOPs (2.1) fall into two categories: scalarizing and nonscalarizing methods [34, 36]. The scalarizing methods convert an MOP into a single objective

program (SOP) by replacing the vector-valued objective $\mathbf{f}$ with a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, and by the possible addition of constraints. SOP reformulations of MOPs such as the weighted-sum method [46], the weighted-t power method [119, 75], and the weighted-quadratic method [112] introduce scalarizing functions f and do not introduce additional constraints, while SOP reformulations that introduce additional constraints include, for example, the ε-constraint method [17] and Benson's method [8, 18]. Under certain conditions, the optimal solutions of these SOPs yield efficient solutions for the original MOP. Each SOP instance typically yields one efficient solution for the MOP. Thus, a finite sample approximation of the efficient set may be obtained through the use of multiple SOP computations of efficient points for the MOP.

Nonscalarizing methods for computing efficient solutions include approaches using optimality concepts other than Pareto, (such as lexicographic methods and max-ordering methods), descent methods transferred from nonlinear programming, and set-oriented methods. Lexicographic methods [33], for example, apply single objective minimization in a recursive manner along some ordering of objective functions reflecting their importance. In the steepest descent methods, a search direction is derived from the gradient and/or Hessian information of the objective functions of the MOP. These methods include variants for the constrained and unconstrained case (see [39, 31, 38], for example). Set-oriented methods [41, 43, 42, 35, 99], in contrast to all previously presented approaches, find a solution set of the MOP without using scalarizing functions or other optimality concepts.

2.1.2 Problem Statement

The problem of generating efficient solutions for MOP (2.1) when the data of (2.1) is not explicitly available to any one solver is now described. To this end, MOP (2.1) is assumed to be of the following All-in-One (AiO) decomposable form given by

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & \mathbf{f}\left(\mathbf{x}_1,\ldots,\mathbf{x}_m\right) \\ \text{s.t.} \quad & \mathbf{h}\left(\mathbf{x}_1,\ldots,\mathbf{x}_m\right) = \mathbf{0} \\ & \mathbf{x}_i \in X_i, \quad i = 1,\ldots,m, \end{aligned} \tag{2.2}$$

where $X := \prod_{i=1}^{m} X_i$, $X_i \subseteq \mathbb{R}^{n_i}$, $i = 1,\ldots,m$, and $\mathbf{x} := [\mathbf{x}_1,\ldots,\mathbf{x}_m]$. The AiO decomposable format given by (2.2) corresponds to the division of MOP (2.1) along interdisciplinary boundaries so that each subproblem i, $i = 1,\ldots,m$, has its own decision space $\mathbb{R}^{n_i}$ and its own local feasible

set $X_i \subseteq \mathbb{R}^{n_i}$. The constraint function $\mathbf{h}$ has an interdisciplinary scope and typically arises from the copying of variables that may be necessary in order to obtain the decomposable format (2.2). Thus, a distinction is made between the *decomposable* constraint $\mathbf{x} \in X := \prod_{i=1}^{m} X_i$ and the *coordinating* constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$.

By assumption, the AiO decomposable MOP (2.2) is not known explicitly to any solver, but is implicitly available to multiple solvers through knowledge of the following nonintegrable subproblems

$$\begin{aligned} \min_{\mathbf{x}_i} \quad & \mathbf{f}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) = \mathbf{0} \\ & \mathbf{x}_i \in X_i, \end{aligned} \tag{2.3}$$

where the objective function $\mathbf{f}(\,\cdot\,, \ddot{\mathbf{x}}_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}^p$ and the constraint function $\mathbf{h}(\,\cdot\,, \ddot{\mathbf{x}}_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}^q$ are parameterized by fixed values $\ddot{\mathbf{x}}_{\neg i} := (\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_{i-1}, \ddot{\mathbf{x}}_{i+1}, \ldots, \ddot{\mathbf{x}}_m)$ with $\ddot{\mathbf{x}}_j \in X_j$ for $j \neq i$. ($\neg i$ may be read as "not i", referring to the indices $j = 1, \ldots, m$ where $j \neq i$.) Subproblems (2.3) feature the following knowledge and control assumptions:

1. Each subproblem i is associated with a solver having control over its decision variable $\mathbf{x}_i$ and knowledge of its feasible set X_i.

2. Each subproblem i solver may have either direct or indirect knowledge of fixed values $\ddot{\mathbf{x}}_j \in X_j$ determined from the other subproblem $j \neq i$ solvers.

3. For each fixed solution $\ddot{\mathbf{x}}_j \in X_j$, $j \neq i$, each subproblem i solver has knowledge of its vector-valued objective function $\mathbf{f}(\,\cdot\,, \ddot{\mathbf{x}}_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}^p$ and of its vector-valued constraint function $\mathbf{h}(\,\cdot\,, \ddot{\mathbf{x}}_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}^q$. Direct knowledge of the AiO functions $\mathbf{f} : \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \mathbb{R}^p$ and $\mathbf{h} : \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \mathbb{R}^q$ are *not* available to any single solver.

4. However, the AiO functions $\mathbf{f} : \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \mathbb{R}^p$ and $\mathbf{h} : \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \mathbb{R}^q$ are available to a *master coordinator*. The master coordinator is responsible for coordinating and updating the subproblem (2.3) information and distributing this information to the solvers.

The solution approaches developed in multiobjective optimization assume explicit knowledge of an MOP of the form (2.1) or (2.2), and so a new theory and methods are required to address the

distributed computation of efficient solutions for the AiO decomposable problem (2.2) with the use of coordinated computations based on the subproblems (2.3).

The coordinated distribution of optimization has a strong mathematical foundation in the single objective optimization setting with the application [113, 120] of well-studied Gauss-Seidel decomposition/coordination techniques such as the alternating direction method of multipliers (ADMM) [47, 40, 9, 32, 69, 14], the block coordinate descent (BCD) method [118, 94, 12, 9, 49, 114], and the method of multipliers [56, 93, 10, 11]. Most of the rigorously developed convergence theory for the BCD method and the method of multipliers is developed without consideration of the applied requirements of coordinated, distributed optimization, while the convergence theory that does consider this [120] may be extended to address the presence of generalized convexity in the objective functions. This work adapts and extends some of these results to be applied to MOPs that have been reformulated as SOPs.

The coordinated, distributed approaches to single objective optimization mentioned above motivate the adaptation of existing SOP reformulation techniques for multiobjective optimization. Such scalarizations need to respect limits placed on the ability of subproblems (2.3) to exchange information, and so any feature of an SOP reformulation (such as the introduction of new constraints) that could potentially violate such principles needs to be examined. Due to its simplicity, the weighted-sum method is easily adapted for use in the coordinated and distributed computation of efficient points for multiobjective problem. When all problem functions are convex and the constraint set X is convex, the weighted-sum method is sufficient to compute any efficient solution to MOP (2.1) [46]. In the case where f is more generally nonconvex, it is desirable to explore the use of other scalarization techniques. One such possibility is the weighted-t power method [75]. Another possibility is a quadratic scalarization, which is initially proposed in [112] as a possible form taken by the dual of the weighted-Chebyshev method. In this chapter, the role of the quadratic scalarization in the coordinated, distributed computation of efficient solutions of MOPs is examined, while conditions are developed under which the optimal solutions obtained with the use of this type of SOP reformulation are efficient for the MOP.

The foundational developments in this paper proceed in three stages: scalarization, coordination, and distribution of optimization. Scalarization and coordination are addressed by the master coordinator, while the distributed solution approach is carried out by the subproblem solvers based on recurrent updates of the subproblem information passed from the master coordinator. In Section

2.2, scalarization is addressed by posing SOP reformulations of MOP (2.1) having the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_s(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X, \end{aligned} \tag{2.4}$$

where the scalarized objective function $f_s : \mathbb{R}^n \to \mathbb{R}$ is real-valued, and X and $\mathbf{h}$ are modified as necessary. SOP reformulations are developed with the needs of coordination and distributed computation in mind while addressing the presence of nonconvex objective functions.

Once a suitable SOP reformulation (2.4) of (2.1) is established, the first step toward imposing a decomposable structure on the SOP reformulation (2.4) is realized in Section 2.3 by addressing solution techniques for the SOP (2.4) that rely on Lagrangian relaxation of the coordinating constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ resulting in the augmented Lagrangian problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_s(\mathbf{x}) + \mathbf{v}^T \mathbf{h}(\mathbf{x}) + \frac{\mu}{2} \|\mathbf{h}(\mathbf{x})\|_2^2 \\ \text{s.t.} \quad & \mathbf{x} \in X, \end{aligned} \tag{2.5}$$

where $\mathbf{v} \in \mathbb{R}^q$ is a vector of multipliers associated with the relaxed coordinating constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$, and $\mu > 0$ is a penalty coefficient determining the penalty magnitude resulting from violation of the coordinating constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$. Parameters $\mathbf{v}$ and μ are iteratively updated using adaptations of Lagrangian coordination techniques such as the method of multipliers [10, 11] so that optimal solutions for (2.5) are also optimal solutions for (2.4) when there is no duality gap in the setting of quadratic Lagrangian duality [97].

Once the coordinating constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ is relaxed, only the decomposable constraint $\mathbf{x} \in X$ is directly enforced in problem (2.5). Letting

$$f(\mathbf{x}) := f_s(\mathbf{x}) + \mathbf{v}^T \mathbf{h}(\mathbf{x}) + \frac{\mu}{2} \|\mathbf{h}(\mathbf{x})\|_2^2,$$

we have a problem of the form

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & f(\mathbf{x}_1, \ldots, \mathbf{x}_m) \\ \text{s.t.} \quad & \mathbf{x}_1 \in X_1, \ldots, \mathbf{x}_m \in X_m. \end{aligned} \tag{2.6}$$

The distributed computation of an optimal solution for problem (2.6) using the block coordinate descent (BCD) method is developed in Section 2.4. BCD convergence is studied with the requirement that BCD be integrated with a coordination and scalarization approach.

The development and integration of scalarization, coordination, and distributed computation in Section 2.5 yield MultiObjective Decomposition Algorithms (MODAs) for computing efficient solutions for an AiO decomposable MOP (2.2) while using only references to the subproblems (2.3). Applications of the MODAs to a couple of example problems, one convex, the other nonconvex, are given in Section 2.6. Section 2.7 concludes this paper and describes future work.

2.2 Decomposable single objective reformulations

While the generation of efficient solutions for MOPs (2.1) presented as a single integrated problem is well-studied, the generation of efficient solutions for MOPs presented as nonintegrable multiobjective subproblems (2.3) is not addressed. In this section, SOP reformulations (2.4) of MOP (2.1) are developed to 1) address nonconvexity in the objective functions, and 2) be compatible with the coordinated, distributed solution approach developed in Sections 2.3 and 2.4.

The weighted-sum scalarization obtained by replacing objective vector $\mathbf{f}$ in (2.1) with scalar objective $f_w := \sum_{i=1}^{p} w_i f_i$, $w_i \geq 0$, $i = 1, \ldots, p$, is well-studied [46, 34] and analytically simple. The resulting SOP reformulation is given by

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_w(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X, \end{aligned} \tag{2.7}$$

where $\mathbf{w} := [w_1, \ldots, w_p]$, $\mathbf{w} \geq \mathbf{0}$. Because the weighted-sum reformulation (2.7) does not require any modification of the original constraints $\mathbf{x} \in X$ and $\mathbf{h}(\mathbf{x}) = \mathbf{0}$, problem (2.7) is compatible with coordinated, distributed solution approaches that are based on the decomposition $X := \prod_{i=1}^{m} X_i$ and coordination by $\mathbf{h}$. When the individual objective functions f_i, $i = 1, \ldots, p$, are convex, then the weighted-sum scalarization is sufficient for generating any efficient solution (see, e.g., Theorem 4.1 of [34]).

However, when some components f_i of $\mathbf{f}$ are nonconvex over $X := \prod_{i=1}^{m} X_i$, there may be

efficient points for MOP (2.1) that cannot be computed as optimal solutions for any problem of the form (2.7), $\mathbf{w} \geq 0$. Other scalarization methods that address the issue of nonconvexity include the ϵ-constraint method [17] and the weighted-Chebyshev method [121, 124, 13, 103, 102, 67]. Both of these methods introduce nonconvex, nondecomposable constraints when the objective functions f_i, $i = 1, \ldots, p$, are nonconvex, and so these new constraints cannot be directly enforced in the distributed solution approach of Section 2.4, where the decomposition $X := \prod_{i=1}^{m} X_i$ assumes that each X_i, $i = 1, \ldots, m$, is convex. A solution approach based on the treatment of such nonconvex, nondecomposable constraints as augmented Lagrangian-relaxed constraints is addressed in Section 2.3.

The weighted-t power method [119, 75] is similar to the weighted-sum method, except that each objective function f_i, $i = 1, \ldots, p$, is replaced with some power f_i^t, $t > 0$, $i = 1, \ldots, p$. (Without loss of generality, it may be assumed, by translating $\mathbf{f}$ as necessary, that $\mathbf{f}(\mathbf{x}) > \mathbf{0}$ for all $\mathbf{x} \in X$.) Thus, problem (2.4) takes the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{i=1}^{p} w_i \, [f_i(\mathbf{x})]^t \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X. \end{aligned} \tag{2.8}$$

Li [75] shows that any efficient point for MOP (2.1) meeting certain sufficient conditions may be computed using the weighted-t power method for some finite $t > 0$.

Like the weighted-sum method, the weighted-t power method is compatible with coordinated, distributed solution approaches that are based on the decomposition $X := \prod_{i=1}^{m} X_i$ and coordination by $\mathbf{h}$. While the weighted-sum method does not address nonconvexity well, the weighted-t power method may introduce a high degree of nonlinearity for large values of t. In seeking a compromise between these two features, another related approach is considered based on a quadratic form transformation of the objective space. This approach, referred to as the *quadratic scalarization method*, generalizes the $t = 2$ weighted-t power method, and when suitably formulated, is likewise compatible with coordinated, distributed solution approaches based on the decomposition $X := \prod_{i=1}^{m} X_i$ and coordination by $\mathbf{h}$.

The quadratic scalarization method applies an SOP reformulation to MOP (2.1) given by

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_q(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X, \end{aligned} \tag{2.9}$$

where

$$f_q := \frac{1}{2} \left(\mathbf{f} - \mathbf{y}^r\right)^T Q \left(\mathbf{f} - \mathbf{y}^r\right) + \left(\mathbf{f} - \mathbf{y}^r\right)^T \mathbf{q}, \tag{2.10}$$

Q is a positive semidefinite matrix, $\mathbf{q}$ is a $p \times 1$ vector, and $\mathbf{y}^r$ is a reference point in the objective space $\mathbb{R}^p$. The application of the scalarized reformulated problem (2.9) is motivated by the dual of the weighted-Chebyshev method that has been shown in [112] to be equivalent, under certain conditions, to solving a problem of the form (2.9).

Reference points $\mathbf{y}^r$ are commonly taken to be either ideal points $\mathbf{y}^I$ for MOP (2.1) or utopia points $\mathbf{y}^U$ for (2.1). The point $\mathbf{y}^I$, whose components y_i, $i = 1, \ldots, p$, are given by $\mathbf{y}_i := \min_{\mathbf{x} \in X} f_i(\mathbf{x})$, is called an *ideal point* for MOP (2.1). Any point $\mathbf{y}^U = \mathbf{y}^I - \epsilon$ for some $\epsilon \in \mathbb{R}^p_>$ is a *utopia point* for MOP (2.1).

2.2.1 Properties of quadratic scalarization

Lemmas 1 and 2 and Propositions 1 and 2 address the use of the quadratic scalarization for the computation of efficient points for MOPs. The chief sufficient condition addressed in these propositions is membership in the following critical set defined by

$$C_{Q,\mathbf{q},\mathbf{y}^r} := \left\{\mathbf{y} \,:\, Q\left(\mathbf{y} - \mathbf{y}^r\right) + \mathbf{q} > \mathbf{0}\right\}. \tag{2.11}$$

The plots of Fig. 2.1 depict the parabolic level curves for two quadratic scalarization examples of the form (2.10) along with their corresponding critical sets (2.11).

Lemma 1. *Given a function* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$*, let* $f_q : \mathbb{R}^p \to \mathbb{R}$ *be defined as in* (2.10) *where* Q *is a* $p \times p$ *positive semidefinite matrix,* $\mathbf{q}$ *a* $p \times 1$ *column vector, and* $\mathbf{y}^r \in \mathbb{R}^p$ *is a reference point in the objective space. Let* $\mathbf{y}^1 := \mathbf{f}(\mathbf{x}^1)$ *and* $\mathbf{y}^2 := \mathbf{f}(\mathbf{x}^2)$ *for some* $\mathbf{x}^1, \mathbf{x}^2 \in X$*. If* $\mathbf{y}^1 \leq \mathbf{y}^2$ *and* $\mathbf{y}^1 \in C_{Q,\mathbf{q},\mathbf{y}^r}$*, then* $f_q(\mathbf{x}^1) < f_q(\mathbf{x}^2)$*.*

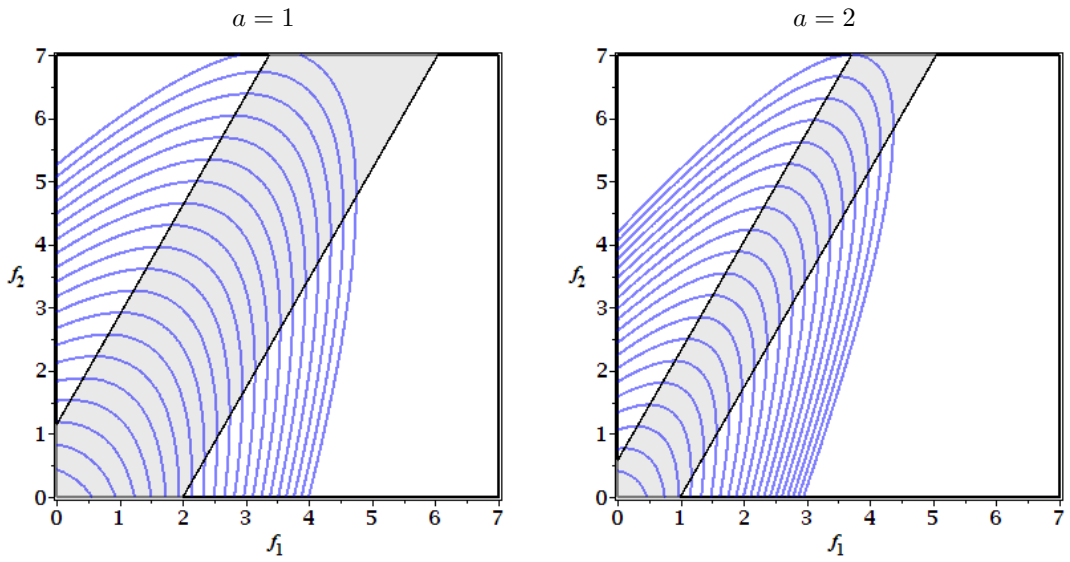

Figure 2.1: Level curves of two quadratic scalarizations (2.10) in the objective space with the critical sets $C_{Q,\mathbf{q},\mathbf{y}^r}$ shaded in gray, where $Q = a\left[-1/2,\ \sqrt{3}/2\right]^T\left[-1/2,\ \sqrt{3}/2\right]$, $\mathbf{q} = \left[\sqrt{3}/2,\ 1/2\right]^T$, and $\mathbf{y}^r = [0,0]^T$

Proof. Using (2.10), calculate

$$\begin{aligned}
f_q(\mathbf{x}^2) &= \frac{1}{2}\left(\mathbf{y}^2-\mathbf{y}^r\right)^T Q\left(\mathbf{y}^2-\mathbf{y}^r\right)+\left(\mathbf{y}^2-\mathbf{y}^r\right)^T\mathbf{q}\\
&= \frac{1}{2}\left(\mathbf{y}^2-\mathbf{y}^1+\mathbf{y}^1-\mathbf{y}^r\right)^T Q\left(\mathbf{y}^2-\mathbf{y}^1+\mathbf{y}^1-\mathbf{y}^r\right)+\left(\mathbf{y}^2-\mathbf{y}^1+\mathbf{y}^1-\mathbf{y}^r\right)^T\mathbf{q}\\
&= \frac{1}{2}\left(\mathbf{y}^2-\mathbf{y}^1\right)^T Q\left(\mathbf{y}^2-\mathbf{y}^1\right)+\left(\mathbf{y}^2-\mathbf{y}^1\right)^T\left(Q\left(\mathbf{y}^1-\mathbf{y}^r\right)+\mathbf{q}\right) \qquad (2.12)\\
&+ \frac{1}{2}\left(\mathbf{y}^1-\mathbf{y}^r\right)^T Q\left(\mathbf{y}^1-\mathbf{y}^r\right)+\left(\mathbf{y}^1-\mathbf{y}^r\right)^T\mathbf{q}\\
&\geq \frac{1}{2}\left(\mathbf{y}^1-\mathbf{y}^r\right)^T Q\left(\mathbf{y}^1-\mathbf{y}^r\right)+\left(\mathbf{y}^1-\mathbf{y}^r\right)^T\mathbf{q} \qquad (2.13)\\
&= f_q(\mathbf{x}^1).
\end{aligned}$$

Inequality (2.13) holds since the first term of (2.12) satisfies

$$\frac{1}{2}\left(\mathbf{y}^2-\mathbf{y}^1\right)^T Q\left(\mathbf{y}^2-\mathbf{y}^1\right)\geq 0$$

by the positive semidefiniteness of Q, and the second term also satisfies

$$\left(\mathbf{y}^2-\mathbf{y}^1\right)^T\left(Q\left(\mathbf{y}^1-\mathbf{y}^r\right)+\mathbf{q}\right)>0$$

by the assumptions that $\mathbf{y}^1 \leq \mathbf{y}^2$ and $\mathbf{y}^1 \in C_{Q,\mathbf{q},\mathbf{y}^r}$. □

Note that the converse of Lemma 1 does not hold. That is, $f_q(\mathbf{x}^1) < f_q(\mathbf{x}^2)$ may hold while $\mathbf{y}^1 \not\leq \mathbf{y}^2$. However, a partial converse of Lemma 1 exists, and this is stated in Lemma 2.

Lemma 2. *Given a function* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$, *let* $f_q : \mathbb{R}^p \to \mathbb{R}$ *be defined as in* (2.10) *where* Q *is a* $p \times p$ *positive semidefinite matrix,* $\mathbf{q}$ *is a* $p \times 1$ *column vector, and* $\mathbf{y}^r \in \mathbb{R}^p$ *is a reference point in the objective space. Let* $\mathbf{y}^1 := \mathbf{f}(\mathbf{x}^1)$ *and* $\mathbf{y}^2 := \mathbf{f}(\mathbf{x}^2)$ *for some* $\mathbf{x}^1, \mathbf{x}^2 \in X$. *If* $f_q(\mathbf{x}^1) \leq f_q(\mathbf{x}^2)$ *and* $\mathbf{y}^2 \in C_{Q,\mathbf{q},\mathbf{y}^r}$, *then* $\mathbf{y}^1 \not\geq \mathbf{y}^2$.

Proof. Assume the opposite, that $\mathbf{y}^1 \geq \mathbf{y}^2$. Since $\mathbf{y}^2 \in C_{Q,\mathbf{q},\mathbf{y}^r}$ also holds, Lemma 1, applied with the roles of $\mathbf{y}^1$ and $\mathbf{y}^2$ reversed, implies that $f_q(\mathbf{x}^1) > f_q(\mathbf{x}^2)$. This contradicts the hypothesis that $f_q(\mathbf{x}^1) \leq f_q(\mathbf{x}^2)$, and so the lemma is established. □

Using Lemma 2, Proposition 1 provides a sufficient condition for a locally optimal solution $\mathbf{x}^*$ for problem (2.9) to be also locally efficient for MOP (2.1).

Proposition 1. *Let the objective function* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$ *in MOP* (2.1) *be continuous on* X, *and let the objective function* $f_q : \mathbb{R}^n \to \mathbb{R}$ *in problem* (2.9) *be computed as in definition* (2.10) *with a positive semidefinite matrix* Q. *If* $\mathbf{x}^* \in X$ *is a local optimal solution for problem* (2.9) *and* $\mathbf{f}(\mathbf{x}^*) \in C_{Q,\mathbf{q},\mathbf{y}^r}$, *then* $\mathbf{x}^*$ *is locally efficient for* (2.1).

Proof. By the continuity of $\mathbf{f}$ and the local optimality at $\mathbf{x}^*$ for problem (2.9), there exists a nonempty open neighborhood $\mathcal{N}(\mathbf{x}^*)$ of $\mathbf{x}^*$ such that the set

$$\mathbf{f}(\mathcal{N}(\mathbf{x}^*)) := \{\mathbf{y} \in \mathbb{R}^p : \mathbf{y} = \mathbf{f}(\mathbf{x}) \text{ for some } \mathbf{x} \in \mathcal{N}(\mathbf{x}^*)\}$$

is contained in $C_{Q,\mathbf{q},\mathbf{y}^r}$ and $f_q(\mathbf{x}^*) \leq f_q(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*) \cap X$. By Lemma 2, it follows that $\mathbf{f}(\mathbf{x}) \not\leq \mathbf{f}(\mathbf{x}^*)$ for all $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*) \cap X$ and so $\mathbf{x}^*$ is locally efficient for MOP (2.1). □

Letting $C_{Q,\mathbf{q},\mathbf{y}^r}$ contain the set $\mathbf{f}(X)$ yields the following result on the computation of globally efficient solutions for (2.1).

Proposition 2. *Let the objective function* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^p$ *in MOP* (2.1) *be continuous on* X, *and let the objective function* $f_q : \mathbb{R}^n \to \mathbb{R}$ *in problem* (2.9) *be computed as in definition* (2.10) *with a positive semidefinite matrix* Q. *If* $\mathbf{x}^* \in X$ *is an optimal solution for problem* (2.9) *and* $\mathbf{f}(X) \subset C_{Q,\mathbf{q},\mathbf{y}^r}$, *then* $\mathbf{x}^*$ *is efficient for* (2.1).

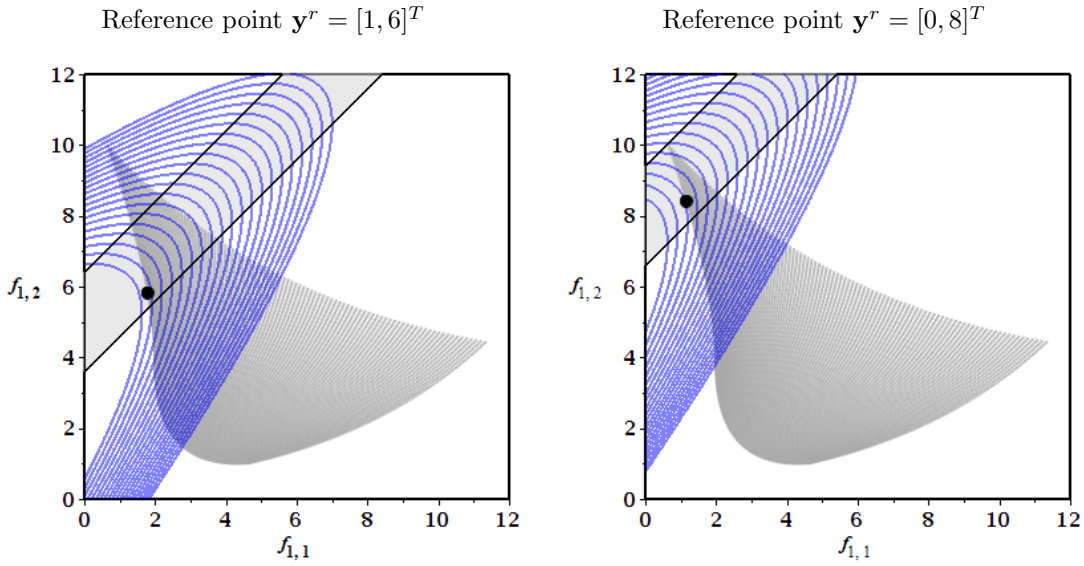

Figure 2.2: Applying the quadratic scalarization method to biobjective problem (2.70). Each Pareto optimal point, depicted with a black dot, lies within the set $C_{Q,\mathbf{q},\mathbf{y}^r}$ depicted with the light-gray shaded region. ($Q = (1/2)[-1, 1]^T[-1, 1]$, $\mathbf{q} = (\sqrt{2}/2)[1, 1]^T$.)

Proof. If $\mathbf{x}^*$ is not efficient for (2.1), then there exists $\mathbf{x} \in X$ for which $\mathbf{f}(\mathbf{x}) \leq \mathbf{f}(\mathbf{x}^*)$. Furthermore, $\mathbf{f}(\mathbf{x}) \in C_{Q,\mathbf{q},\mathbf{y}^r}$ for all $\mathbf{x} \in X$. Therefore, it follows from Lemma 1 that $f_q(\mathbf{x}) < f_q(\mathbf{x}^*)$, contradicting the hypothesis that $\mathbf{x}^*$ is optimal for problem (2.9). □

The plots of Fig. 2.2 illustrate the use of solving problems of the form (2.9) for computing efficient points for the nonconvex biobjective problem (2.70) that is given in Section 2.6.

2.2.2 Quadratic scalarization and the weighted-Chebyshev method

Given a reference point $\mathbf{z} \in \mathbb{R}^p$ and a weight $\mathbf{w} \geq \mathbf{0}$, the weighted-Chebyshev method for computing efficient points for (2.1) is given by

$$\min_{\mathbf{x} \in X,\, \mathbf{h}(\mathbf{x}) = \mathbf{0}} \quad \max_{1 \leq i \leq p} \{w_i \left(f_i(\mathbf{x}) - z_i\right)\}. \tag{2.14}$$

Problem (2.14) may be reformulated by

$$\begin{aligned} \min_{\substack{\mathbf{x}\in X,\, \mathbf{h}(\mathbf{x})=\mathbf{0},\\ \alpha\in\mathbb{R}}} \quad & \alpha \\ \text{s.t.} \quad & w_i \left(f_i(\mathbf{x}) - z_i\right) \leq \alpha, \quad i = 1, \ldots, p, \end{aligned} \tag{2.15}$$

or equivalently

$$\begin{aligned} \min_{\substack{\mathbf{x}\in X,\, \mathbf{h}(\mathbf{x})=\mathbf{0},\\ \mathbf{s}\geqq\mathbf{0},\, \alpha\in\mathbb{R}}} \quad & \alpha \\ \text{s.t.} \quad & w_i \left(f_i(\mathbf{x}) - z_i\right) + s_i = \alpha, \quad i = 1, \ldots, p, \end{aligned} \tag{2.16}$$

where $\mathbf{s} \in \mathbb{R}^p_{\geqq}$ is a slack variable. Substituting α, problem (2.16) becomes

$$\begin{aligned} \min_{\substack{\mathbf{x}\in X,\, \mathbf{h}(\mathbf{x})=\mathbf{0},\\ \mathbf{s}\geqq\mathbf{0}}} \quad & \sum_{i=1}^{p} w_i \left(f_i(\mathbf{x}) - z_i\right) + s_i \\ \text{s.t.} \quad & w_i \left(f_i(\mathbf{x}) - z_i\right) + s_i = w_{i+1} \left(f_{i+1}(\mathbf{x}) - z_{i+1}\right) + s_{i+1}, \quad i = 1, \ldots, p-1, \\ & s_i \geq 0, \quad i = 1, \ldots, p. \end{aligned} \tag{2.17}$$

Problem (2.17) may furthermore be stated in matrix notation as follows:

$$\begin{aligned} \min_{\substack{\mathbf{x}\in X,\, \mathbf{h}(\mathbf{x})=\mathbf{0},\\ \mathbf{s}\geqq\mathbf{0}}} \quad & \mathbf{1}^T \left[D_{\mathbf{w}} \left(\mathbf{f}(\mathbf{x}) - \mathbf{z}\right) + \mathbf{s}\right] \\ \text{s.t.} \quad & U^T \left[D_{\mathbf{w}} \left(\mathbf{f}(\mathbf{x}) - \mathbf{z}\right) + \mathbf{s}\right] = \mathbf{0}, \end{aligned} \tag{2.18}$$

where $\mathbf{1}$ is the length-p column vector of ones, $D_{\mathbf{w}} := \operatorname{diag}\{w_1, \ldots, w_p\}$ is a diagonal $p \times p$ matrix with entries $[D_{\mathbf{w}}]_{i,i} = w_i$, U is a $p \times p-1$ matrix whose columns are 1) of unit length with respect to the Euclidean norm, 2) orthogonal to $\mathbf{1}$, and 3) also pairwise orthogonal to one another. (Such a matrix U may be constructed as a submatrix of an orthonormal matrix whose first column is $\mathbf{1}$. U^T projects any vector in $\mathbb{R}^p$ onto the subspace of vectors that are orthogonal to $\mathbf{1}$.)

Applying the augmented Lagrangian relaxation to the equality constraint of (2.18), we have

$$\min_{\mathbf{s}\geqq\mathbf{0},\, \mathbf{x}\in X,\, \mathbf{h}(\mathbf{x})=\mathbf{0}} \quad L((\mathbf{x}, \mathbf{s}), (\mathbf{v}, a)), \tag{2.19}$$

where

$$L((\mathbf{x},\mathbf{s}),(\mathbf{v},a)) := [D_{\mathbf{w}}(\mathbf{f}(\mathbf{x})-\mathbf{z})+\mathbf{s}]^T(\mathbf{1}+U\mathbf{v}) + \frac{a}{2}\left\|[D_{\mathbf{w}}(\mathbf{f}(\mathbf{x})-\mathbf{z})+\mathbf{s}]^T U\right\|_2^2, \tag{2.20}$$

$a > 0$ is a penalty weight, and $\mathbf{v} \in \mathbb{R}^{p-1}$ is the multiplier associated with the constraint

$$U^T[D_{\mathbf{w}}(\mathbf{f}(\mathbf{x})-\mathbf{z})+\mathbf{s}] = \mathbf{0}. \tag{2.21}$$

The role of problem (2.19) as a relaxation of problem (2.18) implies that, in general, solutions $(\mathbf{x}^*,\mathbf{s}^*)$ for problem (2.19) are not feasible for (2.18). If $\mathbf{w} > 0$, then the matrix $D_{\mathbf{w}}$ in (2.20) is invertible, and the objective (2.20) has the form (2.10) with

$$Q = aD_{\mathbf{w}}U(D_{\mathbf{w}}U)^T, \quad \mathbf{q} = D_{\mathbf{w}}(\mathbf{1}+U\mathbf{v}), \quad \text{and } \mathbf{y}^r = \mathbf{z} - D_{\mathbf{w}}^{-1}\mathbf{s}^* \tag{2.22}$$

Thus, if $(\mathbf{x}^*,\mathbf{s}^*)$ is an optimal solution for problem (2.19) with $\mathbf{w} > 0$, $\mathbf{f}(\mathbf{x}^*) \in C_{Q,\mathbf{q},\mathbf{y}^r}$, and Q, $\mathbf{q}$, and $\mathbf{y}^r$ as defined in (2.22), then $\mathbf{x}^*$ is (locally) efficient for the MOP (2.1) by Proposition 1, even if $(\mathbf{x}^*,\mathbf{s}^*)$ is not feasible for problem (2.18).

In order to strengthen the tie between solutions $(\mathbf{x}^*,\mathbf{s}^*)$ generated for problem (2.19) and the solutions $\mathbf{x}^*$ generated with the weighted-Chebyshev method (2.14), the role of the augmented Lagrangian parameters $\mathbf{v}$ and a is now examined more closely. Motivated by the theory of duality given in [112, 97], we use the augmented Lagrangian problem (2.19) to formulate the dual problem to problem (2.18)

$$\max_{a>0,\mathbf{v}\in\mathbb{R}^{p-1}} \min_{\substack{\mathbf{x}\in X,\,\mathbf{h}(\mathbf{x})=\mathbf{0},\\ \mathbf{s}\geqq\mathbf{0}}} L((\mathbf{x},\mathbf{s}),(\mathbf{v},a)) \tag{2.23}$$

and describe two conditions [97] under which there is no duality gap between the optimal value of the dual problem (2.23) and the optimal value of the primal problem (2.18) (that is, L has a saddle point in the primal variables $(\mathbf{x},\mathbf{s})$ and in the dual variables $(\mathbf{v},a)$):

1. The primal problem (2.18) meets the *quadratic growth condition*: that is, the dual problem (2.23) is feasible, i.e., there exists $a > 0$, $\mathbf{v} \in \mathbb{R}^{p-1}$ such that problem (2.19) is bounded from below.

2. The primal problem (2.18) is *stable of degree 2*; that is, in addition to meeting the quadratic

growth condition, it also meets the second order sufficiency conditions for optimality [6].

Proposition 3. *Let problem* (2.18) *meet the quadratic growth condition and the second order sufficiency conditions for optimality, and let* $(\mathbf{x}^*, \mathbf{s}^*)$ *be an optimal solution for problem* (2.18). *Then there exist a multiplier* $\mathbf{v} \in \mathbb{R}^{p-1}$ *and a penalty coefficient* $a > 0$ *for which* $(\mathbf{x}^*, \mathbf{s}^*)$ *is also an optimal solution for problem* (2.19).

Proof. Due to the quadratic growth condition and the second order sufficiency conditions for optimality, there is no duality gap between the primal problem (2.18) and the dual problem (2.23). Letting $\mathbf{v}$ and a of problem (2.19) be set to the dual optimal values $(\mathbf{v}^*, a^*)$ for the dual problem (2.23), the proposition is proven. □

From Proposition 3, a necessary condition is stated for an efficient point $\mathbf{x}^*$ for MOP (2.1) generated with the weighted-Chebyshev method (2.14), $\mathbf{w} > 0$, to be computable as an optimal solution for an SOP reformulation (2.9) obtained with quadratic scalarization.

Corollary 1. *Let* $\mathbf{x}^*$ *be a weakly efficient solution for* (2.1). *Then, for any fixed utopia point* $\mathbf{z}^U$, *there exists a weight vector* $\mathbf{w}^* > 0$ *for which there is an optimal solution* $(\mathbf{x}^*, \mathbf{s}^*)$ *for problem* (2.18) *with* $\mathbf{w} = \mathbf{w}^*$ *and* $\mathbf{z} = \mathbf{z}^U$. *Furthermore, if problem* (2.18) *with* $\mathbf{w} = \mathbf{w}^*$ *and* $\mathbf{z} = \mathbf{z}^U$ *meets the quadratic growth condition and the second order sufficiency conditions for optimality, then* $\mathbf{x}^*$ *is also an optimal solution for an SOP reformulation* (2.9).

Proof. The existence of $\mathbf{w}^* > 0$ for any fixed utopia point $\mathbf{z}^U$ for which there exists an optimal solution $\mathbf{x}^*$ for problem (2.14) with $\mathbf{w} = \mathbf{w}^*$ and $\mathbf{z} = \mathbf{z}^U$ follows from Choo and Atkins (see Theorem 4.24 of [34]). Because (2.18) is an equivalent reformulation of problem (2.14), then there is an optimal solution $(\mathbf{x}^*, \mathbf{s}^*)$ for problem (2.18) with $\mathbf{w} = \mathbf{w}^*$ and $\mathbf{z} = \mathbf{z}^U$, where $\mathbf{x}^*$ is the same solution that is optimal for (2.14) with $\mathbf{w} = \mathbf{w}^*$ and $\mathbf{z} = \mathbf{z}^U$. Because problem (2.18) with $\mathbf{w} = \mathbf{w}^*$ and $\mathbf{z} = \mathbf{z}^U$ meets the quadratic growth condition and the second order sufficiency conditions, then by Proposition 3, there exists dual optimal $(\mathbf{v}^*, a^*)$ such that $(\mathbf{x}^*, \mathbf{s}^*)$ is also optimal for problem (2.19) with $\mathbf{v} = \mathbf{v}^*$ and $a = a^*$. Thus, $\mathbf{x}^*$ is an optimal solution for a problem of the form (2.9) where Q, $\mathbf{q}$, and $\mathbf{y}^r$ are given by (2.22) with $\mathbf{v} = \mathbf{v}^*$ and $a = a^*$, $\mathbf{w} = \mathbf{w}^*$, $\mathbf{z} = \mathbf{z}^U$, and $\mathbf{s}^*$ is taken from the computed optimal solution $(\mathbf{x}^*, \mathbf{s}^*)$ of problem (2.18). □

In formulating the relaxed problem (2.19), one evaluates the tightness of the relaxation in terms of the gap between the optimal value (2.19) and the optimal value (2.18). (Increasing

tightness corresponds to decreasing gap.) The tightness of the relaxation depends on the multiplier $\mathbf{v}$ and the penalty coefficient $a > 0$ of problem (2.19). The distributed approach of Section 2.4 also requires the relaxation of the problem (2.4) coordination constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$, resulting in problem (2.5). In order to tighten the relaxation occurring in either problem (2.5) or problem (2.19), the multiplier $\mathbf{v}$ and the penalty coefficients $\mu > 0$ and $a > 0$ are iteratively updated using concepts from Lagrangian coordination that are analyzed in Section 2.3. Without a loss of generality, the Lagrangian coordination approach addresses the relaxation (2.5) of problem (2.4) in particular.

2.2.3 Generating a sample of weight vectors w

The need to construct finite samples of length p weight vectors $\mathbf{w}$ appears frequently in the practical application of the weighted-sum scalarization. As observed at the end of Section 2.2.2, this need also arises in the application of the quadratic scalarization as motivated by the dual of the weighted-Chebyshev method.

The goal in this section is to describe the generation of $\mathbf{w}$-samples in terms of probability spaces. From probability theory (see, e.g., [48]), a probability space has associated with it a sample space Ω, a collection $\mathcal{E}$ of subsets $K \subseteq \Omega$ called *events*, and a probability function $P : \mathcal{E} \to [0, 1]$ assigning a probability that a sampled point is contained in each event $K \in \mathcal{E}$.

The collection of events $\mathcal{E}$ satisfies certain rules such as

1. $\emptyset \in \mathcal{E}$,
2. $K \in \mathcal{E}$ implies $\Omega \backslash K \in \mathcal{E}$,
3. $\cup_{i \in I} K_i \in \mathcal{E}$ whenever $K_i \in \mathcal{E}$ for each $i \in I$. (I is a countable index set.)

The probability function $P : \mathcal{E} \to [0, 1]$ is defined so that

1. P is additive under taking disjoint unions $P(\cup_{i \in I} K_i) = \sum_{i \in I} P(K_i)$, and
2. $P(\Omega) = 1$.

Two sample spaces Ω are considered. In the continuous setting, define

$$\mathcal{W}_p := \left\{ \mathbf{w} \in \mathbb{R}^p_> \, : \, \sum_{i=1}^{p} w_i = 1 \right\},$$

where $p \in \mathbb{Z},\ p \geq 2$, is the length of each weight vector. In the discrete setting, define the sample space

$$\mathcal{W}_{N,p} := \left\{ \mathbf{w} \,:\, w_i \in \left\{ \frac{k}{N},\, k = 0, \ldots, N \right\} \text{ for } i = 1, \ldots, p, \text{ and } \sum_{i=1}^{p} w_i = 1 \right\},$$

where $p \in \mathbb{Z},\ p \geq 2$, is the length of each weight vector and $N \in \mathbb{Z},\ N \geq 1$, is a parameter specifying the refinement of the discretization. (That is, each element of $\mathcal{W}_{N,p}$ is associated with an ordered p-tuple of nonnegative integers such that the p components sum to N.)

We consider three probability spaces, where each probability space corresponds to a sampling technique.

1. Let $\Omega = \mathcal{W}_p$, and let $\mathcal{E} = 2^\Omega$ where 2^Ω is the set of all subsets of Ω. Let $\mathbf{w}$ be generated by taking p uniformly sampled real numbers from the interval $(0, 1) \subset \mathbb{R}$ and then normalizing so that $\sum_{i=1}^{p} w_i = 1$. Although the probability function P is not specified here, insight into P may be obtained from the left plot of Fig. 3 depicting a sample of weight vectors generated in this manner for $p = 3$.

2. Let $\Omega = \mathcal{W}_{N,p}$. Combinatorially, we have $|\mathcal{W}_{N,p}| = \binom{N+p-1}{p-1}$. If $\mathcal{E} = 2^\Omega$, then P may be defined in a number of ways. One definition of P is induced from the assumption that each outcome of $\mathcal{W}_{N,p}$ is equally likely to be sampled, that is, define $P(\{\mathbf{w}\}) := \frac{1}{|\mathcal{W}_{N,p}|}$ for each singleton event $\{\mathbf{w}\}$ where $\mathbf{w} \in \Omega$. This definition of P gives a uniform distribution on the probability space. The possible sample points are depicted in the center plot of Fig. 3.

3. Let $\Omega = \mathcal{W}_p$. The collection of events $\mathcal{E}$ is generated by $\emptyset$ and all subsets $S \subseteq \mathcal{W}_p$ having the following form:

   (a) Define the $p \times p$ matrix B_k by

$$B_k[i, j] := \begin{cases} \frac{1}{2} & i = j \neq k \\ 1 & i = j = k \\ \frac{1}{2} & i \neq j,\ j = k \\ 0 & \text{otherwise.} \end{cases}$$

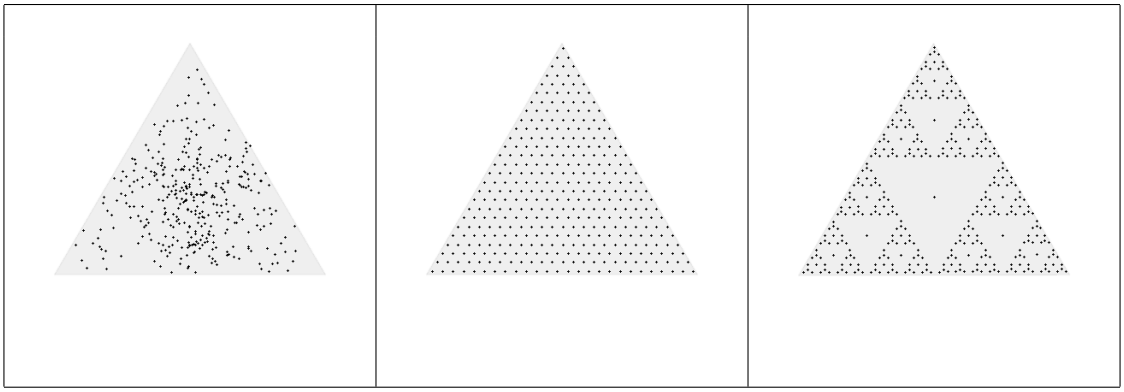

Figure 2.3: Three sampling techniques demonstrated

For example, for $p = 4$ and $k = 2$, we have

$$B_2 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}.$$

(b) For $n \in \mathbb{Z}$, $n \geq 0$, define $S := \text{Conv}(B)$, where $p \times p$ matrix $B = B_{i_1} B_{i_2} \cdots B_{i_n}$, $i_1, \ldots, i_n \in \{1, \ldots, p\}$ and $\text{Conv}(\cdot)$ is the operation of taking the convex hull of a set of vertices given by the rows of matrix B. In the case where $n = 0$, set $B = I$, the $p \times p$ identity matrix.

The probability function P is uniquely determined once the values of $P(K)$ are specified for the events of the above form $K = S$ such that the probability function requirements are not violated. The probabilities $P(K)$ for K not of the above form $K = S$ may be inferred from the requirements of the probability function P. For $p = 3$, the subsets $S \subseteq \mathcal{W}_p$ are visualized as a nested pattern resembling the Sierpinski triangle. This is evident from the right plot of Fig. 3 whose points depict the barycenters of sets having the above form S. These same points may be taken as the generated weight vectors $\mathbf{w}$.

2.3 Lagrangian coordination

In the formulation of MOP (2.1), a distinction is made between the decomposable constraint $\mathbf{x} \in X := \prod_{i=1}^{m} X_i$ and the coordination constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$. The latter includes constraints that are necessary for enforcing consistency between different subproblem solutions in the presence of inter-subproblem coupling; other constraints included in $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ result from the SOP reformulation (2.18). Any distributed solution approach is based on a decomposition $X := \prod_{i=1}^{m} X_i$; any other constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ must be relaxed under decomposition. In this section, an iterative Lagrangian coordination technique is analyzed for solving the scalarized problem (2.4) by relaxing the constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ and incorporating it into the objective function as augmented Lagrange terms. Due to the intended use of iterative, distributed computation techniques, the Lagrangian coordination technique is analyzed under the assumption that solution $\mathbf{x}$ updates are computed inexactly.

By setting $f := f_s$, problem (2.4) is rewritten slightly for notational simplicity in the following manner:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{x} \in X. \end{aligned} \tag{2.24}$$

Assuming that X is a convex set and that f and $\mathbf{h}$ are continuously differentiable, we examine iterative approaches to generating a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ of approximations to $(\mathbf{x}^*, \mathbf{v}^*)$, where $(\mathbf{x}^*, \mathbf{v}^*)$ satisfies the necessary conditions of optimality for problem (2.24) given by

$$\left[\nabla f(\mathbf{x}^*) + (\mathbf{v}^*)^T \nabla \mathbf{h}(\mathbf{x}^*)\right](\mathbf{x} - \mathbf{x}^*) \geq 0 \quad \text{for all } \mathbf{x} \in X \tag{2.25}$$

$$\mathbf{h}(\mathbf{x}^*) = \mathbf{0}. \tag{2.26}$$

After taking the augmented Lagrangian relaxation of problem (2.24) given by

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) + \mathbf{v}^T \mathbf{h}(\mathbf{x}) + \frac{\mu}{2} \left\|\mathbf{h}(\mathbf{x})\right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x} \in X, \end{aligned} \tag{2.27}$$

the following stationary point condition for problem (2.27) satisfied by $\widetilde{\mathbf{x}} \in X$ may be stated:

$$\left[\nabla f(\widetilde{\mathbf{x}}) + (\mathbf{v} + \mu \mathbf{h}(\widetilde{\mathbf{x}}))^T \nabla \mathbf{h}(\widetilde{\mathbf{x}})\right](\mathbf{x} - \widetilde{\mathbf{x}}) \geq 0 \quad \text{for all } \mathbf{x} \in X, \tag{2.28}$$

where $\mathbf{v}$ and μ are treated as fixed parameters.

The following proposition proven in Li et al. [120] states conditions on the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ under which a critical pair $(\mathbf{x}^*, \mathbf{v}^*)$ may be computed; the pair $(\mathbf{x}^*, \mathbf{v}^*)$ is critical in the sense that it satisfies the necessary conditions of optimality (2.25) and (2.26) for problem (2.24).

Proposition 4. *For problem* (2.24)*, let the objective function* f *and constraint function* $\mathbf{h}$ *be continuously differentiable, and the set* $X \subseteq \mathbb{R}^n$ *be closed and convex. Given starting values* $\mathbf{v}^1$ *and* μ^1*, let a sequence* $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$ *be generated satisfying the following assumptions for each* $k \geq 1$*:*

1. *each* $\widetilde{\mathbf{x}}^k$ *is a global minimum for problem* (2.27) *with* $\mathbf{v} = \mathbf{v}^k$ *and* $\mu = \mu^k$*;*
2. *each multiplier* $\mathbf{v}^k$ *is determined* a priori *and the sequence* $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ *is bounded;*
3. *each penalty coefficient* μ^k *satisfies* $0 < \mu^k < \mu^{k+1}$*; furthermore,* $\lim_{k\to\infty} \mu^k = \infty$*;*
4. *each* $\widetilde{\mathbf{v}}^k$ *is computed to satisfy*

$$\widetilde{\mathbf{v}}^k = \mathbf{v}^k + \mu^k \mathbf{h}(\widetilde{\mathbf{x}}^k). \tag{2.29}$$

If the sequence $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$ *has a limit point* $(\mathbf{x}^*, \mathbf{v}^*)$*, then* $(\mathbf{x}^*, \mathbf{v}^*)$ *satisfies the necessary conditions of optimality* (2.25) *and* (2.26) *for problem* (2.24)*. Furthermore,* $\mathbf{x}^*$ *is a globally optimal solution for problem* (2.24)*.*

When the third assumption of Proposition 4 that each $\widetilde{\mathbf{x}}^k$, $k \geq 1$, is a global minimum for problem (2.27) with $\mathbf{v} = \mathbf{v}^k$ and $\mu = \mu^k$ is weakened so that $\widetilde{\mathbf{x}}^k$ only satisfies the stationary point condition (2.28), the following extension of Proposition 4 is stated and proven below in Proposition 5.

Proposition 5. *For problem* (2.24)*, let the objective function* f *and constraint function* $\mathbf{h}$ *be continuously differentiable, and the set* $X \subseteq \mathbb{R}^n$ *be closed and convex. Given starting values* $\mathbf{v}^1$ *and* μ^1*, let a sequence* $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$ *be generated satisfying the following assumptions for each* $k \geq 1$*:*

1. *each* $\widetilde{\mathbf{x}}^k$ *satisfies the stationary point condition* (2.28) *for problem* (2.27) *with* $\mathbf{v} = \mathbf{v}^k$ *and* $\mu = \mu^k$*;*

2. *each multiplier* $\mathbf{v}^k$ *is determined* a priori *and the sequence* $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ *is bounded;*

3. *each penalty coefficient* μ^k *satisfies* $0 < \mu^k < \mu^{k+1}$*, and* $\lim_{k\to\infty} \mu^k = \infty$*;*

4. *each multiplier* $\widetilde{\mathbf{v}}^k$ *is computed by equation* (2.29)*.*

If the sequence $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$ *has a limit point* $(\mathbf{x}^*, \mathbf{v}^*)$*, then* $(\mathbf{x}^*, \mathbf{v}^*)$ *satisfies the necessary conditions of optimality* (2.25) *and* (2.26) *for problem* (2.24)*.*

Proof. Due to (2.29), condition (2.28) may be rewritten for each $k \geq 1$ as

$$\left[\nabla f(\widetilde{\mathbf{x}}^k) + \left(\widetilde{\mathbf{v}}^k\right)^T \nabla \mathbf{h}(\widetilde{\mathbf{x}}^k)\right]\left(\mathbf{x} - \widetilde{\mathbf{x}}^k\right) \geq 0 \quad \text{for all } \mathbf{x} \in X.$$

Let $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k\in K}$ be a subsequence converging to $(\mathbf{x}^*, \mathbf{v}^*)$. The continuous differentiability of f and $\mathbf{h}$ imply that

$$\left[\nabla f(\mathbf{x}^*) + (\mathbf{v}^*)^T \nabla \mathbf{h}(\mathbf{x}^*)\right](\mathbf{x} - \mathbf{x}^*) \geq 0 \quad \text{for all } \mathbf{x} \in X$$

also holds. By the assumptions that sequence $\left\{\widetilde{\mathbf{v}}^k\right\}_{k=1}^{\infty}$ has a limit point $\mathbf{v}^*$, sequence $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ is bounded, and $\lim_{k\to\infty} \mu^k = \infty$, it follows that $\mathbf{h}(\mathbf{x}^*) = \mathbf{0}$. □

Propositions 4 and 5 suggest the iterative generation (as opposed to *a priori* generation) of the multiplier sequence $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ given by the following method of multipliers [56, 93, 10, 11] update rule satisfying

$$\mathbf{v}^{k+1} = \mathbf{v}^k + \mu^k \mathbf{h}(\widetilde{\mathbf{x}}^k) \quad k \geq 1. \tag{2.30}$$

In practical applications, update (2.30) is applied with the substitution $\widetilde{\mathbf{x}}^k = \mathbf{x}^k$ resulting in updates satisfying

$$\mathbf{v}^{k+1} = \mathbf{v}^k + \mu^k \mathbf{h}(\mathbf{x}^k) \quad k \geq 1 \tag{2.31}$$

that are motivated by the approximate computations $\mathbf{x}^k$ of $\widetilde{\mathbf{x}}^k$. In order to gain insight into the convergence of the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ satisfying (2.31), Proposition 6 is proven in the more general context where $\mathbf{x}^k$ is an approximation of $\widetilde{\mathbf{x}}^k$ for each $k \geq 1$, but $\widetilde{\mathbf{x}}^k$ itself has no particular meaning in the context of problem (2.27).

Proposition 6. *Let* $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^q$*, and* $\mu^k \in \mathbb{R}$*,* $k \geq 1$*. If* $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ *is a sequence satisfying* (2.31) *and* $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ *converges, while* $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$ *is a sequence satisfying* (2.29) *and*

$$0 \leq \left\|\mathbf{x}^k - \widetilde{\mathbf{x}}^k\right\|_2 < \epsilon_k \, , k \geq 1, \quad \lim_{k\to\infty} \epsilon_k = 0, \quad \lim_{k\to\infty} \mu^k \left\|\mathbf{h}(\mathbf{x}^k) - \mathbf{h}(\widetilde{\mathbf{x}}^k)\right\|_2 = 0, \tag{2.32}$$

then any limit point $(\mathbf{x}^*, \mathbf{v}^*)$ *of the sequence* $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ *is also a limit point of the sequence* $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$.

Proof. Let $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k\in K}$ be a subsequence converging to $(\mathbf{x}^*, \mathbf{v}^*)$. By assumption (2.32), $\left\|\mathbf{x}^k - \widetilde{\mathbf{x}}^k\right\|_2 < \epsilon_k$ for each $k \geq 1$, and $\epsilon_k \to 0$, and so $\left\{\widetilde{\mathbf{x}}^k\right\}_{k\in K}$ converges to $\mathbf{x}^*$. Also, combining (2.31) and (2.29) yields

$$\left\|\mathbf{v}^{k+1} - \widetilde{\mathbf{v}}^k\right\|_2 = \mu^k \left\|\mathbf{h}(\mathbf{x}^k) - \mathbf{h}(\widetilde{\mathbf{x}}^k)\right\|_2 \quad k \geq 1.$$

Since $\left\{\mu^k \left\|\mathbf{h}(\mathbf{x}^k) - \mathbf{h}(\widetilde{\mathbf{x}}^k)\right\|_2\right\}_{k\in K}$ converges to zero by assumption (2.32), and since $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ converges, it also follows that $\left\{\mathbf{v}^k - \widetilde{\mathbf{v}}^k\right\}_{k\in K}$ converges to zero and so $\left\{\widetilde{\mathbf{v}}^k\right\}_{k\in K}$ converges to $\mathbf{v}^*$. Thus, the limit point $(\mathbf{x}^*, \mathbf{v}^*)$ of the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ is also a limit point of the sequence $\left\{(\widetilde{\mathbf{x}}^k, \widetilde{\mathbf{v}}^k)\right\}_{k=1}^{\infty}$. □

The assumption $\lim_{k\to\infty} \mu^k \left\|\mathbf{h}(\mathbf{x}^k) - \mathbf{h}(\widetilde{\mathbf{x}}^k)\right\|_2 = 0$ of Proposition 6 suggests that the rate-of-increase of μ^k needs to be restrained, while the inexactness tolerance ϵ_k needs to vanish at a sufficiently fast rate.

Propositions 4, 5, and 6 imply the convergence result stated in Corollary 2 addressing the generation of $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ using update (2.31) corresponding to inexact computations of $\mathbf{x}^k$, $k \geq 1$.

Corollary 2. *For problem* (2.24)*, let the objective function* f *and the constraint function* $\mathbf{h}$ *be continuously differentiable, and the set* $X \subseteq \mathbb{R}^n$ *be closed and convex. Given starting values* $\mathbf{v}^1$ *and* $\mu^1 > 0$*, let a sequence* $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ *be generated satisfying* (2.31)*,* $\left\{\mathbf{v}^k\right\}_{k=1}^{\infty}$ *be convergent, and* $\lim_{k\to\infty} \mu^k = \infty$*, where* $\mu^k > 0$*,* $k \geq 1$*. If* $\left\{\widetilde{\mathbf{x}}^k\right\}_{k=1}^{\infty}$ *is a sequence whose elements* $\widetilde{\mathbf{x}}^k$*,* $k \geq 1$*, each satisfy the stationary point condition* (2.28) *with* $\mathbf{v} = \mathbf{v}^k$ *and* $\mu = \mu^k$*, and* (2.32) *holds, then any limit point* $(\mathbf{x}^*, \mathbf{v}^*)$ *of the sequence* $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ *satisfies the necessary conditions for optimality* (2.25) *and* (2.26) *for problem* (2.24)*. Furthermore, if each* $\widetilde{\mathbf{x}}^k$*,* $k \geq 1$ *is a globally optimal solution for problem* (2.27) *with* $\mathbf{v} = \mathbf{v}^k$ *and* $\mu = \mu^k$*, then* $\mathbf{x}^*$ *is also a globally optimal solution for problem* (2.24)*.*

The generation of the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$, where $\mathbf{x}^k$ is optimal for each problem (2.27), $k \geq 1$, with $\mathbf{v} = \mathbf{v}^k$, $\mu = \mu^k$, $\mathbf{v}^{k+1} = \mathbf{v}^k + \mu^k \mathbf{h}(\mathbf{x}^k)$, and $\lim_{k\to\infty} \mu^k = \infty$, corresponds to the well-known method of multipliers. The textbooks of Bertsekas [10, 11] provide convergence analyses of various strengths corresponding to different sets of assumptions on f, $\mathbf{h}$, X, and on the accuracy of updates $\mathbf{x}^k$, $k \geq 1$. For example, under the assumption of twice continuous differentiability on f, $\mathbf{h}$, and the assumption $X \subseteq \mathbb{R}^n$, proofs are provided for a linear rate of convergence to a local minimum–Lagrange multiplier pair $(\mathbf{x}^*, \mathbf{v}^*)$ for the sequence generated by the method of multipliers. The conditions for the linear rate of convergence are proven most rigorously under the assumptions that f, $\mathbf{h}$ are twice continuously differentiable, $X = \mathbb{R}^n$, and updates $\mathbf{x}^k$, $k \geq 1$ are exact. Corollary 2 provides an extension of these results with a study of convergence due to inexact updates $\mathbf{x}^k$, $k \geq 1$, where $\mathbf{x}^k \in X \subseteq \mathbb{R}^n$. Each update $\mathbf{x}^k$, $k \geq 1$, approximates either a globally optimal solution for problem (2.27) with $\mathbf{v} = \mathbf{v}^k$ and $\mu = \mu^k$, or more generally, $\mathbf{x}^k$ approximates a point satisfying the stationary point condition (2.28) for the same problem.

In this section, we make no assumption on the convexity of the objective function of problem (2.27), and so the convergence analysis for iterative solution approaches for solving problem (2.27) focuses on the convergence of the iteration sequences $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ to solutions $\overline{\mathbf{x}}$, where each $\mathbf{x}^k$, $k \geq 1$, (approximately) satisfies the stationary point condition (2.28). This continues to be the case in Section 2.4, where a distributed solution approach is applied for computing approximations of solutions satisfying the stationary point condition (2.28) for problem (2.27).

2.4 Distributed computation using the block coordinate descent (BCD) method

Once a solution technique is established for solving the SOP reformulation (2.4) using references to the relaxed problem (2.5), then (2.5) may be solved in a distributed manner. The decomposability of problem (2.5) is stated through (2.6). The goal of this section is to study the convergence of an existing Gauss-Seidel approach, known as the block coordinate descent (BCD) method, for the distributed computation of optimal solutions for problem (2.6); the convergence analysis extends existing analyses addressing certain generalized convexity assumptions that are found in [49, 114]. Given a decomposable SOP of the form (2.6) and a starting solution $\mathbf{x}^0$, BCD generates a sequence

Algorithm 1 The BCD method

function BCD(f, $X_1 \times \cdots \times X_m$, $\mathcal{I}$, $\mathbf{x}^0$)
  $\ddot{\mathbf{x}} \leftarrow \mathbf{x}^0$
  for $k = 1, 2, \ldots$ **do**
    $i \leftarrow \mathcal{I}_k, \quad \ddot{\mathbf{x}}_i \leftarrow \underset{\mathbf{x}_i \in X_i}{\operatorname{argmin}} f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}), \quad \mathbf{x}^k \leftarrow \ddot{\mathbf{x}}$
  end for
  return $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$
end function

$\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ by applying the following updates for $k \geq 1$ satisfying:

$$i = \mathcal{I}(k), \quad \mathbf{x}_i^k \in \underset{\mathbf{x}_i \in X_i}{\operatorname{argmin}} f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}^{k-1}), \quad \mathbf{x}_{\neg i}^k = \mathbf{x}_{\neg i}^{k-1}, \tag{2.33}$$

where $f(\,\cdot\,, \ddot{\mathbf{x}}_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}$ is a function parameterized by a fixed $\ddot{\mathbf{x}}_{\neg i}$ and $\mathcal{I}$ is a sequence whose elements take on index values $i = 1, \ldots, m$ with the k^{th} element referenced by $\mathcal{I}(k)$. Thus, each element $\mathbf{x}^k$, $k \geq 1$, corresponds to an update of a single block coordinate $\mathbf{x}_i$, $i = \mathcal{I}(k)$, obtained as an optimal solution of the subproblem

$$\min_{\mathbf{x}_i \in X_i} \quad f(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}^{k-1}). \tag{2.34}$$

In this section, $\mathcal{I}$ is assumed to give the *fixed cyclic order* of indices given by

$$\mathcal{I} := \{1, 2, \ldots, m, 1, 2, \ldots\}. \tag{2.35}$$

That is, $i := \mathcal{I}(k)$ satisfies $k - 1 \equiv i - 1 \mod m$. Generalizations of the index sequence (2.35) are studied in Tseng [114]. The BCD method as described above is stated in Algorithm 1.

The presence of limit points $\overline{\mathbf{x}}$ in the BCD sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$, and their properties in the context of problem (2.6) are well studied [9, 49, 114]. Under the assumption that f is a differentiable convex function, the optimality of $\overline{\mathbf{x}}$ for problem (2.6) is examined (see, for example, [9]), while under more generalized convexity assumptions, the stationary point condition

$$\nabla_{\mathbf{x}} f(\overline{\mathbf{x}})\,(\mathbf{x} - \overline{\mathbf{x}}) \geq 0 \quad \text{for all } \mathbf{x} \in X \tag{2.36}$$

corresponding to necessary conditions of optimality over convex set X is examined (see, for example, [49]).

In Proposition 7, convergence results similar to those given in [49, 114] are stated and proven in terms of the limiting behavior of vicinities within a sequence, rather than in terms of the pointwise limiting behavior within a sequence. Given a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ and a subsequence $\left\{\mathbf{x}^k\right\}_{k\in K}$, a *vicinity sequence* that is *based* on the subsequence $\left\{\mathbf{x}^k\right\}_{k\in K}$ *within* the sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ is a sequence

$$\left\{\left(\mathbf{x}^{k-t_l},\ldots,\mathbf{x}^{k-1},\mathbf{x}^k,\mathbf{x}^{k+1},\ldots,\mathbf{x}^{k+t_u}\right)\right\}_{k\in K}$$

of finite-tuples, i.e., *vicinities*, of the form

$$\left(\mathbf{x}^{k-t_l},\ldots,\mathbf{x}^{k-1},\mathbf{x}^k,\mathbf{x}^{k+1},\ldots,\mathbf{x}^{k+t_u}\right),\quad k\in K,$$

where $t_l, t_u \in \mathbb{Z}_{\geq}$ and $k - t_l \geq 1$. Each vicinity $\left(\mathbf{x}^{k-t_l},\ldots,\mathbf{x}^{k-1},\mathbf{x}^k,\mathbf{x}^{k+1},\ldots,\mathbf{x}^{k+t_u}\right)$, $k \in K$, includes, i.e., is *based* on some element $\mathbf{x}^k$, $k \in K$, within the subsequence $\left\{\mathbf{x}^k\right\}_{k\in K}$, and consists of consecutive neighbors *within* the original sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$.

Using the concept of the vicinity sequence, the convergence of the sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ generated by BCD applied to (2.6) with a starting point $\mathbf{x}^0 \in X$ is examined under the following *BCD Problem Assumptions* and *BCD Sequence Assumptions*:

BCD Problem Assumptions:

1. The objective function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable;
2. The local constraint sets X_i, $i = 1,\ldots,m$, are closed and convex;
3. Given an initial point $\mathbf{x}^0 \in X$, the level set X^0 defined by

$$X^0 := \left\{\mathbf{x} \in X \,:\, f(\mathbf{x}) \leq f(\mathbf{x}^0)\right\} \tag{2.37}$$

   is compact;

BCD Sequence Assumptions:

1. The block index sequence $\mathcal{I}$ is defined as in (2.35);
2. The sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ satisfies (2.33) for all $k \geq 1$.

The BCD Problem Assumption 3 and the BCD Sequence Assumption 1 in particular allow for the extraction of the following vicinity subsequence from $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$.

Subsequence Extractions:

1. $\left\{\mathbf{x}^k\right\}_{k\in K_1}$, $K_1 \subseteq \{1,2,\ldots\}$, converges to $\overline{\mathbf{x}}$, a limit point of $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$;

2. $\left\{\mathbf{x}^k\right\}_{k\in K_2}$, $K_2 \subseteq K_1$, is such that each element $\mathbf{x}^k$, $k \in K_2$, is the result of a specific block coordinate $\mathbf{x}_t$ update for some fixed $t \in \{1,\ldots,m\}$;

3. $\left\{\mathbf{x}^k\right\}_{k\in K_3}$, $K_3 \subseteq K_2$, is such that a vicinity sequence

$$\left\{\left(\mathbf{x}^{k-t+1},\ldots,\mathbf{x}^k,\ldots,\mathbf{x}^{k-t+m}\right)\right\}_{k\in K_3} \tag{2.38}$$

   based on $\left\{\mathbf{x}^k\right\}_{k\in K_3}$ within $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ converges to a limit vicinity $\left(\overline{\mathbf{x}}^1,\overline{\mathbf{x}}^2,\ldots,\overline{\mathbf{x}}^m\right)$ for some $t \in \{1,\ldots,m\}$.

A limit vicinity $\left(\overline{\mathbf{x}}^1,\overline{\mathbf{x}}^2,\ldots,\overline{\mathbf{x}}^m\right)$ of a vicinity sequence (2.38) extracted from a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ is referred to as a limit vicinity $\left(\overline{\mathbf{x}}^1,\overline{\mathbf{x}}^2,\ldots,\overline{\mathbf{x}}^m\right)$ *of* $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ for brevity.

The vicinity sequence terminology clarifies the connection between the optimality conditions for the subproblems (2.34) that are directly enforced on the BCD-computed solutions $\mathbf{x}^k$, $k \geq 1$, and any satisfaction of optimality conditions for the underlying AiO problem (2.6) that is indirectly-enforced on limit points $\overline{\mathbf{x}}$ of $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$. Consequently, this terminology results in an easier statement of BCD convergence proofs and additional insight into BCD convergence.

The following lemma gives preliminary results concerning limit vicinities $\left(\overline{\mathbf{x}}^1,\overline{\mathbf{x}}^2,\ldots,\overline{\mathbf{x}}^m\right)$ of a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ generated with BCD.

Lemma 3. *Given $\mathbf{x}^0 \in X$, let a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ be generated with BCD applied to problem (2.6). If, given the same $\mathbf{x}^0 \in X$, problem (2.6) satisfies the BCD Problem Assumptions and the sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ satisfies the BCD Sequence Assumptions, then the following claims hold for any limit vicinity $\left(\overline{\mathbf{x}}^1,\overline{\mathbf{x}}^2,\ldots,\overline{\mathbf{x}}^m\right)$ of $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$:*

1. $f(\overline{\mathbf{x}}^1) = f(\overline{\mathbf{x}}^2) = \cdots = f(\overline{\mathbf{x}}^m)$,

2. $\overline{\mathbf{x}}_i^{\ell-1} = \overline{\mathbf{x}}_i^{\ell}$ *for* $i = 1,\ldots,m$, $\ell = 2,\ldots,m$, *and* $i \neq \ell$.

Proof. Claim 1 follows from Proposition 2 of [49]. To prove claim 2, first observe that the consecutive elements within each vicinity $\left(\mathbf{x}^{k-t+1},\ldots,\mathbf{x}^k,\ldots,\mathbf{x}^{k-t+m}\right)$, $k \in K_3$, correspond to consecutive elements within the original BCD sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$. Second observe that, by the definition of

a BCD update, each element $\mathbf{x}^k$, $k \geq 1$, of the BCD sequence $\{\mathbf{x}^k\}_{k=1}^{\infty}$ differs from its following element $\mathbf{x}^{k+1}$ in at most one block coordinate, and so any two consecutive elements within each vicinity

$\left(\mathbf{x}^{k-t+1}, \ldots, \mathbf{x}^k, \ldots, \mathbf{x}^{k-t+m}\right)$, $k \in K_3$, differ in at most one block coordinate. Specifically, any two consecutive elements $\mathbf{x}^{k-t+i-1}$ and $\mathbf{x}^{k-t+i}$, $k \in K_3$, $i = 2, \ldots, m$, differ only in the values of their block coordinates $\mathbf{x}_i$; this follows by the role of t in Subsequence Extractions 2 and 3. These observations hold true in the limit $\left(\overline{\mathbf{x}}^1, \overline{\mathbf{x}}^2, \ldots, \overline{\mathbf{x}}^m\right)$, and the claim of part 2 follows. □

The statement of Proposition 7 requires the definition of pseudoconvexity. A continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be *pseudoconvex* over a convex set $X \subseteq \mathbb{R}^n$ if the implication

$$\nabla f(\mathbf{x}_1)(\mathbf{x}_2 - \mathbf{x}_1) \geq 0 \implies f(\mathbf{x}_2) \geq f(\mathbf{x}_1)$$

holds for all $\mathbf{x}_1, \mathbf{x}_2 \in X$. Of particular interest is the pseudoconvexity condition given by:

$$\begin{aligned} &f(\overline{\mathbf{x}}_1^\ell, \ldots, \overline{\mathbf{x}}_{a-1}^\ell, \cdot, \ldots, \cdot, \overline{\mathbf{x}}_{b+1}^\ell, \ldots, \overline{\mathbf{x}}_m^\ell) : \textstyle\prod_{i=a}^b \mathbb{R}^{n_i} \to \mathbb{R}, \\ &\qquad \text{for each } \overline{\mathbf{x}}^\ell,\ a \leq \ell \leq b, \text{is pseudoconvex over } \textstyle\prod_{i=a}^b X_i, \end{aligned} \tag{2.39}$$

where $\overline{\mathbf{x}}^\ell$ is a component of some limit vicinity $\left(\overline{\mathbf{x}}^1, \overline{\mathbf{x}}^2, \ldots, \overline{\mathbf{x}}^m\right)$ and $a, b \in \mathbb{Z}$, $1 \leq a \leq b \leq m$, are bounding indices.

Proposition 7. *Given $\mathbf{x}^0 \in X$, let a sequence $\{\mathbf{x}^k\}_{k=1}^{\infty}$ be generated with BCD applied to problem (2.6). With the same $\mathbf{x}^0 \in X$, let problem (2.6) satisfy the BCD Problem Assumptions and let the sequence $\{\mathbf{x}^k\}_{k=1}^{\infty}$ satisfy the BCD Sequence Assumptions. If the pseudoconvexity condition (2.39) holds for some limit vicinity $(\overline{\mathbf{x}}^1, \ldots, \overline{\mathbf{x}}^m)$ of $\{\mathbf{x}^k\}_{k=1}^{\infty}$ with bounding indices $a, b \in \mathbb{Z}$, $1 \leq a \leq b \leq m$, then, for each ℓ, $a \leq \ell \leq b$, we have*

$$\begin{aligned} f(\overline{\mathbf{x}}_1^\ell, \ldots, \overline{\mathbf{x}}_m^\ell) \leq f(\overline{\mathbf{x}}_1^\ell, \ldots, \overline{\mathbf{x}}_{a-1}^\ell, \mathbf{x}_a, \ldots, \mathbf{x}_b, \overline{\mathbf{x}}_{b+1}^\ell, \ldots, \overline{\mathbf{x}}_m^\ell) \\ \text{for all } \mathbf{x}_i \in X_i,\ i = a, \ldots, b. \end{aligned} \tag{2.40}$$

Furthermore, the following stationary point conditions are satisfied:

1. for $a > 1$,

$$\nabla_{\mathbf{x}_{a-1},\mathbf{x}_a,\ldots,\mathbf{x}_b} f(\overline{\mathbf{x}}^{a-1}) \begin{bmatrix} \mathbf{x}_{a-1} - \overline{\mathbf{x}}_{a-1}^{a-1} \\ \mathbf{x}_a - \overline{\mathbf{x}}_a^{a-1} \\ \vdots \\ \mathbf{x}_b - \overline{\mathbf{x}}_b^{a-1} \end{bmatrix} \geq 0 \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = a-1, a, \ldots, b, \tag{2.41}$$

2. for $b < m$,

$$\nabla_{\mathbf{x}_a,\ldots,\mathbf{x}_b,\mathbf{x}_{b+1}} f(\overline{\mathbf{x}}^{b}) \begin{bmatrix} \mathbf{x}_a - \overline{\mathbf{x}}_a^{b} \\ \vdots \\ \mathbf{x}_b - \overline{\mathbf{x}}_b^{b} \\ \mathbf{x}_{b+1} - \overline{\mathbf{x}}_{b+1}^{b} \end{bmatrix} \geq 0 \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = a, \ldots, b, b+1. \tag{2.42}$$

Proof. Taking the optimality of $\overline{\mathbf{x}}^b$ over $\mathbf{x}_b \in X_b$, implied by the inequality

$$f(\overline{\mathbf{x}}_1^b, \ldots, \overline{\mathbf{x}}_m^b) \leq f(\overline{\mathbf{x}}_1^b, \ldots, \overline{\mathbf{x}}_{b-1}^b, \mathbf{x}_b, \overline{\mathbf{x}}_{b+1}^b, \ldots, \overline{\mathbf{x}}_m^b) \quad \text{for all } \mathbf{x}_b \in X_b \tag{2.43}$$

as the basis case, it is shown inductively for $\ell = a, \ldots, b$ that

$$f(\overline{\mathbf{x}}_1^\ell, \ldots, \overline{\mathbf{x}}_m^\ell) \leq f(\overline{\mathbf{x}}_1^\ell, \ldots, \overline{\mathbf{x}}_{\ell-1}^\ell, \mathbf{x}_\ell, \ldots, \mathbf{x}_b, \overline{\mathbf{x}}_{b+1}^\ell, \ldots, \overline{\mathbf{x}}_m^\ell) \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = \ell, \ldots, b. \tag{2.44}$$

Assuming the inductive hypothesis (2.44) is true for some ℓ, $a < \ell \leq b$, we show an analogous inequality for $\ell - 1$. Using the facts that $f(\overline{\mathbf{x}}^{\ell-1}) = f(\overline{\mathbf{x}}^\ell)$ and $\overline{\mathbf{x}}_i^{\ell-1} = \overline{\mathbf{x}}_i^\ell$ for $i \neq \ell$ (Lemma 3), inequality (2.44) becomes

$$f(\overline{\mathbf{x}}_1^{\ell-1}, \ldots, \overline{\mathbf{x}}_m^{\ell-1}) \leq f(\overline{\mathbf{x}}_1^{\ell-1}, \ldots, \overline{\mathbf{x}}_{\ell-1}^{\ell-1}, \mathbf{x}_\ell, \ldots, \mathbf{x}_b, \overline{\mathbf{x}}_{b+1}^{\ell-1}, \ldots, \overline{\mathbf{x}}_m^{\ell-1}) \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = \ell, \ldots, b. \tag{2.45}$$

The inequality defining optimality at $\overline{\mathbf{x}}^{\ell-1}$ over $X_{\ell-1}$ given by

$$f(\overline{\mathbf{x}}_{\ell-1}^{\ell-1}, \overline{\mathbf{x}}_{\neg\ell-1}^{\ell-1}) \leq f(\mathbf{x}_{\ell-1}, \overline{\mathbf{x}}_{\neg\ell-1}^{\ell-1}) \quad \text{for all } \mathbf{x}_{\ell-1} \in X_{\ell-1}, \tag{2.46}$$

together with inequality (2.45) imply the following necessary conditions for minimization on the convex set $\prod_{i=\ell-1}^{b} X_i$ given by

$$\nabla_{\mathbf{x}_{\ell-1},\ldots,\mathbf{x}_b} f(\overline{\mathbf{x}}^{\ell-1}) \begin{bmatrix} \mathbf{x}_{\ell-1} - \overline{\mathbf{x}}_{\ell-1}^{\ell-1} \\ \vdots \\ \mathbf{x}_b - \overline{\mathbf{x}}_b^{\ell-1} \end{bmatrix} \geq 0 \quad \text{for all } \mathbf{x}_{\ell-1} \in X_{\ell-1}, \ldots, \mathbf{x}_b \in X_b. \tag{2.47}$$

The application of the pseudoconvexity condition (2.39) to inequality (2.47) implies the desired result (2.44) for $\ell - 1$. Thus, inequality (2.40) has been verified once $\ell = a$ is taken and it is noted by Lemma 3 that $f(\overline{\mathbf{x}}^a) = f(\overline{\mathbf{x}}^{a+1}) = \cdots = f(\overline{\mathbf{x}}^b)$ and $\overline{\mathbf{x}}_j^a = \overline{\mathbf{x}}_j^{a+1} = \cdots = \overline{\mathbf{x}}_j^b$ for $j < a$ and $j > b$.

The stationary point condition (2.41) is proven by taking one more induction step from $\ell = a$ to $\ell = a-1$, and identifying the resulting stationary point condition (2.47) with the stationary point condition (2.41) that is desired.

The stationary point condition (2.42) is proven as the necessary condition for optimality over the convex set $\prod_{i=a}^{b+1} X_i$ associated with the optimality implied by the following two inequalities. The first inequality takes the form (2.40) with $\ell = b$. The second inequality is obtained from the inequality defining optimality at $\overline{\mathbf{x}}^{b+1}$ over X_{b+1} given by

$$f(\overline{\mathbf{x}}_{b+1}^{b+1}, \overline{\mathbf{x}}_{\neg b+1}^{b+1}) \leq f(\mathbf{x}_{b+1}, \overline{\mathbf{x}}_{\neg b+1}^{b+1}) \quad \text{for all } \mathbf{x}_{b+1} \in X_{b+1}. \tag{2.48}$$

Inequality (2.48) becomes

$$f(\overline{\mathbf{x}}_{b+1}^{b}, \overline{\mathbf{x}}_{\neg b+1}^{b}) \leq f(\mathbf{x}_{b+1}, \overline{\mathbf{x}}_{\neg b+1}^{b}) \quad \text{for all } \mathbf{x}_{b+1} \in X_{b+1} \tag{2.49}$$

by Lemma 3. Inequality (2.40) with $\ell = b$ and inequality (2.49) imply the stationary point condition (2.42). □

Proposition 7 yields conditions under which BCD produces a solution $\mathbf{x}^*$ that is either optimal for (2.6) or otherwise satisfies the stationary point condition (2.36). This is stated below in Corollary 3. Analogues for implications 1 and 3 of Corollary 3 appear in [49] and [114], respectively, while implication 2 is new.

Corollary 3. *Given a starting point $\mathbf{x}^0 \in X$, let a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ be generated with BCD applied*

to problem (2.6). *With the same* $\mathbf{x}^0 \in X$, *let problem* (2.6) *satisfy the BCD Problem Assumptions and let the sequence* $\{\mathbf{x}^k\}_{k=1}^{\infty}$ *satisfy the BCD Sequence Assumptions. Let the pseudoconvexity condition* (2.39) *hold for some limit vicinity* $(\overline{\mathbf{x}}^1, \ldots, \overline{\mathbf{x}}^m)$ *of* $\{\mathbf{x}^k\}_{k=1}^{\infty}$ *for some bounding indices* a, b, $1 \leq a \leq b \leq m$.

1. *If* $a = 1$, $b = m$, *then each point* $\overline{\mathbf{x}}^\ell$, $\ell = 1, \ldots, m$, *is a globally optimal solution for problem* (2.6).

2. *If* $a = 2$, $b = m$, *then the point* $\overline{\mathbf{x}}^1$ *satisfies the stationary point condition*

$$\nabla_{\mathbf{x}_1,\ldots,\mathbf{x}_m} f(\overline{\mathbf{x}}^1) \begin{bmatrix} \mathbf{x}_1 - \overline{\mathbf{x}}_1^1 \\ \vdots \\ \mathbf{x}_m - \overline{\mathbf{x}}_m^1 \end{bmatrix} \geq 0 \quad \textit{for all } \mathbf{x}_i \in X_i,\ i = 1, \ldots, m \tag{2.50}$$

   for problem (2.6).

3. *If* $a = 1$ *and* $b = m - 1$, *then the point* $\overline{\mathbf{x}}^{m-1}$ *satisfies the stationary point condition*

$$\nabla_{\mathbf{x}_1,\ldots,\mathbf{x}_m} f(\overline{\mathbf{x}}^{m-1}) \begin{bmatrix} \mathbf{x}_1 - \overline{\mathbf{x}}_1^{m-1} \\ \vdots \\ \mathbf{x}_m - \overline{\mathbf{x}}_m^{m-1} \end{bmatrix} \geq 0 \quad \textit{for all } \mathbf{x}_i \in X_i,\ i = 1, \ldots, m \tag{2.51}$$

   for problem (2.6).

Conditions other than pseudoconvexity, such as the uniqueness of minimization for the subproblems (2.34), have also been studied [114]. Proposition 8 is extracted from the proof of Theorem 4.1(c) of [114] and is used in Corollary 4 as a bridge between the stationary point conditions (2.41) and (2.42).

Proposition 8. *Given a starting point* $\mathbf{x}^0 \in X$, *let a sequence* $\{\mathbf{x}^k\}_{k=1}^{\infty}$ *be generated with BCD applied to problem* (2.6). *With the same* $\mathbf{x}^0 \in X$, *let problem* (2.6) *satisfy the BCD Problem Assumptions and let the sequence* $\{\mathbf{x}^k\}_{k=1}^{\infty}$ *satisfy the BCD Sequence Assumptions. If, for some limit vicinity* $(\overline{\mathbf{x}}^1, \ldots, \overline{\mathbf{x}}^m)$ *of* $\{\mathbf{x}^k\}_{k=1}^{\infty}$, *each function* $f(\,\cdot\,, \overline{\mathbf{x}}^i_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}$, $1 \leq c < i < d \leq m$, *is uniquely minimized over* $\mathbf{x}_i \in X_i$, *then* $\overline{\mathbf{x}}^c = \overline{\mathbf{x}}^{c+1} = \cdots = \overline{\mathbf{x}}^{d-1}$. *Setting* $\overline{\mathbf{x}} := \overline{\mathbf{x}}^c = \overline{\mathbf{x}}^{c+1} = \cdots = \overline{\mathbf{x}}^{d-1}$, *the*

stationary point condition

$$\nabla_{\mathbf{x}_c,\ldots,\mathbf{x}_d} f(\overline{\mathbf{x}}) \begin{bmatrix} \mathbf{x}_c - \overline{\mathbf{x}}_c \\ \vdots \\ \mathbf{x}_d - \overline{\mathbf{x}}_d \end{bmatrix} \geq 0 \quad \textit{for all } \mathbf{x} \in X \tag{2.52}$$

is satisfied for problem (2.6).

Proof. By Lemma 3, $f(\overline{\mathbf{x}}^{i-1}) = f(\overline{\mathbf{x}}^i)$ and $\overline{\mathbf{x}}^{i-1}_{\neg i} = \overline{\mathbf{x}}^i_{\neg i}$ for $i = 2, \ldots, m$. Thus, $f(\overline{\mathbf{x}}^{i-1}_i, \overline{\mathbf{x}}^i_{\neg i}) = f(\overline{\mathbf{x}}^{i-1}_i, \overline{\mathbf{x}}^{i-1}_{\neg i}) = f(\overline{\mathbf{x}}^i_i, \overline{\mathbf{x}}^i_{\neg i})$. Due to the assumption that $f(\,\cdot\,, \overline{\mathbf{x}}^i_{\neg i})$ is uniquely minimized over $\mathbf{x}_i \in X_i$ for $i = c+1, \ldots, d-1$, we have $\overline{\mathbf{x}}^{i-1}_i = \overline{\mathbf{x}}^i_i$ for $i = c+1, \ldots, d-1$. Thus, it follows that $\overline{\mathbf{x}}^c = \overline{\mathbf{x}}^{c+1} = \cdots = \overline{\mathbf{x}}^{d-1}$ and so define $\overline{\mathbf{x}} := \overline{\mathbf{x}}^\ell$, for any $\ell = c, \ldots, d-1$. Then

$$f(\overline{\mathbf{x}}_i, \overline{\mathbf{x}}_{\neg i}) \leq f(\mathbf{x}_i, \overline{\mathbf{x}}_{\neg i}) \quad \text{for all } \mathbf{x}_i \in X_i, \tag{2.53}$$

for each $i = c, \ldots, d-1$. It remains to show that inequality (2.53) also holds for $i = d$. To this end, consider the inequality implied by optimality over X_d given by

$$f(\overline{\mathbf{x}}^d_d, \overline{\mathbf{x}}^d_{\neg d}) \leq f(\mathbf{x}_d, \overline{\mathbf{x}}^d_{\neg d}) \quad \text{for all } \mathbf{x}_d \in X_d. \tag{2.54}$$

Once Lemma 3 is used to substitute $f(\overline{\mathbf{x}}^d)$ with $f(\overline{\mathbf{x}}^{d-1})$ and $\overline{\mathbf{x}}^d_{\neg d}$ with $\overline{\mathbf{x}}^{d-1}_{\neg d}$ in (2.54), inequality (2.53) follows for $i = d$ since $\overline{\mathbf{x}}^{d-1} = \overline{\mathbf{x}}$. Combining the necessary conditions of optimality at $\overline{\mathbf{x}}$ implied by the inequalities (2.53), $i = c, \ldots, d$, we have the desired inequality (2.52). □

Corollary 4. *Given a starting point $\mathbf{x}^0 \in X$, let a sequence $\{\mathbf{x}^k\}_{k=1}^\infty$ be generated with BCD applied to problem* (2.6). *With the same $\mathbf{x}^0 \in X$, let problem* (2.6) *satisfy the BCD Problem Assumptions and let the sequence $\{\mathbf{x}^k\}_{k=1}^\infty$ satisfy the BCD Sequence Assumptions. If, for some limit vicinity $(\overline{\mathbf{x}}^1, \ldots, \overline{\mathbf{x}}^m)$ of $\{\mathbf{x}^k\}_{k=1}^\infty$ and for some indices $c, d \in \mathbb{Z}$, $1 \leq c < d \leq m$, the pseudoconvexity condition* (2.39) *is satisfied with the bounding indices $a = 1$, $b = c$, and also with $a = d$, $b = m$; and each function $f(\,\cdot\,, \overline{\mathbf{x}}^i_{\neg i}) : \mathbb{R}^{n_i} \to \mathbb{R}$, $c < i < d$, is uniquely minimized over $\mathbf{x}_i \in X_i$, then the stationary point condition* (2.36) *is satisfied for problem* (2.6) *with $\overline{\mathbf{x}} := \overline{\mathbf{x}}^c = \overline{\mathbf{x}}^{c+1} = \cdots = \overline{\mathbf{x}}^{d-1}$.*

Proof. By Proposition 7, the stationary point condition (2.42) with $a = 1$ and $b = c$ given by

$$\nabla_{\mathbf{x}_1,\ldots,\mathbf{x}_c,\mathbf{x}_{c+1}} f(\overline{\mathbf{x}}^c) \begin{bmatrix} \mathbf{x}_1 - \overline{\mathbf{x}}_1^c \\ \vdots \\ \mathbf{x}_c - \overline{\mathbf{x}}_c^c \\ \mathbf{x}_{c+1} - \overline{\mathbf{x}}_{c+1}^c \end{bmatrix} \geq 0 \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = 1, \ldots, c, c+1, \tag{2.55}$$

and the stationary point condition (2.41) with $a = d$ and $b = m$ given by

$$\nabla_{\mathbf{x}_{d-1},\mathbf{x}_d,\ldots,\mathbf{x}_m} f(\overline{\mathbf{x}}^{d-1}) \begin{bmatrix} \mathbf{x}_{d-1} - \overline{\mathbf{x}}_{d-1}^{d-1} \\ \mathbf{x}_d - \overline{\mathbf{x}}_d^{d-1} \\ \vdots \\ \mathbf{x}_m - \overline{\mathbf{x}}_m^{d-1} \end{bmatrix} \geq 0 \qquad \text{for all } \mathbf{x}_i \in X_i,\ i = d-1, d, \ldots, m, \tag{2.56}$$

for problem (2.6) are met. By Proposition 8, the stationary point condition (2.52) holds with $\overline{\mathbf{x}} = \overline{\mathbf{x}}^c = \cdots = \overline{\mathbf{x}}^{d-1}$. Setting $\overline{\mathbf{x}} := \overline{\mathbf{x}}^c = \cdots = \overline{\mathbf{x}}^{d-1}$, the stationary point conditions (2.52), (2.55), and (2.56) are combined yielding (2.36). □

Propositions 7, 8 and Corollaries 3, 4 build on the foundations established in [49, 114] in the following manner:

1. The concepts of the vicinity sequence and the limiting vicinity are made more explicit, facilitating the development of proofs and improving theoretical insight. The role of the pseudoconvexity condition (2.39) as a means of "gluing together" stationary point conditions is made more clear.

2. The proof of inequality (2.40) in Proposition 7 is fundamentally different from the proof establishing the analogous inequality in Theorem 4.1 of [114]. The latter proof naturally leads to the stationary point condition (2.42), whose analogs are present in Theorem 4.1(a) in [114]; the proof given in Proposition 7 leads more naturally to the stationary point condition (2.41), which is new.

3. Finally, the integration of the uniqueness of minimization conditions (Proposition 8) and the

pseudoconvexity conditions (2.39) allows for the "gluing together" of the stationary point condition (2.52) with the two stationary point conditions (2.41) and (2.42). This integration as accomplished in Corollary 4 is also new.

2.5 Multi-Objective Decomposition Algorithms (MODAs)

The developments of Sections 2.2, 2.3, and 2.4 are now ready to be integrated and applied for the distributed computation of efficient points in a multiobjective optimization setting. The integrated application of Lagrangian coordination and the BCD method to an SOP reformulation (2.4) of MOP (2.1) is now addressed.

2.5.1 Integrating Lagrangian coordination with the BCD method

The goal of this subsection is to state and analyze an algorithm for generating a sequence $\left\{\mathbf{x}^k\right\}_{k=1}^{\infty}$ using BCD applied to problem (2.6) while applying updates of $\mathbf{v}$ and μ to tighten the relaxation (2.6) of problem (2.4). Motivated by an integration of the convergence results given in Corollaries 2, 3, and 4, Alg. 2 is stated. The first undefined argument $\tau^1 > 0$ is used to generate a monotonically nonincreasing sequence $\left\{\tau^k\right\}_{k=1}^{\infty}$ of positive convergence tolerance values. The second undefined argument ζ specifies the block update index i at which updates of $\mathbf{v}$ and μ can occur.

Proposition 9 provides conditions under which the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ generated by Alg. 2 converges to a pair $(\mathbf{x}^*, \mathbf{v}^*)$ satisfying the necessary conditions of optimality (2.25) for problem (2.4).

Proposition 9. *Let a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ be generated by Alg. 2 applied to problem (2.4), and let $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}_{k=1}^{\infty}$ converge to $(\mathbf{x}^*, \mathbf{v}^*)$. Let the decomposable relaxation (2.6) of problem (2.4) satisfy the BCD Problem Assumptions for each $\mathbf{v} = \mathbf{v}^k$, $\mu = \mu^k$, $k \geq 1$, and for all starting points $\mathbf{x}^0 \in X$.*

Define $L_{\mathbf{v},\mu} : \prod_{i=1}^{m} \mathbb{R}^{n_i} \to \mathbb{R}$ as in Alg. 2, and define $K := \{k_1, k_2, \ldots,\} \subseteq \{1, 2 \ldots,\}$ as the sequence of indices at which the calls to `CheckConvergence` within Alg. 2 return `true`. Let each $(\overline{\mathbf{x}}^{1,k_j}, \ldots, \overline{\mathbf{x}}^{m,k_j})$, $k_j \in K$, denote a limit vicinity of the sequence computed with BCD applied to problem (2.6) with $\mathbf{v} = \mathbf{v}^{k_j}$, $\mu = \mu^{k_j}$. If there exists a sequence $\left\{(\overline{\mathbf{x}}^{1,k_j}, \ldots, \overline{\mathbf{x}}^{m,k_j})\right\}_{k_j \in K}$ such that the following assumptions hold:

1. $\lim_{j\to\infty} \left\|\mathbf{x}^{k_j} - \overline{\mathbf{x}}^{\zeta,k_j}\right\|_2 = 0$, *where ζ such that $1 \leq \zeta \leq m$ is an input to Alg. 2.*

2. *For all* $(\overline{\mathbf{x}}^{1,k_j},\ldots,\overline{\mathbf{x}}^{m,k_j})$, $k_j \in K$, *there exist indices* $c,d \in \mathbb{Z}$, *such that* $1 \leq c \leq \zeta < d \leq m$ *and*

   (a) *pseudoconvexity condition* (2.39) *holds for both* $a = 1, b = c$ *and* $a = d, b = m$ *with* $f = L_{\mathbf{v},\mu}$ *and* $(\overline{\mathbf{x}}^1,\ldots,\overline{\mathbf{x}}^m) = (\overline{\mathbf{x}}^{1,k_j},\ldots,\overline{\mathbf{x}}^{m,k_j})$;

   (b) $L_{\mathbf{v},\mu}(\mathbf{x}_i, \overline{\mathbf{x}}^{i,k_j}_{\neg i})$ *is uniquely minimized over* X_i *for* $c < i < d$;

3. $0 < \mu^{k_j} < \mu^{k_{j+1}}$ *for all* $j \geq 1$, $\lim_{j\to\infty} \mu^{k_j} = \infty$ *and* $\lim_{j\to\infty} \mu^{k_j} \left\|\mathbf{h}(\mathbf{x}^{k_j}) - \mathbf{h}(\overline{\mathbf{x}}^{\zeta,k_j})\right\|_2 = 0$,

then $(\mathbf{x}^*, \mathbf{v}^*)$ *satisfies the necessary conditions of optimality* (2.25) *for problem* (2.4). *If assumption 2 is replaced with the alternative assumption that for all* $(\overline{\mathbf{x}}^{1,k_j},\ldots,\overline{\mathbf{x}}^{m,k_j})$, $k_j \in K$, *the pseudoconvexity condition* (2.39) *holds for* $a = 1, b = m$, *then* $\mathbf{x}^*$ *is optimal for problem* (2.4).

Proof. The BCD Sequence Assumptions as directly enforced in Alg. 2, the BCD Problem Assumptions, and assumption 2 imply by Corollary 4 that each limit point $\overline{\mathbf{x}}^{\zeta,k_j}$, $k_j \in K$, satisfies the stationary point condition (2.28) for problem (2.6) with $\mathbf{v} = \mathbf{v}^{k_j}$ and $\mu = \mu^{k_j}$ (and each $\overline{\mathbf{x}}^{\zeta,k_j}$, $1 \leq \zeta \leq m$, is optimal for problem (2.6) with $\mathbf{v} = \mathbf{v}^{k_j}$ and $\mu = \mu^{k_j}$ for every $k_j \in K$ under the alternative version of assumption 2 by Corollary 3). Once assumptions 1 and 3 are added, it follows by Corollary 2 that $(\mathbf{x}^*, \mathbf{v}^*)$ satisfies the necessary conditions of optimality (2.25) for problem (2.4) (and $\mathbf{x}^*$ is globally optimal for (2.4) under the alternative version of assumption 2). □

2.5.2 Algorithm

The **D**ecision **S**pace **D**ecomposition **A**lgorithm (DSDA) is now stated for computing efficient solutions for the AiO decomposable problem (2.2) using computations based on the subproblems (2.3). First, a quadratic scalarization of the form (2.19) is applied to the AiO decomposable problem (2.2). The resulting problem is given by

$$\begin{aligned} \min_{\mathbf{x}_1\in X_1,\ldots,\mathbf{x}_m\in X_m,\mathbf{s}\geqq\mathbf{0}} \quad & \mathbf{1}^T\left[D_{\mathbf{w}}\left(\mathbf{f}(\mathbf{x}_1,\ldots,\mathbf{x}_m) - \mathbf{z}^U\right) + \mathbf{s}\right] \\ \text{s.t.} \quad & \begin{bmatrix} U^T\left[D_{\mathbf{w}}\left(\mathbf{f}(\mathbf{x}_1,\ldots,\mathbf{x}_m) - \mathbf{z}^U\right) + \mathbf{s}\right] \\ \mathbf{h}(\mathbf{x}_1,\ldots,\mathbf{x}_m) \end{bmatrix} = \mathbf{0}, \end{aligned} \tag{2.57}$$

Algorithm 2 BCD and Lagrange Coordination Algorithm.

function COOR(f, $\prod_{i=1}^{m} X_i$, $\mathbf{h}$, $\mathbf{x}^0$, $\mathbf{v}^1$, μ^1, τ^1, ζ)
  $\ddot{\mathbf{x}} \leftarrow \mathbf{x}^0, \quad L_{\mathbf{v},\mu}(\mathbf{x}) \leftarrow f(\mathbf{x}) + \mathbf{v}^1\mathbf{h}(\mathbf{x}) + \frac{\mu^1}{2}\|\mathbf{h}(\mathbf{x})\|_2^2$
  for $k = 1, 2, \ldots$ **do**
    $i \leftarrow (k-1 \mod m) + 1, \quad \ddot{\mathbf{x}}_i \leftarrow \underset{\mathbf{x}_i \in X_i}{\operatorname{argmin}} L_{\mathbf{v},\mu}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}), \quad \mathbf{x}^k \leftarrow \ddot{\mathbf{x}}$
    if CheckConvergence(k, ζ, τ^k) **then**
      $\mathbf{v}^{k+1} \leftarrow \mathbf{v}^k + \mu^k\,\mathbf{h}(\mathbf{x}^k), \quad \mu^{k+1} \leftarrow c\mu^k,\ 1 \le c \le 10, \quad \tau^{k+1} \leftarrow \frac{\tau^k}{2}$
      $L_{\mathbf{v},\mu}(\mathbf{x}) \leftarrow f(\mathbf{x}) + \mathbf{v}^{k+1}\mathbf{h}(\mathbf{x}) + \frac{\mu^{k+1}}{2}\|\mathbf{h}(\mathbf{x})\|_2^2$
    else
      $\mathbf{v}^{k+1} \leftarrow \mathbf{v}^k, \quad \mu^{k+1} \leftarrow \mu^k, \quad \tau^{k+1} \leftarrow \tau^k$
    end if
  end for
  return $\left\{\left(\mathbf{x}^k, \mathbf{v}^k\right)\right\}_{k=1}^{\infty}$
end function
function CHECKCONVERGENCE(k, ζ, τ^k)
  return ($k > 2\,m$ **and** $\sum_{i=0}^{m-1} \left\|\mathbf{x}^{k-i-m} - \mathbf{x}^{k-i}\right\|_2 < \tau^k$ **and** $k - 1 \equiv \zeta - 1 \mod m$)
end function

where $\mathbf{z}^U \in \mathbb{R}^p$ is a reference utopia point. The single objective reformulation (2.57) decomposes into the subproblems

$$\begin{array}{ll} \min\limits_{\mathbf{x}_i \in X_i} & \mathbf{1}^T \left[D_{\mathbf{w}}\left(\mathbf{f}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) - \mathbf{z}^U\right) + \ddot{\mathbf{s}}\right] \\ \text{s.t.} & \left[\begin{array}{c} U^T \left[D_{\mathbf{w}}\left(\mathbf{f}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) - \mathbf{z}^U\right) + \ddot{\mathbf{s}}\right] \\ \mathbf{h}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) \end{array}\right] = \mathbf{0}, \end{array} \tag{2.58}$$

for $i = 1, \ldots, m$, with the addition of one more subproblem given by

$$\begin{array}{ll} \min\limits_{\mathbf{s} \geqq \mathbf{0}} & \mathbf{1}^T \left[D_{\mathbf{w}}\left(\mathbf{f}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_m) - \mathbf{z}^U\right) + \mathbf{s}\right] \\ \text{s.t.} & \left[\begin{array}{c} U^T \left[D_{\mathbf{w}}\left(\mathbf{f}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_m) - \mathbf{z}^U\right) + \mathbf{s}\right] \\ \mathbf{h}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_m) \end{array}\right] = \mathbf{0}. \end{array} \tag{2.59}$$

DSDA consists of the repeated application of Alg. 2 to the subproblems (2.58), $i = 1, \ldots, m$, and to the subproblem (2.59), while applying augmented Lagrangian relaxation to the equality constraint $\left[\begin{array}{c} U^T \left[D_{\mathbf{w}}\left(\mathbf{f}(\mathbf{x}_1, \ldots, \mathbf{x}_m) - \mathbf{z}^U\right) + \mathbf{s}\right] \\ \mathbf{h}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \end{array}\right] = \mathbf{0}$ of (2.57). Based on the use of the quadratic scalarization as analyzed in Proposition 3, Proposition 10 states conditions under which the points $\mathbf{x}^* \in E$ computed by Alg. 3 applied to the AiO decomposable problem (2.2) are weakly efficient for (2.2).

Algorithm 3 Decision Space Decomposition Algorithm (DSDA)

function DSDA($\mathbf{f}$, $\prod_{i=1}^{m} X_i$, $\mathbf{h}$, $\mathcal{W}$, $\mathbf{x}^0$, $\mathbf{v}^1$, μ^1, $\mathbf{z}^U$, τ^1, ζ)
  $E \leftarrow \emptyset$, $S \leftarrow \{\mathbf{s} \in \mathbb{R}^p : \mathbf{s} \geqq \mathbf{0}\}$
  for $\mathbf{w} \in \mathcal{W}$ **do**
    $U \leftarrow$ OrthonormalBasisMatrix($\{\mathbf{w}\}^{\perp}$), $D_{\mathbf{w}} \leftarrow$ DiagonalMatrix($\mathbf{w}$)
    $f(\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{s}) \leftarrow \mathbf{1}^T \left[D_{\mathbf{w}} \left(\mathbf{f}(\mathbf{x}_1, \ldots, \mathbf{x}_m) - \mathbf{z}^U \right) + \mathbf{s} \right]$
    $\overline{\mathbf{h}}(\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{s}) \leftarrow \begin{bmatrix} U^T \left[D_{\mathbf{w}} \left(\mathbf{f}(\mathbf{x}_1, \ldots, \mathbf{x}_m) - \mathbf{z}^U \right) + \mathbf{s} \right] \\ \mathbf{h}(\mathbf{x}_1, \ldots, \mathbf{x}_m) \end{bmatrix}$
    $\mathbf{s}^0 \leftarrow \mathbf{0}$
    $(\overline{\mathbf{x}}, \overline{\mathbf{s}}, \overline{\mathbf{v}}) \leftarrow$ LimitPoint(Coor(f, $(\prod_{i=1}^{m} X_i) \times S$, $\overline{\mathbf{h}}$, $(\mathbf{x}^0, \mathbf{s}^0)$, $\mathbf{v}^1$, μ^1, τ^1, ζ))
    $E \leftarrow E \cup \{\overline{\mathbf{x}}\}$
  end for
  return E
end function

Proposition 10. *During the execution of Alg. 3, let the AiO decomposable MOP* (2.2) *be reformulated as problem* (2.57) *for some fixed weight* $\mathbf{w} > \mathbf{0}$ *and utopia point* $\mathbf{z}^U$*, and let problem* (2.57) *meet the quadratic growth condition and the second order sufficiency conditions for optimality. If the application of Alg. 2 applied to problem* (2.57) *with the subproblem decomposition* (2.58) *and* (2.59) *produces an optimal solution* $(\mathbf{x}^*, \mathbf{s}^*)$ *to problem* (2.57)*, then* $\mathbf{x}^*$ *is weakly efficient for MOP* (2.1)*.*

The optimality of $(\mathbf{x}^*, \mathbf{s}^*)$ for (2.57) depends, of course, on certain conditions being met during the execution of Alg. 2. Sufficient conditions for $(\mathbf{x}^*, \mathbf{s}^*)$ to be optimal for (2.57) are given in Proposition 9.

The second algorithm OSDA computes solutions for a decomposable MOP of the form

$$\begin{aligned} \min_{\mathbf{x}_1, \ldots, \mathbf{x}_m} \quad & [\mathbf{f}_1(\mathbf{x}_1), \ldots, \mathbf{f}_m(\mathbf{x}_m)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \mathbf{0} \\ & \mathbf{x}_i \in X_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2.60}$$

The AiO problem (2.60) is a specialization of the AiO problem (2.2) in that the decision space decomposition $\prod_{i=1}^{m} \mathbb{R}^{n_i}$ is associated with an objective space decomposition $\prod_{i=1}^{m} \mathbb{R}^{p_i}$ appearing in a partition $\mathbf{f} := [\mathbf{f}_1, \ldots, \mathbf{f}_m]$, $\mathbf{f}_i : \mathbb{R}^{n_i} \to \mathbb{R}^{p_i}$, $i = 1, \ldots, m$, of the objective functions. OSDA computes efficient solutions for the AiO decomposable problem (2.60) using computations based on

the subproblem decomposition

$$\begin{aligned} \min_{\mathbf{x}_i} \quad & \mathbf{f}_i(\mathbf{x}_i) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_i, \ddot{\mathbf{x}}_{\neg i}) = \mathbf{0} \\ & \mathbf{x}_i \in X_i. \end{aligned} \tag{2.61}$$

The computations are performed on the following single-objective reformulations of (2.60) and (2.61). First, apply a quadratic scalarizations f_{q_i} to each $\mathbf{f}_i$ defined by

$$f_{q_i} := \frac{1}{2}(\mathbf{f}_i(\mathbf{x}_i) - \mathbf{y}_i^r)^T Q_i(\mathbf{f}_i(\mathbf{x}_i) - \mathbf{y}_i^r) + (\mathbf{f}_i(\mathbf{x}_i) - \mathbf{y}_i^r)^T \mathbf{q}_i \tag{2.62}$$

for $i = 1, \ldots, m$, where Q_i is a $p_i \times p_i$ positive semidefinite matrix, $\mathbf{q}_i$ is a $p_i \times 1$ vector, and $\mathbf{y}_i^r \in \mathbb{R}^{p_i}$ is a reference point. After applying the scalarizations (2.62), the AiO decomposable problem (2.60) becomes

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & [f_{q_1}(\mathbf{x}_1), \ldots, f_{q_m}(\mathbf{x}_m)] \\ \text{s.t.} \quad & \mathbf{h}\left(\mathbf{x}_1, \ldots, \mathbf{x}_m\right) = \mathbf{0} \\ & \mathbf{x}_i \in X_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2.63}$$

Applying the weighted-sum method to problem (2.63) yields problem (2.64) whose objective function is additively separable.

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & \sum_{i=1}^{m} \alpha_i f_{q_i}(\mathbf{x}_i) \\ \text{s.t.} \quad & \mathbf{h}\left(\mathbf{x}_1, \ldots, \mathbf{x}_m\right) = \mathbf{0} \\ & \mathbf{x}_i \in X_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2.64}$$

The additive separability of the objective function in problem (2.64) suggests the use of the *alternating direction method of multipliers* (ADMM) as a solution approach [47, 40, 9, 32, 69, 14]. OSDA consists of the repeated application of ADMM to subproblems of the form (2.64) as given in Algorithm 4.

Proposition 11 gives conditions under which OSDA computes efficient points for an objective space decomposed problem (2.60).

Algorithm 4 **O**bjective-**S**pace **D**ecomposition **A**lgorithm (OSDA)

function OSDA($\mathbf{f}_1, \ldots, \mathbf{f}_m$, $\prod_{i=1}^m X_i$, $\mathbf{h}$, $\prod_{i=1}^m \mathcal{P}_i$, $\mathcal{A}$, $\mathbf{x}^0$, $\mathbf{v}^0$, μ, τ)
  $E \leftarrow \emptyset$
  for $\prod_{i=1}^m (\mathbf{y}_i^r, \mathbf{q}_i, Q_i) \in \prod_{i=1}^m \mathcal{P}_i$ **and** $\alpha \in \mathcal{A}$ **do**
    for $i = 1, \ldots, m$ **do**
      $f_i \leftarrow (\mathbf{f}_i - \mathbf{y}_i^r)^T Q_i (\mathbf{f}_i - \mathbf{y}_i^r) + (\mathbf{f}_i - \mathbf{y}_i^r)^T \mathbf{q}_i$
      $f_i \leftarrow \alpha_i f_i$
    end for
    $(\overline{\mathbf{x}}, \overline{\mathbf{v}}) \leftarrow$ ADMM($f_1, \ldots, f_m$, $\prod_{i=1}^m X_i$, $\mathbf{h}$, $\mathbf{x}^0$, $\mathbf{v}^0$, μ, τ)
    if $\mathbf{f}_i(\overline{\mathbf{x}}_i) \in C_{Q_i, \mathbf{q}_i, \mathbf{y}_i^r}$ for each $i = 1, \ldots, m$ **then**
      $E \leftarrow E \cup \{\overline{\mathbf{x}}\}$
    end if
  end for
  return E
end function

---

function ADMM($f_1, \ldots, f_m$, $\prod_{i=1}^m X_i$, $\mathbf{h}$, $\mathbf{x}^0$, $\mathbf{v}^0$, μ, τ)
  $\ddot{\mathbf{x}} \leftarrow \mathbf{x}^0$, $\mathbf{v} \leftarrow \mathbf{v}^0$
  repeat
    $\mathbf{x}^{last} \leftarrow \ddot{\mathbf{x}}$
    for $i = 1, \ldots, m$ **do**
      $\ddot{\mathbf{x}}_i \leftarrow \underset{\mathbf{x}_i \in X_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \mathbf{v}^T \mathbf{h}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_{i-1}, \mathbf{x}_i, \ddot{\mathbf{x}}_{i+1}, \ldots, \ddot{\mathbf{x}}_m)$
        $+ \frac{\mu}{2} \|\mathbf{h}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_{i-1}, \mathbf{x}_i, \ddot{\mathbf{x}}_{i+1}, \ldots, \ddot{\mathbf{x}}_m)\|_2^2$
    end for
    $\mathbf{v} \leftarrow \mathbf{v} + \mu\, \mathbf{h}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_m)$
  until $\big(||\mathbf{x}^{last} - \ddot{\mathbf{x}}|| < \tau\big)$ **and** $\|\mathbf{h}(\ddot{\mathbf{x}}_1, \ldots, \ddot{\mathbf{x}}_m)\| < \tau$
  return $(\ddot{\mathbf{x}}, \mathbf{v})$
end function

Proposition 11. *Let a two-block $m = 2$ objective space decomposition* (2.60) *be applied to problem* (2.1). *Assume that the constraint sets $X_1 \subseteq \mathbb{R}^{n_1}$, $X_2 \subseteq \mathbb{R}^{n_2}$ are closed and convex; that $\mathbf{h} : X_1 \times X_2 \to \mathbb{R}^q$ is linear; that the set*

$$\left(\prod_{i=1}^m X_i \right) \cap \{\mathbf{x} : \mathbf{h}(\mathbf{x}) = \mathbf{0}\}$$

is nonempty; and the scalarized problem (2.64) *has an optimal solution. If f_{q_i} is proper, continuous, and convex for $i = 1, \ldots, p$, then a solution $\mathbf{x}^* \in E$ computed by Algorithm 4 is locally efficient for the AiO problem* (2.60).

Proof. The optimality of the solution $\mathbf{x}^* \in E$ for the scalarized decomposable problem (2.64) follows from the classic convergence results for the two-block $m = 2$ decomposition found, for example, in [9, 32, 69]. Since the critical set (2.11) condition of Proposition 1 is tested for the generated $\mathbf{x}^*$ in

Algorithm 4, this same optimal solution $\mathbf{x}^*$ is also locally efficient for the AiO problem (2.60) by Proposition 1. □

Studies of ADMM convergence is an active area of research, where the convergence of ADMM for the general multi-block case $m \geq 3$ under the same convexity assumptions on f_i is still an open question. In practice, the ADMM convergence results established for $m = 2$ seem to hold in the general multi-block case $m \geq 3$ [105, 90]; currently, proofs of convergence for the $m \geq 3$ case are given by imposing a stronger convexity requirement on the functions f_i [53], or by introducing a predictor-corrector mechanism into ADMM [55]. Needless to say, the ongoing developments of ADMM convergence studies will impact the convergence studies of algorithms such as Algorithm 4.

In Section 2.6, examples are provided for illustrating the use of OSDA and DSDA applied to a nonconvex MOP, and for demonstrating the weight-generating schemes and quadratic scalarization presented in Section 2.2.

2.6 Examples

The first example, based on an MOP with four convex objective functions, serves to illustrate the effect on the AiO computation of efficient points resulting from the use of different techniques for generating weight vectors. OSDA is also applied to this example as a multidisciplinary MOP decomposed into two subproblems in order to show the coordination of the two subproblems in computing the AiO efficient points.

The second example is based on a nonconvex MOP with four objective functions. The nonconvexity of this problem results in the weighted-sum scalarization being unable to compute many of the efficient points. This example is first used in its AiO form to demonstrate the ability of the quadratic form scalarization for computing efficient designs that are not computable using the weighted-sum scalarization. The problem is then decomposed into two subproblems, and the efficient points are computed using both DSDA and OSDA.

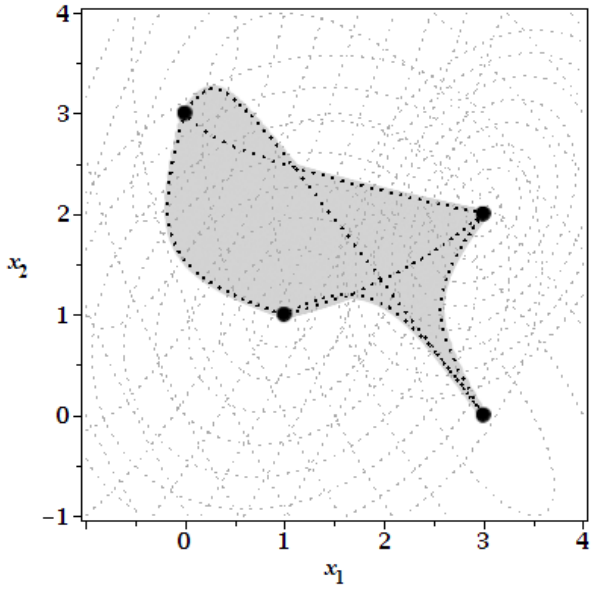

Figure 2.4: Efficient set and pairwise efficient curves for problem (2.66)

2.6.1 Convex MOP

For the first example problem, define $f_{i,j} : \mathbb{R}^2 \to \mathbb{R}$ for $i, j = 1, 2$ as follows.

$$\begin{aligned} f_{1,1}(x_1, x_2) &:= 4(x_1 - 0)^2 - 3(x_1 - 0)(x_2 - 3) + (x_2 - 3)^2 \\ f_{1,2}(x_1, x_2) &:= 2(x_1 - 3)^2 + 2(x_1 - 3)(x_2 - 0) + (x_2 - 0)^2 \\ f_{2,1}(x_1, x_2) &:= 5(x_1 - 1)^2 - 2(x_1 - 1)(x_2 - 1) + 5(x_2 - 1)^2 \\ f_{2,2}(x_1, x_2) &:= 17(x_1 - 3)^2 - 10(x_1 - 3)(x_2 - 2) + 11(x_2 - 2)^2 \end{aligned} \tag{2.65}$$

Defining $\mathbf{f} := [f_{1,1}, f_{1,2}, f_{2,1}, f_{2,2}]$, we have the following MOP.

$$\min_{x_1 \in \mathbb{R}, x_2 \in \mathbb{R}} \mathbf{f}(x_1, x_2) \tag{2.66}$$

Because each objective function $f_{i,j}$, $i, j = 1, 2$ is a convex quadratic form, each efficient point for (2.66) may be computed using the weighted-sum scalarization. The efficient set for problem (2.66) is plotted in Fig. 2.4 together with equal-value level curves, objective-wise minima for each objective function, and efficient curves taken for each pair of objectives.

Figure 2.5 shows the results of applying various weight sampling schemes for computing the

efficient points for the AiO MOP (2.66).

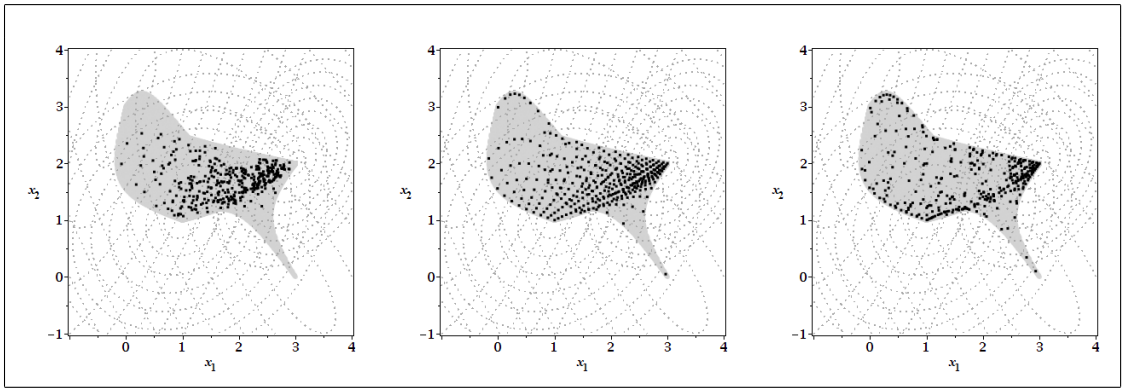

Figure 2.5: Random sampling (364 pts), uniform sampling (364 pts), and Sierpinski sampling (341 pts) applied to problem (2.66)

In order to apply OSDA, the objective space is decomposed into two subproblems so that Subproblem 1 is composed of $f_{1,1}$ and $f_{1,2}$ and Subproblem 2 is composed of $f_{2,1}$ and $f_{2,2}$. Based on the decomposition in the objective space, we induce a decomposition on the decision space by introducing copies of x_1 and x_2 written as

original variable	subproblem 1 copy	subproblem 2 copy
x_1	$x_{1,1}$	$x_{1,2}$
x_2	$x_{2,1}$	$x_{2,2}$

For compactness of notation, define

$$\begin{aligned} \mathbf{f}_1 &:= [f_{1,1}, f_{1,2}], & \mathbf{f}_2 &:= [f_{2,1}, f_{2,2}], \\ \mathbf{w}_1 &:= [w_{1,1}, w_{1,2}], & \mathbf{w}_2 &:= [w_{2,1}, w_{2,2}], \\ \mathbf{x}_1 &:= [x_{1,1}, x_{1,2}], & \mathbf{x}_2 &:= [x_{2,1}, x_{2,2}]. \end{aligned}$$

The constraint for coordinating the copies is written $\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0}$, where

$$\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} x_{1,1} - x_{2,1} \\ x_{1,2} - x_{2,2} \end{bmatrix}.$$

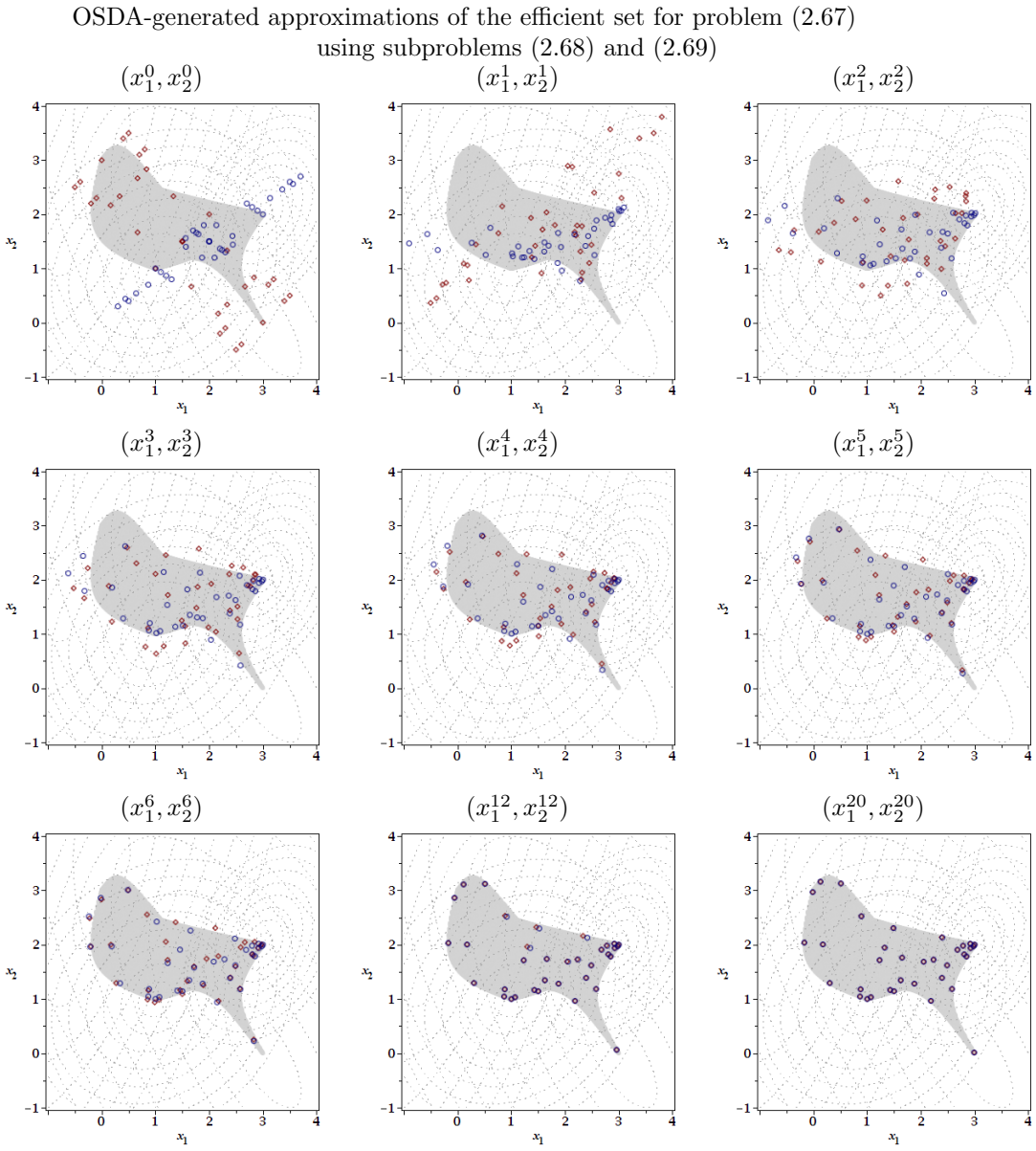

Figure 2.6: Coordination of level-specific copies of the AiO efficient set. The subproblem (2.68) approximations of the AiO efficient set are depicted with diamonds, and subproblem (2.69) approximations of the AiO efficient set are depicted with open circles.

The following AiO formulation for the decomposed problem is obtained

$$\begin{aligned} \min_{\mathbf{x}_1,\mathbf{x}_2\in\mathbb{R}^2} \quad & [\mathbf{f}_1(\mathbf{x}_1), \mathbf{f}_2(\mathbf{x}_2)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0}. \end{aligned} \tag{2.67}$$

The subproblem decomposition is given by

$$\begin{aligned} \min_{\mathbf{x}_1\in\mathbb{R}^2} \quad & [\mathbf{f}_1(\mathbf{x}_1)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \ddot{\mathbf{x}}_2) = \mathbf{0} \end{aligned} \tag{2.68}$$

and

$$\begin{aligned} \min_{\mathbf{x}_2\in\mathbb{R}^2} \quad & [\mathbf{f}_2(\mathbf{x}_2)] \\ \text{s.t.} \quad & \mathbf{h}(\ddot{\mathbf{x}}_1, \mathbf{x}_2) = \mathbf{0}. \end{aligned} \tag{2.69}$$

The application of OSDA to problem (2.67) is depicted in Fig. 2.6 for a given sample of weight vectors. Each plot (x_1^k, x_2^k), $k \geq 1$, depicts the result of the previous iteration's update of x_1, x_2, and $\mathbf{v}$. Subproblem (2.68) approximations to AiO efficient points for the AiO problem (2.67) are given by open diamonds, and analogous subproblem (2.69) approximations are given by open circles. In reviewing the plots of Fig. 2.6, satisfaction of the consistency constraints is realized for the later iterations k due to the updates of $\mathbf{v}$, which appears in the plots as the diamonds and circles overlapping. Furthermore, the subproblem approximations also converge to efficient points for the AiO problem (2.67); this appears in the plots as the diamonds and circles moving inside of the shaded gray region.

2.6.2 Nonconvex MOP

We define two nonlinear subproblems that will later be coordinated into one AiO problem. For subproblem 1, let $\mathbf{f}_1 : \mathbb{R}^2 \to \mathbb{R}^2$ be defined by

$$\mathbf{f}_1(\mathbf{x}_1) := \begin{bmatrix} -\frac{1}{3}\left({x_{1,1}}^2 - 6\,x_{1,1}x_{1,2} - 2\,{x_{1,2}}^2 - 6\right) \\ \frac{1}{20}\left(8\,{x_{1,1}}^2 + 46\,x_{1,1} + 76 - 14\,x_{1,2}x_{1,1} - 60\,x_{1,2} + 16\,{x_{1,2}}^2\right) \end{bmatrix}^T,$$

where $\mathbf{x}_1 := [x_{1,1}, x_{1,2}]$. For $X_1 := \{\mathbf{x}_1 \in \mathbb{R}^2 \,:\, 0 \leq x_{1,1}, x_{1,2} \leq 2\}$, define the MOP

$$\min_{\mathbf{x}_1 \in X_1} \quad \mathbf{f}_1(\mathbf{x}_1). \tag{2.70}$$

For subproblem 2, let $\mathbf{f}_2 : \mathbb{R}^2 \to \mathbb{R}^2$ be defined by

$$\mathbf{f}_2(\mathbf{x}_2) := \begin{bmatrix} 4\,{x_{2,1}}^2 + 10\,x_{2,1}x_{2,2} - 3\,{x_{2,2}}^2 + 4 \\ -5\,{x_{2,1}}^2 + 2\,x_{2,1}x_{2,2} + 2\,{x_{2,2}}^2 + 6 \end{bmatrix}^T,$$

where $\mathbf{x}_2 := [x_{2,1}, x_{2,2}]$. For $X_2 := \{\mathbf{x}_2 \in \mathbb{R}^2 \,:\, 0 \leq x_{2,1}, x_{2,2} \leq 2\}$, define the MOP

$$\min_{\mathbf{x}_2 \in X_2} \quad \mathbf{f}_2(\mathbf{x}_2). \tag{2.71}$$

Because subproblems (2.70) and (2.71) are both nonconvex, the quadratic scalarization is employed.

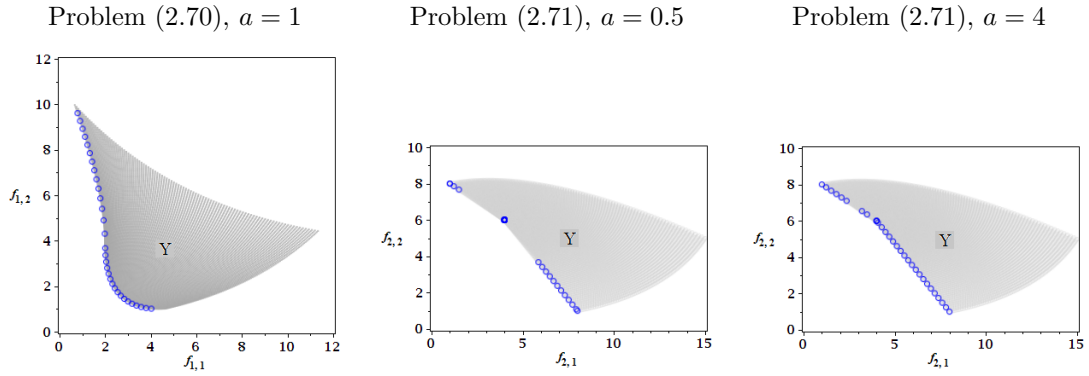

Figure 2.7: Applying the quadratic scalarization with $Q = (a/2)[-1,1]^T[-1,1]$, $\mathbf{q} = (\sqrt{2}/2)[1,1]^T$, varied $a > 0$, and a collection of reference points $\mathbf{y}^r$. Each shaded gray region depicts the set Y of feasible points in the subproblem-specific objective space, and circles are the computed Pareto outcomes.

The application of the quadratic scalarization to subproblems (2.70) and (2.71) is depicted in Fig. 2.7.

Next, we compare the use of weighted-sum scalarization and quadratic scalarization applied

to a combined, AiO problem

$$\begin{aligned} \min_{\mathbf{x}_1 \in X_1,\, \mathbf{x}_2 \in X_2} \quad & \mathbf{f}(\mathbf{x}_1, \mathbf{x}_2) := [\mathbf{f}_1(\mathbf{x}_1),\, \mathbf{f}_2(\mathbf{x}_2)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) := \begin{bmatrix} x_{1,1} - x_{2,1} \\ x_{1,2} - x_{2,2} \end{bmatrix} = \mathbf{0} \end{aligned} \tag{2.72}$$

where coupling between the subproblems (2.70) and (2.71) is embedded into problem (2.72) through the constraint $\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0}$. (Thus, problem (2.72) is of the form (2.2).) The efficient solutions generated with each type of scalarization are presented in the top two plots of Fig. 2.8. The top-left plot depicts the efficient solutions computed using the weighted-sum scalarization. The use of the quadratic scalarization in the top-right plot of Fig. 2.8 is obtained from the reformulation of problem (2.72) into a collection of problems of the form (2.18) with a fixed $\mathbf{z} = \mathbf{0}$, and multiple $\mathbf{w} > \mathbf{0}$ taken from a collection of weight vectors. In comparing the top two plots of Fig. 2.8, one sees that the use of the quadratic scalarization allows for the computation of efficient points that are not computable using the weighted-sum scalarization.

The bottom two plot of Fig. 2.8 illustrates the use of the quadratic scalarization for computing efficient points for problem (2.72) with the subproblem decomposition

$$\begin{aligned} \min_{\mathbf{x}_1 \in X_1} \quad & \mathbf{f}_1(\mathbf{x}_1) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \ddot{\mathbf{x}}_2) = \mathbf{0} \end{aligned} \tag{2.73}$$

$$\begin{aligned} \min_{\mathbf{x}_2 \in X_2} \quad & \mathbf{f}_2(\mathbf{x}_2) \\ \text{s.t.} \quad & \mathbf{h}(\ddot{\mathbf{x}}_1, \mathbf{x}_2) = \mathbf{0} \end{aligned} \tag{2.74}$$

using DSDA (left) and OSDA (right). (Thus, the subproblems (2.73) and (2.74) are of the form (2.3).) The consistency between the two subproblem solutions in each plot is evidenced by the diagonal crosses (subproblem (2.73) computations) being inside of the open circles (subproblem (2.74) computations). For reference, the bottom two plots also shows the same points displayed in the top-right plot as small gray dots.

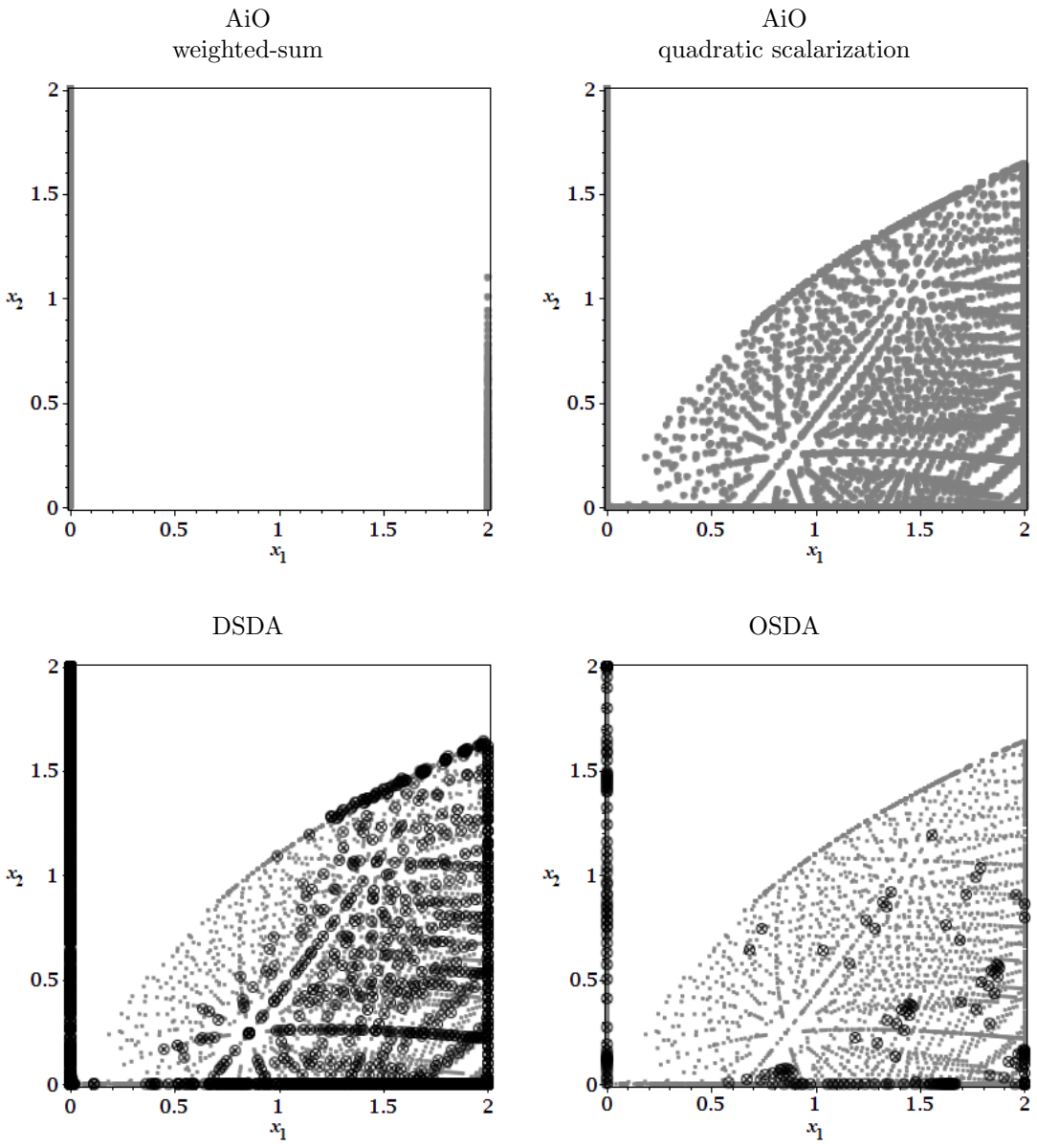

Figure 2.8: Efficient points for problem (2.72) obtained with the weighted-sum and quadratic scalarizations applied to the AiO problem (2.72) (top), and efficient point obtained using DSDA and OSDA applied to the subproblem decomposition (2.73) and (2.74) (bottom)

2.7 Conclusion

By refining and integrating previous analyses of scalarization methods, Lagrangian coordination methods, and the block coordinate descent (BCD) method, this paper sets a foundation for the development of two MODA algorithms, DSDA and OSDA, that compute sequences of approx-

imations to efficient solutions for an AiO MOP that is presented as a nonintegrable collection of subproblems. Conditions for the convergence of these sequences to solutions that are efficient for the AiO MOP, or at least satisfy relevant necessary conditions, are readily available from the theoretical developments of the algorithms' constituent parts. The application of DSDA and OSDA results in a nontrivial computation of efficient points in a coordinated and distributed manner for an example problem that is nonconvex, nonintegrated, and multiobjective.

In the way of future work, the following may be outlined. In the area of multiobjective scalarization, comparison of the use of quadratic scalarization with the use of weighted-t power method is needed. Furthermore, the (pseudo)convexifying effect of the quadratic scalarizations warrants investigation. Conditions on the objective functions and conditions on the quadratic scalarization need to be examined to gain insight. For Lagrangian coordination, a rate-of-convergence analysis and questions addressing the existence of limit points need to be addressed. For BCD, example problems can be developed and studied to provide insight into the new convergence results developed in Section 2.4. Furthermore, the convergence of ADMM for the multi-block $m \geq 3$ decomposition is an active area of research. The integration of BCD and Lagrangian coordination can be extended to address other assumptions about the inexact computation of BCD updates and method of multiplier updates; for example, the block coordinate $\mathbf{x}_i$ updates may be computed using approaches related to sequential quadratic programming [106]. Finally, the restrictions implied in the presentation of MOP (2.1) as the collection of nonintegrable subproblems (2.3) can be motivated from the object-oriented programming concepts in computer science, and the role of mathematical tools such as Lagrangian coordination and Gauss-Seidel decomposition can be explained according to these concepts.

Acknowledgments

The views presented in this work do not necessarily represent the views of our sponsors, the Automotive Research Center, a center of excellence of the US Army TACOM, and the National Science Foundation, grant number CMMI-1129969, whose support is greatly appreciated.

Chapter 3

Equitable Multiobjective Optimization Applied to the Design of a Hybrid Electric Vehicle Battery

[This chapter contains the contents of a paper titled "Equitable Multiobjective Optimization Applied to the Design of a Hybrid Electric Vehicle Battery"; authors are Brian Dandurand, Paolo Guarneri, Georges Fadel, and Margaret M. Wiecek; this paper is published in the *Journal of Mechanical Engineering*, Vol. 135, No. 4.]

3.1 Introduction

Due to the high energy density and high cell voltage, the Lithium-ion (Li-ion) battery is a promising technology for propulsion applications. However, the performance of Li-ion batteries worsens drastically at extreme temperatures (above 65 °C or below 0 °C) [2]. The thermal working conditions affect the performance as well as the life and the safety [5] of batteries. Temperature influences the electrochemical phenomena which determine the battery functioning. Performance

degradation in terms of power and capacity fade is observed in [85, 86] when charge/discharge cycles are repeated at high temperatures. Battery cells working at high temperatures may experience stability problem due to the exothermic chemical reactions. Such reactions will trigger other exothermic reactions and the positive feedback between the temperature and current may lead to battery explosion. An efficient heat rejection from the cells is of primary concern to avoid this undesirable situation. To achieve the desired voltage and current required for different applications, the cells are packed together in modules which in turn are connected in parallel or in series (see Fig. 3.1). Electrical unbalance among the cells limits the battery performance and reliability. The capacity of elements connected in series is limited by the element with the smallest capacity that will potentially experience the subsequent overcharging and premature failure. Uniform heat transfer and even temperature distribution are key issues to guarantee electrical balance and depend strongly on the cell packaging and layout [2] and on the thermal management strategies adopted.

The need for expensive and complex thermal management systems has in fact kept the Li-ion technology from becoming the first choice for Hybrid Electric Vehicle (HEV) and Electric Vehicles (EV) applications, however few studies systematically investigating thermal management strategies can be found. In [126], the battery pack of the Toyota Prius battery is tested and monitored under different conditions of driving cycles. The battery pack is cooled by the air conditioned drawn from the cabin. Air cooling systems have the advantage of the simplicity while they do not seem promising to achieve even temperature distributions due to the low convective heat exchange coefficients and low densities. The latter results in large air temperature gradients. To mitigate temperature unevenness, different layouts are adopted; in the Toyota Prius the air is divided into parallel flows to be distributed to the battery modules arranged at different distances from the air inlet while in the Honda Insight the batteries in the module are stacked in a single column [5]. Some cells are properly screened to balance the difference in heat exchange coefficients. Air distributors are used to increase the convective heat coefficient of the air flow near the outlet where the air reaches its maximum temperatures. In [78] a reciprocating air flow is studied to overcome the one-directional flow issue and improve evenness of temperatures. Beside air cooling, the usage of liquids is more suitable despite the increased complexity required of the battery due to the larger density and better heat exchange coefficients. In [65] the cells are cooled using plates as in the concept of cooling fuel cells in which the evenness of temperatures is a major issue. In particular, the topology of air channels in the plates is optimized considering the air pressure drop, the distribution of the plate

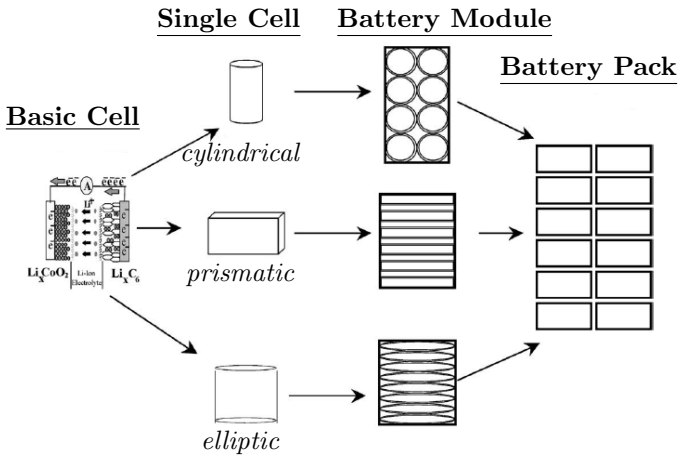

Figure 3.1: The battery layout

temperature and the average temperature. A different approach based on phase change material (PCM) is proposed in [1, 81, 98]. This design solution relies on the latent melting/solidification heat of the paraffin wax melting at about 40 °C (close to the optimal cell operating temperature). During the change of phase, the material is like a sink (or source) capable of exchanging heat while keeping a constant temperature.

Since the cell layout determines the heat exchange uniformity, the layout can be defined in order to ease the design of thermal management strategies. In literature, optimization has been used to solve layout/packaging problems. The problem is defined in terms of the positioning of some components in an available enclosure while satisfying geometrical constraints (i.e., no component overlap). In [80] the layout of a truck is optimized considering criteria such as compactness, vehicle dynamics and maintainability; in [104] the layout of mechanical components is defined through optimization; the layout of electronic components is optimized in [16] while including thermal constraints. In this study, the cell spacing inside of a battery module is determined to target the optimal operating cell temperature and evenness of temperature.

The goal of the presented research is the development of an optimization model and a methodology to find configurations or layouts of the cell pack of a Li-ion battery which results in optimal heat distribution of cells during operation. To this end, we represent the battery design problem as a multiobjective problem and solve it using a refinement of the Pareto optimality in the form of equitability (see [71, 72], for example). Pareto optimality can be refined in the equitability

sense when the criteria have common meaning and units, the criteria are anonymous or symmetrically indistinguishable from one another, and when the Pigou-Dalton principle of transfer, which implements a more even redistribution of the outcomes, results in improved outcomes. While the Pareto optimality has been widely used to solve engineering problems, the application of the equitable preference is new. It yields a smaller set than the set of Pareto efficient designs with specific equity properties that are desirable for the battery design problem.

The paper is structured as follows. Section 3.2 describes the problem of laying out the internal components of the Li-ion battery so that a desirable thermal outcome is realized during operation. This same section defines in more detail the concept of equitable optimality and works toward developing a model formulation whose Pareto efficient designs are the equitable efficient designs to the battery problem. Section 3.3 describes the optimization environment, the computation of temperatures through simulation, and the computation of efficient battery designs based on the models given in Section 3.2. Section 3.4 concludes the paper by summarizing what has been accomplished and outlining the future work of refining the existing methodology to coordinate the vehicle-level design problem with the battery-level design problem.

3.2 Development of the optimization model

In designing the internal layout of battery cells, thermal considerations are important. In particular, good layout designs give cell temperatures that are close to a given target (40 °C) while maintaining evenness of temperature distribution.

The thermal distribution of the cells depends on properties such as the shape of the cells, the size of the cells, the configuration of cell layout, and the spacing of the cells. The cells are assumed to be arranged in a triangular lattice. The diagram in Fig. 3.2 illustrates the appearance of a possible internal design.

Let p_{cell} be a unitless scalar that determines the uniform distance of adjacent cells from one another. A value of $p_{cell} = 1$ indicates that the centers of two adjacent cells are one cell diameter distant from one another (i.e., the cells touch one another). This is not permissible and so $p_{cell} > 1$ is required. In general, p_{cell} determines the spacing of adjacent cells via the relation $d_{center} = p_{cell} \cdot d_{diameter}$ where d_{center} denotes the distance between the centers of two adjacent cells, and $d_{diameter}$ denotes some measure of the cell size. In the case of circular cells, $d_{diameter}$

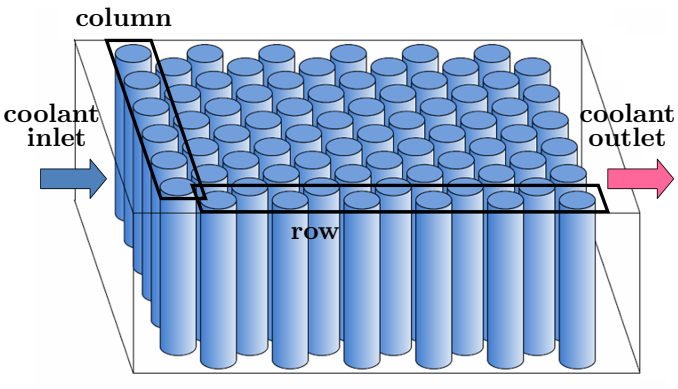

Figure 3.2: Illustration of geometric layout of battery components (12 columns, 6 rows)

corresponds to the diameter of the cell. The scalar p_{cell} is also referred to as the *spacing factor*.

The design problem in terms of maintaining a target operational temperature T_0 for each battery cell is a multiobjective optimization problem containing 72 objective functions corresponding to the 72 battery cells. Due to the restrictions of the current thermal model, battery cells are aggregated into columns. Each column has an equal number of battery cells. The thermal model is designed to simulate one representative temperature for each column. Let the columns be indexed by $k \in K$. Denote by T_k the temperature generated by the thermal model for column k.

The squared temperature deviation D_k from the target temperature T_0 associated with column k is obtained as the following (squared) difference

$$D_k := (T_k - T_0)^2 \quad \text{for each } k \in K. \tag{3.1}$$

The temperature T_k for each $k \in K$ depends on the design variable $\mathbf{x}$ defined as $\mathbf{x} = [p_{cell}, \mathbf{s}]$. The scalar variable p_{cell} determines spacing of adjacent cells while the vector variable $\mathbf{s}$ denotes other shape/layout parameters shared at the vehicle level. The design variable $\mathbf{x}$ is to be taken from an unspecified feasible set X. Consequently, the squared difference D_k for each $k \in K$ also depends on $\mathbf{x}$ by the definition given in (3.1). The problem of minimizing column temperature squared deviations becomes a multiobjective optimization problem and assumes the following form.

$$\min_{\mathbf{x} \in X} [D_1(\mathbf{x}), \ldots, D_m(\mathbf{x})] \tag{3.2}$$

The concept of Pareto optimality, being a standard way to approach multiobjective optimization problems, can be applied to find the set of efficient battery designs for problem (3.2). A battery design $\mathbf{x} \in X$ is said to be *Pareto efficient* provided any other design that improves the temperature deviation of one column causes the deterioration of the temperature deviation of at least one other column. A *Pareto optimal outcome* refers to the vector of temperature deviations associated with the columns of battery cells corresponding to a Pareto efficient design.

3.2.1 Model in terms of spacing

In formulation (3.2), the design variables determine such properties as the shape of the battery cells, the geometric layout of the cells, e.g., how many columns the cells are grouped into, and how far apart adjacent cells are spaced. The following optimization problem is formulated considering the spacing p_{cell} as a design variable while any values associated with shaping parameter $\mathbf{s}$ are fixed. Under this assumption, problem (3.2) becomes

$$\min_{p_{cell} \in (1.0, 2.0]} [D_1(p_{cell}), \ldots, D_m(p_{cell})] \tag{3.3}$$

As stated earlier, the lower bound $p_{cell} > 1$ is necessary because the battery cells cannot touch one another. The upper bound $p_{cell} \leq 2$ is set because of the restricted availability of space within the battery pack.

As with problem (3.2), the designs for which the improvement of temperature deviation of one column results in a degraded temperature deviation for at least one other column are referred to as the *Pareto efficient designs* for problem (3.3), and the corresponding temperature deviations are referred to as *Pareto optimal outcomes*.

There are two traditional ways to solve the multiobjective problem (3.3). One is to convert it into a single-objective problem, whose optimal solution will also be a solution to problem (3.3), and the other is to approximate the set of Pareto efficient designs of problem (3.3) and then select a preferred Pareto efficient design [34]. In the first approach, the type of scalarization will heavily affect the optimal solution found. In the other, the choice of a preferred Pareto efficient design may be cumbersome if the approximation contains a large number of points. Making use of the battery design context, the minimization of the highest squared temperature deviation, i.e., the min-max formulation, or the minimization of the sum of the squared temperature deviations, i.e., the min-

sum formulation, would be reasonable scalarizations of problem (3.3). However, this straightforward observation can be taken much further for the battery design. First, optimal solutions to the min-max problem or the min-sum problem are, under certain conditions, elements of a bigger set known as the set of equitable efficient designs. Second, the battery design problem inherently satisfies the properties required by the equitable efficiency to be the preference defining the solution set of a multiobjective optimization problem.

To benefit from the special properties of the battery design problem and simultaneously generalize the min-max scalarization we propose a new solution approach to the multiobjective design problem (3.3) based on the concept of equitable efficiency. To explain this concept, we first present the special properties of the objective functions in (3.2) and (3.3).

3.2.2 Equitable preference

The multiobjective problems (3.2) and (3.3) are of special form due to three properties of the objective functions modeling optimal temperature of the cells.

1. Each objective function represents the same physical property (temperature) and hence assume values that are measured on a common scale. The objective functions are therefore *comparable*.

2. The distribution of temperature among the cells is important while the assignment of the specific temperature to a specific cell is not, which means that the cells remain anonymous within the module and, consequently, the objective functions are *anonymous* (*impartial*). This is demonstrated in Fig. 3.3.

3. Evenly distributed temperature deviations are preferred. That is, the cell temperatures satisfy the *Pigou-Dalton principle of transfers*: given two distinct cell columns indexed with $k_1, k_2 \in K$ with temperature deviations D_{k_1} and D_{k_2}, if $D_{k_1} > D_{k_2}$, then the battery design with the temperature deviations $[D_1, \ldots, D_{k_1} - \Delta, \ldots, D_{k_2} + \Delta, \ldots, D_m]$, where $0 < \Delta < D_{k_1} - D_{k_2}$, is preferred to the battery design with the temperature deviations $[D_1, \ldots, D_{k_1}, \ldots, D_{k_2}, \ldots, D_m]$. This is demonstrated in Fig. 3.4.

The three properties: comparability, anonymity, and the Pigou-Dalton principle of transfers are characteristic for the *equitable preference* ([71, 72]) which is a refinement of the Pareto preference. According to the latter, a design is Pareto efficient if no objective function can be improved without

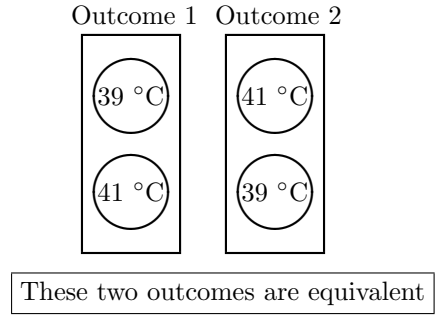

Figure 3.3: Illustrating anonymity of objectives

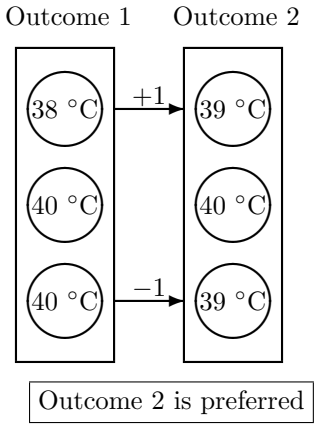

Figure 3.4: Illustrating Pigou-Dalton principle of transfers

deteriorating another one. However, there are situations in many areas of human activity where an outcome whose objective values are more evenly distributed is considered better than another outcome whose objective values are less evenly distributed, even if these two outcomes are non-comparable in the Pareto sense. The Pigou-Dalton principle postulates that a transfer from a higher-valued objective value of an outcome to a lower-valued objective value yields a better outcome.

The design of batteries creates an opportunity to introduce the equitable preference into engineering design and investigate its significance. Due to mathematical properties, the set of equitable designs is always a subset of the set of Pareto efficient designs and so equitable designs are such Pareto efficient designs that preserve equity among the criteria which is understood in the context of the three properties. The reduction of the set of optimal designs is desirable since it is known that the Pareto preference does not restrict the available design choices sufficiently as the number of objective functions increases [26].

3.2.3 Obtaining equitable temperature deviations

In this section the methodology for obtaining equitable efficient designs is first presented on a general multiobjective optimization problem and then applied to the battery design optimization problem (3.3). Let $\mathbf{f} := [f_1, \ldots, f_m]$ be the vector of objective functions. Using the results in [72] and other references therein, the multiobjective problem

$$\min_{\mathbf{x} \in X} \ \mathbf{f}(\mathbf{x}) \tag{3.4}$$

is reformulated into another multiobjective problem whose Pareto efficient solutions are the equitable efficient solutions of (3.4). To this end, define the ordering operator $\theta_q(\mathbf{f}(\mathbf{x})), q = 1, \cdots, m$, to return the q^{th} largest component of $\mathbf{f}$. Consequently $\theta_1(\mathbf{f}(\mathbf{x})) \geq \cdots \geq \theta_m(\mathbf{f}(\mathbf{x}))$ and so $[\theta_1(\mathbf{f}(\mathbf{x})), \ldots, \theta_m(\mathbf{f}(\mathbf{x}))]$ is simply a reordering of the components of $\mathbf{f}(\mathbf{x})$ from largest to smallest. Next, define the sum of the q largest components

$$\overline{\theta}_q := \sum_{i=1}^{q} \theta_i(\mathbf{f}(\mathbf{x}))$$

for $q = 1, \cdots, m$ and formulate the multiobjective problem

$$\min_{\mathbf{x} \in X} \ [\overline{\theta}_1(\mathbf{f}(\mathbf{x})), \ldots, \overline{\theta}_m(\mathbf{f}(\mathbf{x}))]. \tag{3.5}$$

Corollary 1 of [72] states that a feasible design of problem (3.4) is equitable efficient for problem (3.4) if and only if it is Pareto efficient for problem (3.5). Thus, a solvable formulation for problem (3.5) can be used to obtain the equitable efficient designs of problem (3.4).

The value $\overline{\theta}_q(\mathbf{f}(\mathbf{x}))$ is known to coincide with the optimal objective value of the following

linear program

$$\begin{gathered}\min_{z_q,\, t_q,\, d_{q,k}} \; z_q \\ \text{subject to} \\ z_q = q\, t_q + \textstyle\sum_{k=1}^{m} d_{q,k} \\ t_q + d_{q,k} \geq f_k(\mathbf{x}), \quad d_{q,k} \geq 0, \quad \text{for} \quad k = 1, \ldots, m\end{gathered} \tag{3.6}$$

where t_q, $d_{q,k}$, and z_q are new auxiliary variables aiding in the computation of $\overline{\theta}_q(\mathbf{f}(\mathbf{x}))$. The variables t_q, $d_{q,k}$, and z_q have no meaning in the context of problem (3.4). Note that $\mathbf{x}$ is a fixed constant in problem (3.6). The construction of linear program (3.6) and the fact that its minimum objective value is $\overline{\theta}_k$ is developed and proven in [117].

Using the optimal objective value of problem (3.6) to evaluate $\overline{\theta}_q(\mathbf{f}(\mathbf{x}))$ of problem (3.5) for $q = 1, \ldots, m$, problem (3.7) may be stated as an equivalent reformulation of problem (3.5) [72].

$$\begin{gathered}\min_{\mathbf{x},\, z_q,\, t_q,\, d_{q,k}} \; [z_1, \ldots, z_m] \\ \text{subject to} \\ z_q = q t_q + \textstyle\sum_{k=1}^{m} d_{q,k}, \quad t_q \;\text{ free}, \quad q = 1, \ldots, m, \\ t_q + d_{qk} - f_k(\mathbf{x}) \geq 0, \quad d_{q,k} \geq 0, \quad q, k = 1, \ldots, m \\ \mathbf{x} \in X\end{gathered} \tag{3.7}$$

Problem (3.7) with the identification $f_k = D_k$, $k = 1, \ldots, m$, $\mathbf{x} = p_{cell}$, and $X = (1.0, 2.0]$ can therefore be used to compute equitable efficient designs of problem (3.3).

As an application of this theory, we revisit the mentioned earlier min-max formulation of

the battery design problem (3.2) given as

$$\min_{p_{cell}\in(1.0,2.0]} \max_{k\in K}\{D_k(p_{cell})\} \tag{3.8}$$

With the identifications $f_k = D_k$, $k = 1, \ldots, m$, $\mathbf{x} = p_{cell}$, and $X = (1.0, 2.0]$ applied to problem (3.5), one may see that an optimal design of (3.8) corresponds to the single-objective optimization problem of minimizing $\overline{\theta}_1$ since $\overline{\theta}_1$ is by definition the maximum value of the components $D_k, k \in K$. In this manner, one sees that when an optimal design of (3.8) is unique, it is also Pareto efficient for problem (3.5) (with $f_k = D_k$, $k = 1, \ldots, m$, $\mathbf{x} = p_{cell}$ and $X = (1.0, 2.0]$) [100] and thus equitable for problem (3.3).

Similarly, the min-sum formulation of the battery design problem (3.2) given as

$$\min_{p_{cell}\in(1.0,2.0]} \sum_{k\in K} D_k(p_{cell}) \tag{3.9}$$

is shown to be a special case of problem (3.5). Again, with the identifications $f_k = D_k$, $k = 1, \ldots, m$, $\mathbf{x} = p_{cell}$, and $X = (1.0, 2.0]$ applied to problem (3.5), one may see that an optimal design of (3.9) corresponds to the single-objective optimization problem of minimizing $\overline{\theta}_m$ since, by definition, $\overline{\theta}_m = \sum_{k\in K} D_k$. Therefore, when the optimal design of (3.9) is uniquely obtained, it is a Pareto efficient design of (3.5) (with $f_k = D_k$, $\mathbf{x} = p_{cell}$ and $X = (1.0, 2.0]$) and thus equitable for (3.3).

Due to the relative ease of formulating a min-max or min-sum problem, the formulation of either problem (3.8) or (3.9) is preferred to the formulation (via (3.7)) of problem (3.5) when one equitable design is preferred to the range of equitable designs available by solving problem (3.5). However, the battery layout design problem may benefit from the available range of equitable designs obtainable by solving problem (3.5). The battery layout design is subject to some feasibility constraints related to the arrangement of the battery itself and other components inside of the vehicle. Determination of optimal vehicle layouts is an optimization problem in which position of the battery is one of the design variables. In this context, the battery is also a morphable component because its aspect ratio is optimized to improve its performance. The optimal battery aspect ratio obtained by either the min-max or min-sum formulation may be in conflict with some geometric constraints and space availability at the vehicle level. With the availability of a range of equitable designs, the designer has the possibility to choose a battery layout that is more suitable for the vehicle level

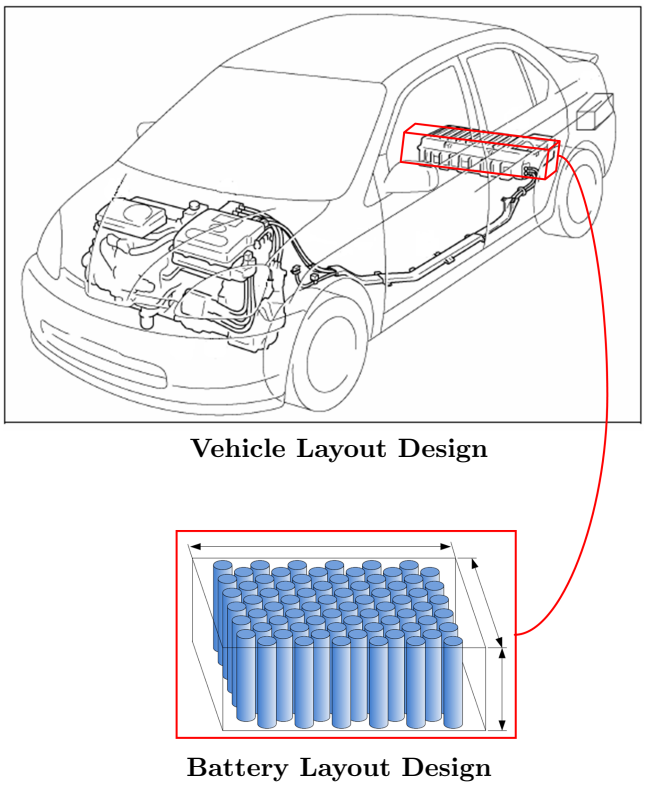

Figure 3.5: The relationship between the layout designs at the battery and vehicle levels

layout. The relationship between the layout designs at the battery and vehicle levels is represented in Fig. 3.5.

3.3 Optimization

In this section, the results of the implementation of both problems (3.3) and (3.7) using MATLAB [109] interfaced with a SIMULINK [110] model are presented.

The temperature functions T_k for each column $k \in K$ are not given in closed form and each function evaluation T_k at a given spacing factor value of p_{cell} must be computed through simulation using a lumped parameter model[1].

The model simulates steady state conditions with uniform heat generation in each cell and coolant flow crossing the battery pack from one side to the other. The heat exchange coefficient is

[1]See Acknowledgments.

estimated through classic numerical correlations involving Nusselt, Reynolds and Prandtl dimensionless numbers. The flow is considered 1D and the cells in the same column have the same temperature as mentioned in Section 3.2. At the current modeling stage, the design variables available to define the layout determine the cell spacing and the number of cell columns used. The model is sufficiently simple to keep the computational time reasonable yet meaningful to explore different packaging solutions and test the proposed optimization based on the equitability.

3.3.1 Convexity of temperature deviations in terms of p_{cell}

The temperature values T_k, $k \in K$, generated with the SIMULINK model in terms of p_{cell} are used to form the column temperature deviations D_k, $k \in K$, according to formula (3.1). Figure 3.6 depicts the absolute values of the column temperature deviations obtained from the simulated column temperatures. There are 72 cells grouped into 12 columns. For each column, the temperature deviation follows a convex trend as p_{cell} varies.

From Fig. 3.6 and Tab. 3.1, one can see the conflict among the twelve objective functions as they reach their respective minima at different values of p_{cell}. This conflicting behavior of the objective functions (i.e., the deviations from the target temperature) reflects the thermodynamics that is simulated in the model. Even if the battery layout is symmetric, the thermal conditions and the consequent temperature distribution are not. The coolant flows, with a given mass flow rate, from one side to the other increasing its temperature. Since the total heat rejected Q, the coolant mass flow rate $m_{coolant}$ and the coolant heat capacity $c_{coolant}$ are constant parameters in the model, the coolant temperature rise is not a function of the cell arrangement.

$$T_{out} - T_{in} = \frac{Q}{m_{coolant} \cdot c_{coolant}}$$

As qualitatively shown in Fig. 3.7, the only difference is the gradient of the coolant temperature and the cell-to-coolant heat exchange coefficient.

The conflict and impossibility to achieve uniform temperature can be explained using the following equation.

$$T_{cell} - T_{coolant} = \Delta T = \frac{Q}{hS}, \tag{3.10}$$

where Q is the thermal power to be rejected, S is the cell outer surface and h is the convective heat

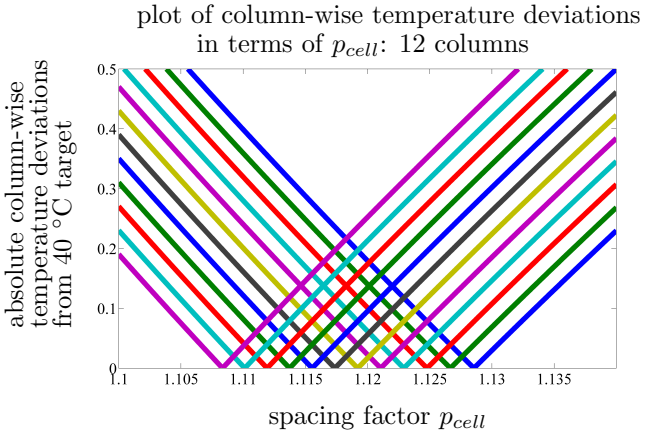

Figure 3.6: Evaluating convexity of absolute temperature deviations in terms of spacing factor p_{cell}.

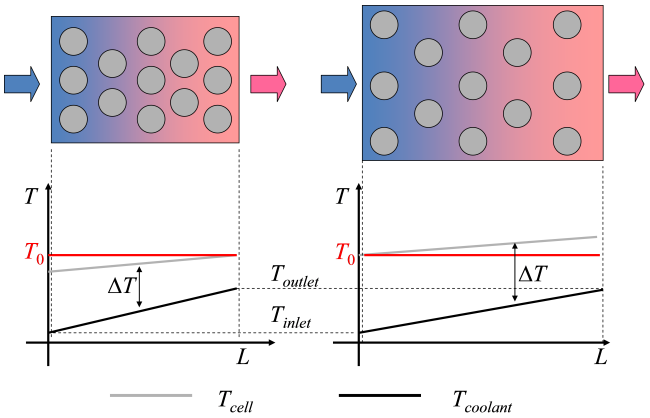

Figure 3.7: Cell coolant temperature drop (black line is coolant temperature, gray line is cell temperature)

transfer coefficient. Since Q and S are constant parameters, given the coolant temperature $T_{coolant}$, the cell temperature is determined by the coefficient h. More compact arrangements determine, for a given coolant mass flow rate, higher flow velocities and, therefore, higher h. The temperature drop ΔT is lower allowing the cell close to the outlet to work at the ideal temperature T_0 (left-hand side of Fig. 3.7). On the other hand, increasing the cell spacing reduces the flow speed and the transfer coefficient h. Due to higher temperature drop ΔT, the cells close to the inlet surrounded by cold coolant can reach the ideal condition T_0 (right-hand side of Fig. 3.7).

Table 3.1: Individual design minima for the twelve columns

Column	Optimal p_{cell}	Column	Optimal p_{cell}
1	1.1286	7	1.1174
2	1.1267	8	1.1155
3	1.1248	9	1.1137
4	1.1229	10	1.1119
5	1.1211	11	1.1101
6	1.1192	12	1.1084

3.3.2 Applying the weighted-sum scalarization

The weighted-sum scalarization is used to convert the multiobjective problem (3.4) into the single objective optimization problem

$$\min_{\mathbf{x}\in X} \sum_{k=1}^{m} w_k \, f_k(\mathbf{x}), \tag{3.11}$$

where for each k, $w_k \geq 0$, and for at least one k, $w_k > 0$. It is well-known that whenever the objective functions of a multiobjective problem are convex over the feasible region that is also convex, then an optimal solution to the weighted-sum scalarization (3.11) is a weak Pareto efficient design of problem (3.4) [34]. That is, to each weak Pareto efficient design, there corresponds at least one nonnegative $\mathbf{w} = [w_1, \ldots, w_m]$. *Weak Pareto efficiency* is a generalization of Pareto efficiency, where designs that are alternatives to a weak Pareto efficient design may result in *no change* in at least one criterion while all other criteria are improved. When the weighted-sum scalarization (3.11) has a unique optimal solution, or when strictly positive weights are used, then the optimal solution of (3.11) is a Pareto efficient solution of (3.4) [34]. In comparison to weak Pareto efficiency, *Pareto efficiency* requires alternate designs to result in deterioration of at least one criterion when at least one other criterion is improved. Thus, under the assumption that the temperature deviation functions D_k are convex as suggested in Fig. 3.6, the weighted-sum scalarization with nonnegative weights is sufficient for the computation of any weak Pareto efficient solution of (3.3). The convexity of the constraints of (3.7) is inherited from the convexity of the objectives of (3.3), and so the weighted-sum scalarization is sufficient to generate any weak Pareto efficient solution of (3.7) and thus also of (3.5).

3.3.3 Constructing samples of weight vectors

The computation of efficient battery designs depends on the sample of weight vectors that are applied to problems (3.3) and (3.7). Three problem specific schemes are suggested below for constructing these weight vector samples.

Scheme 1: For problem (3.3), the following scheme is motivated by the observation that physically close cell columns tend to have similar temperatures, and the conflict among the objectives based on these temperatures is minimal. For this reason, most of the computational effort is allocated to generating diverse weights corresponding to a representative subset of cell columns that are physically separate and maximally conflicting. To each of these representative cell columns k, a positive weight w_k is assigned. Then, a (near) zero weight is assigned to each cell column that is not indicated as representative. (Such non-representative cell columns should be physically close to a representative cell column.) One possible implementation of this scheme is given below.

6 columns : Define $\mathbf{w}$ so that w_k is positive for $k = 1, 3, 4, 6$. Set $w_k = 0$ for $k \neq 1, 3, 4, 6$.

9 columns : Define $\mathbf{w}$ so that w_k is positive for $k = 1, 4, 6, 9$. Set $w_k = 0$ for $k \neq 1, 4, 6, 9$.

12 columns : Define $\mathbf{w}$ so that w_k is positive for $k = 1, 5, 8, 12$. Set $w_k = 0$ for $k \neq 1, 5, 8, 12$.

18 columns : Define $\mathbf{w}$ so that w_k is positive for $k = 1, 7, 12, 18$. Set $w_k = 0$ for $k \neq 1, 7, 12, 18$.

In order to guarantee that designs generated with Scheme 1 are Pareto efficient for problem (3.4), the zero-valued components of each weight vector are reassigned with positive values that are small relative to the other nonzero components.

Scheme 2: For problem (3.7), an approach similar to Scheme 1 is employed with a slight modification. Objective values z_q with consecutive indices q will have similar values analogous to the similar temperature values of physically close battery cell columns. For each $q < m$, the difference $\overline{\theta}_{q+1} - \overline{\theta}_q > 0$ is large when q is close to one, and small when q is close to m. This follows from the definition of $\overline{\theta}$. Thus, conflict among consecutive z_q values will be most prominent for z_q values where q close to one. This motivates an allocation of nonzero w_q weight values that is biased more heavily toward indices q closer to one and less heavily toward indices q closer to m, as demonstrated with the following implementation of this scheme.

6 columns : Define $\mathbf{w}$ so that w_q is positive for $q = 1, 2, 3, 6$. Set $w_q = 0$ for $q \neq 1, 2, 3, 6$.

9 columns : Define $\mathbf{w}$ so that w_q is positive for $q = 1, 2, 4, 9$. Set $w_q = 0$ for $q \neq 1, 2, 4, 9$.

12 columns : Define $\mathbf{w}$ so that w_q is positive for $q = 1, 3, 6, 12$. Set $w_q = 0$ for $q \neq 1, 3, 6, 12$.

18 columns : Define $\mathbf{w}$ so that w_q is positive for $q = 1, 4, 9, 18$. Set $w_q = 0$ for $q \neq 1, 4, 9, 18$.

In order to guarantee that the designs generated with Scheme 2 are Pareto efficient for problem (3.7) and thus equitable efficient for problem (3.4), the zero-valued components of each weight vector are reassigned with positive values that are small relative to the other nonzero components.

When it is desirable to compute only a few equitable efficient designs for problem (3.4), then a scheme of the following form may be useful.

Scheme 3: The application of these three weight vectors to problem (3.7) is intended to supply a small but diverse sample of equitable efficient designs to problem (3.4).

1. $\mathbf{w} = [100, 1, 0.1, 0.01, \ldots, 10^{-(m-2)}]$ is applied to (3.7) to approximate a min-max reformulation of problem (3.4).

2. $\mathbf{w} = [1, 1, \ldots, 1]$ is applied to (3.7) to obtain an equitable efficient design to problem (3.4) that is intermediate to any design resulting from either a min-max or min-sum reformulation of (3.4).

3. $\mathbf{w} = [10^{-(m-2)}, 10^{-(m-1)}, \ldots, 0.1, 1, 100]$ is applied to (3.7) to approximate a min-sum reformulation of problem (3.4).

3.3.4 Results

We consider four scenarios with the numbers of columns $m = 6, 9, 12, 18$. Therefore, the multiobjective problems solved have between 6 and 18 objective functions. Computations are performed for problems (3.3) and (3.7). For both the Pareto efficient designs resulting from (3.3) and the equitable efficient designs resulting from (3.7), the corresponding outcomes are taken as the temperature deviations D_k (rather than z_q in the equitable case since these values have no meaning in the design problem).

The results obtained for the four scenarios are reported in Tabs. 3.2-3.5 and Fig. 3.8. All computed designs in Tabs. 3.2-3.5 are Pareto efficient, while some of them are additionally equitable. The left-most columns of the Tabs. 3.2-3.5 report these designs, while the right-most columns indicate

Table 3.2: Pareto efficient battery designs and condensed Pareto optimal temperature deviations (6 columns, 12 rows)

$\mathbf{p_{cell}}$	$D_{1:2}$	$D_{3:4}$	$D_{5:6}$	**Classification**
1.11270	0.5868	0.4316	0.1988	**Pareto**
1.11568	0.4573	0.2270	0.3257	**Pareto**
1.11683	0.3968	0.1984	0.3968	**min-max**
1.11683	0.3967	0.1984	0.3969	**equitable**
1.11683	0.3967	0.1984	0.3968	**equitable**
1.11684	0.3964	0.1984	0.3972	**equitable**
1.11684	0.3963	0.1984	0.3972	**min-sum**
1.11800	0.3245	0.2280	0.4575	**Pareto**
1.12101	0.1980	0.4302	0.5847	**Pareto**

Table 3.3: Pareto efficient battery designs and condensed Pareto optimal temperature deviations (9 columns, 8 rows)

$\mathbf{p_{cell}}$	$D_{1:3}$	$D_{4:6}$	$D_{7:9}$	**Classification**
1.11215	0.8891	0.6594	0.3247	**Pareto**
1.11694	0.6870	0.3435	0.4863	**Pareto**
1.11874	0.5947	0.2803	0.5947	**min-max**
1.11874	0.5947	0.2804	0.5946	**equitable**
1.11874	0.5947	0.2804	0.5946	**equitable**
1.11876	0.5936	0.2811	0.5957	**equitable**
1.11876	0.5936	0.2811	0.5957	**min-sum**
1.12060	0.4831	0.3459	0.6875	**Pareto**
1.12548	0.3231	0.6560	0.8842	**Pareto**

their classification. For reference, these same results associated with the min-max problem (3.8) and the min-sum problem (3.9) corresponding to problem (3.3) are also given.

The designs reported in Tabs. 3.2-3.5 and Fig. 3.8 that are Pareto efficient but not equitable efficient for problem (3.3) are computed with the use of weight vectors that are obtained using Scheme 1 of Section 3.3.3. These weight vectors are applied to problem (3.3). Due to the narrow range of points that are equitable efficient for problem (3.3), the equitable efficient points reported in Tabs. 3.2-3.5 and Fig. 3.8 are those computed with the use of weight vectors obtained using Scheme 3 of Section 3.3.3. These weight vectors are applied to problem (3.7) with $f_k = D_k$, $k = 1, \ldots, m$, $\mathbf{x} = p_{cell}$ and $X = (1.0, 2.0]$.

The remaining columns of Tabs. 3.2-3.5 report the objective values of the multiobjective problems solved, but condensed into 3-dimensional vectors $A := [A_1, A_2, A_3]$. This reduction of the objective space is done solely for reporting purposes *after* the computation of the Pareto and

Table 3.4: Pareto efficient battery designs and condensed Pareto optimal temperature deviations (12 columns, 6 rows)

$\mathbf{p_{cell}}$	$D_{1:4}$	$D_{5:8}$	$D_{9:12}$	**Classification**
1.10941	1.1880	0.8816	0.4365	**Pareto**
1.11587	0.9160	0.4717	0.6493	**Pareto**
1.11829	0.7931	0.3965	0.7931	**min-max**
1.11829	0.7930	0.3965	0.7931	**equitable**
1.11829	0.7930	0.3965	0.7931	**equitable**
1.11829	0.7931	0.3965	0.7930	**equitable**
1.11833	0.7911	0.3965	0.7950	**min-sum**
1.12080	0.6433	0.4738	0.9172	**Pareto**
1.12745	0.4342	0.8758	1.1793	**Pareto**

Table 3.5: Pareto efficient battery designs and condensed Pareto optimal temperature deviations (18 columns, 4 rows)

$\mathbf{p_{cell}}$	$D_{1:6}$	$D_{7:12}$	$D_{13:18}$	**Classification**
1.10402	1.7746	1.3098	0.6424	**Pareto**
1.11364	1.3689	0.7066	0.9840	**Pareto**
1.11721	1.1902	0.5951	1.1902	**min-max**
1.11721	1.1901	0.5951	1.1902	**equitable**
1.11721	1.1901	0.5951	1.1902	**equitable**
1.11729	1.1856	0.5951	1.1946	**equitable**
1.11729	1.1856	0.5951	1.1946	**min-sum**
1.12103	0.9671	0.7167	1.3741	**Pareto**
1.13105	0.6374	1.2981	1.7562	**Pareto**

equitable efficient designs. For $0 < i < j \leq m$ integers, define

$$D_{i:j}(p_{cell}) := \left\| \left[\sqrt{D_i},\ \sqrt{D_{i+1}},\ \ldots,\ \sqrt{D_{j-1}},\ \sqrt{D_j} \right] \right\|_2$$

where $\|\cdot\|_2$ is the Euclidean 2-norm. Taking the square root of each D_k component in the definition of $D_{i:j}$ turns the squared deviation D_k into the absolute deviation $\sqrt{D_k} = |T_k - T_0|$. The following condensed outcomes are given.

$m = 6$: Outcomes are reported as $A = [D_{1:2}, D_{3:4}, D_{5:6}]$.

$m = 9$: Outcomes are reported as $A = [D_{1:3}, D_{4:6}, D_{7:9}]$.

$m = 12$: Outcomes are reported as $A = [D_{1:4}, D_{5:8}, D_{9:12}]$.

$m = 18$: Outcomes are reported as $A = [D_{1:6}, D_{7:12}, D_{13:18}]$.

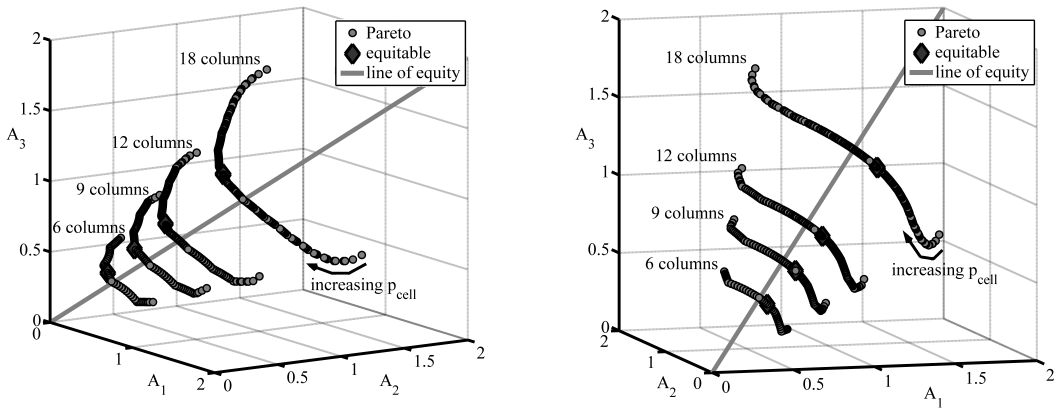

Figure 3.8: Condensed Pareto optimal and equitable outcomes with line of equity (two views)

From Tabs. 3.2-3.5 it is seen that the computed equitable designs are contained between the min-max and the min-sum designs as expected. In each of these tables, the values for p_{cell} are given to five decimal places to reveal the presence of a narrow equitable range. This amount of precision otherwise has no practical value.

Figure 3.8 depicts the condensed Pareto optimal outcomes together with the condensed equitable outcomes for all scenarios. These outcomes are plotted against the line of equity which denotes where the three components of the condensed outcomes are equal to one another.

In general, the Pareto and equitable efficient design agree with the results presented in Tab. 3.1. Due to the thermal conditions, it is not possible to achieve the ideal target temperature for all the cells as explained in Section 3.3.1. The cell spacing that is optimal for the cells close to the inlet is not optimal for the cells far from it and vice versa. The equitable efficient solutions represent the designs in between the two designs that are depicted in Fig 3.7. The central columns of Tabs. 3.2-3.5 show that the temperature deviations D_k are functions of the cells spacing. It is worth noticing that for some Pareto efficient designs which are not equitable, a temperature unevenness of about 1°C is observed, while a 5°C unevenness may be detrimental for the battery [78].

The effect of the aspect ratio is evident from the plots in Fig. 3.8 and the data from Tabs. 3.2-3.5. When the constraints given from the vehicle level do not restrict the feasible choices for the number of columns, the use of fewer columns clearly result in better sets of Pareto efficient designs due to their smaller ranges and a location closer to the origin of the objective space.

3.4 Conclusion

This paper outlines a methodology for finding optimal designs for the battery layout. The battery cell temperatures need to meet two criteria: (i) closeness to target temperature, (ii) evenness of temperature distribution. Because of these two needs, designs are sought that are not only Pareto efficient (to address the first criterion) but are also efficient with respect to the equitable preference relation. In effect, this work establishes a methodology based on the theory developed in the optimization literature to obtain the equitable designs computationally.

In the current approach, only spacing of adjacent cells is varied. As improved battery models become available, and as this battery design problem is interfaced with the design problem at the upper (underhood) level, more design variables such as the shape of battery pack and the geometric arrangement of cells will be considered. The methodology developed in this work will be adapted for this purpose.

Acknowledgments

We thank Dr. Dohoy Young and his team from the University of Michigan Dearborn who supplied us with the thermal models for the batteries. The views presented in this work do not necessarily represent the views of our sponsors, the Automotive Research Center, a center of excellence of the US Army TACOM, and the National Science Foundation, grant number CMMI-1129969, whose support is greatly appreciated.

Chapter 4

Bilevel multiobjective packaging optimization for automotive design

[This chapter contains the contents of a paper submitted for review in the journal: *Structural and Multidisciplinary Optimization* on May 7th, 2013 titled "Bilevel multiobjective packaging optimization for automotive design"; the authors are Brian Dandurand, Paolo Guarneri, Georges Fadel, and Margaret M. Wiecek.]

4.1 Introduction

Multiobjective optimization problems (MOPs) having the integrated form

$$\begin{aligned} \min \quad & \mathbf{f}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in X, \end{aligned} \tag{4.1}$$

underlie many engineering design problems, where $\mathbf{f} = (f_1, \ldots, f_p)$ is a vector of objective functions and $\mathbf{x}$ is a vector of design variables taken from a feasible set $X \subseteq \mathbb{R}^n$. Due to a conflict among the objective functions, there typically is no $\mathbf{x} \in X$ minimizing every objective function simultaneously. Thus, the concept of optimality is provided by Pareto efficiency. A solution $\mathbf{x}^* \in X$ is *weakly (Pareto) efficient* for problem (4.1) if there does *not* exist any other $\mathbf{x} \in X$ for which $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ for all $i = 1, \ldots, p$. If, additionally, any improvement in one criterion $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ resulting from

the change of $\mathbf{x}^*$ to $\mathbf{x}$ results in deterioration $f_j(\mathbf{x}^*) < f_j(\mathbf{x})$ of at least one other criterion f_j, then $\mathbf{x}^*$ is said to be *(Pareto) efficient*. The set of weakly (Pareto) efficient solutions for problem (4.1) is denoted by $\mathrm{E}_W(X, f)$, and the set of (Pareto) efficient solutions for problem (4.1) is denoted by $\mathrm{E}(X, f)$.

There are many methods for computing efficient points for MOPs that are presented as single integrated problems of form (4.1). In general, there are two classes of approaches: scalarization and nonscalarizing methods [36]. These approaches convert the MOP into a single objective problem (SOP), a sequence of SOPs, or another MOP. Under some assumptions, solution sets of these new programs yield solutions of the original problem. Scalarization methods explicitly employ a scalarizing function to accomplish the conversion while nonscalarizing methods use other means. Among the nonscalarizing approaches there are methods using optimality concepts other than Pareto, descent methods transferred from single objective optimization, and a new class of set-oriented methods. The latter group also includes evolutionary algorithms that have become very successful in engineering design [27, 101].

Complex systems engineering design has motivated the development of a new class of methods dealing with multiple design disciplines, with each discipline generating multiple design criteria. The complexity is reflected in the system composition of subsystems and components. Multiple disciplines originate from various science and engineering areas such as fluid dynamics, thermodynamics, structures, etc., that interact with each other within the design process, while multiple criteria are required to describe the system performance. In theory, the resulting all-in-one (AiO) multiobjective and multidisciplinary design optimization (MDO) problem could be solved for its efficient set. In reality, the underlying AiO problem having form (4.1) is never solved directly because the designs of the system, the subsystems, and the components are typically assigned to independent engineering teams with complementary background and expertise. Each team deals with a multiobjective optimization subproblem related to a discipline or subsystem or component. These disciplines cannot be dealt with independently because they together contribute to the overall system behavior. The distinction among the teams is in concert with the properties of the optimization subproblems which, in general, have different domains and characteristics, belong to different disciplines, and require different solution algorithms. Due to the different backgrounds of the teams and the limited understanding of each group towards the other disciplines, the information exchanged is restricted only to some optimization quantities, either variables or functions. The flow of incomplete

information across the disciplines, i.e., among the design teams, is a particular characteristic of the multidisciplinary design.

Many papers present applications of multiobjective MDO in various areas of engineering design ([79] [91] [66] [30] [22] [74] [28]). Methodologies such as Multiobjective Collaborative Optimization ([107] [95]), Multiobjective Concurrent Subspace Optimization (CSSO) ([61] [63] [64] [62] [60]), and a bilevel method ([125]) have also been developed. For multilevel systems, an approach based on the use of lower-level efficient designs as targets for upper-level designs and the method of Analytical Target Cascading is proposed in [77].

Another direction of research involves the development of genetic algorithms for multiobjective MDO. In [51], [50], [52] genetic algorithms are developed for MDO optimization problems with global and local variables and demonstrate their applicability to engineering design problems. A genetic algorithm for multiobjective CSSO is proposed in [88].

An approach to the computation of the AiO efficient designs with sharing of information between subsystems is presented in [59], [20]. The information shared between the subsystems is passed in the form of approximated objective functions. A comparison of information passing strategies is performed in [58]. The set of AiO efficient designs is compared with the designs computed for the decomposed counterpart which is coordinated with MDO and game-theoretic approaches.

Some of the proposed approaches and methodologies recognize the need for computing tradeoffs within each subproblem and between the subproblems and some others capture the essence of multidisciplinary design in which mathematical models are kept internal to each discipline and may render a complete exchange of information impossible. However, a majority of these methods are not supported with rigorous mathematical results on the convergence of proposed algorithms or the completeness of the implicitly computed AiO efficient set. To address the latter and facilitate tradeoff analysis for design problems with a large number of performance criteria, a reduction of the original problem to a a family of bicriteria subproblems allowing designers to effectively use decision-making in merely two dimensions is developed in [37]. In a similar vein, a theoretical examination of the relationships between efficient solutions of a multiobjective quasiseparable MDO problem and efficient solutions of its separable counterpart is provided in [44].

The work presented in this paper further contributes to the mathematical modeling and optimization of decomposable design problems. A Multiobjective Decomposition Algorithm (MODA) is proposed for computing the efficient points of an MOP in a distributed manner, while only having

access to the efficient points of the subproblems. The algorithm models the design process which is conducted by two engineering teams working independently with distinct mathematical models. The limited exchange of information between the subproblems is nevertheless sufficient for the computation of the AiO efficient designs. The algorithm makes use of a Gauss-Seidel decomposition technique, known as *block coordinate descent* (BCD), and the method of multipliers for coordinating the subproblems. While the mathematical details on the convergence of MODA are contained in [23], the current paper emphasizes the need to respect the autonomy of design disciplines within an engineering context.

MODA is applicable to design problems of many types. In this paper, it is applied to the vehicle layout design as a packaging problem approached with the paradigm of multiobjective multidisciplinary optimization. In packaging optimization problems, the objective and constraint functions are strongly related to the shape and the location of the components to be placed into a given enclosure. The shape of the vehicle components shape depends on their functionality, that is, the component design solutions that are based on engineering considerations affect the overall vehicle layout. This results in components that morph during the optimization process as their performance is optimized. A study of packaging optimization with morphing components is presented in [29], in which a bilevel scheme is exploited to optimize the overall layout of the vehicle underhood at the upper level while targeting the desired volume of the water reservoir at the lower level. In the current paper, the layout of components in the underhood of the vehicle is optimized at the upper level, while optimizing the design of one of these components, a Lithium-ion (Li-ion) battery, at the lower level. The Li-ion battery pack must not only be designed under demanding thermal criteria, but must itself be optimally placed within the underhood of the vehicle. The problem at each level corresponds to the design of a system requiring highly specialized knowledge, yet the two problems display a necessary interaction due to the placement of the battery within the vehicle underhood.

The paper is organized as follows. In Section 4.2, a bilevel formulation of the packaging problem is proposed. MODA is developed in Section 4.3, and propositions are stated describing convergence conditions. Once MODA is stated, a detailed description of the vehicle level subproblem, the battery level subproblem, and a description of the solvers assigned to each subproblem are given in Section 4.4. In Section 4.5, details on the implementation of MODA for the vehicle packaging optimization are provided, including a discussion of favorable scaling of consistency constraints and of weighting schemes that address the limited information flow that is allowed between the

subproblems. Numerical results of the implementation are presented showing the MODA capability of exploring the tradeoffs generated by the multiple criteria at each level. A brief discussion of the weighting scheme for the scalarization is also given. Concluding remarks are given in Section 4.6.

4.2 Bilevel problem formulation

The vehicle layout design problem is modeled as a packaging optimization problem in which the design variables define the locations of components inside an enclosure in order to optimize some objective functions while satisfying design constraints. The typically utilized objective functions are related to the layout compactness and the position of center of gravity. Other objective functions may be chosen depending on the problem. In the case of vehicle optimization, vehicle dynamics, rollover safety, or temperature distribution of underhood components may be considered as well. These functions require dedicated simulation models which take as input the locations of components. It should be noticed that vehicle dynamics and rollover safety could be, for simplicity, reconducted to the optimization of the moment of inertia and the position of the center of gravity. Design constraints will certainly include the overlapping between the components, which is not allowed. It is clear that the design criteria, either objectives and constraints, strongly depend on the shape of the components to be arranged. This tie between the criteria and the component shapes is a peculiar feature of packaging optimization problems and is highlighted in the following formulation of the optimization problem at the vehicle level.

$$\begin{aligned} \min_{\mathbf{x}_v} \quad & \mathbf{f}_v(\mathbf{x}_l, \mathbf{s}) \\ \text{s.t.} \quad & \mathbf{x}_l \in X_l. \end{aligned} \tag{4.2}$$

The vector of design variables $\mathbf{x}_l$ collects the component locations and the vector $\mathbf{s}$ collects the parameters describing the shape of the components. The vector of objective functions is specified by $\mathbf{f}_v$. The set X_l contains the feasible values for $\mathbf{x}_l$.

The well-functioning within each component is also affected by component shape parameters. For example, the radiator performance depends on its geometry and size, while the volume of the water reservoir is decided based on thermal considerations. In electric or hybrid electric vehicles, which are the subject of this paper, the battery geometrical dimensions are a consequence of the

cell layout, which is optimized for thermal performance. The design of the components within the vehicle layout leads to component level optimization problems having the following form.

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & \mathbf{f}_c(\mathbf{x}_c) \\ \text{s.t.} \quad & \mathbf{x}_c \in X_c. \end{aligned} \tag{4.3}$$

The vector of objectives for the component subproblems is specified by $\mathbf{f}_c$, and the vector of component design variables $\mathbf{x}_c$ is taken from component level feasible set X_c. The shape of the component is a consequence of the design solution that is given by the design vector $\mathbf{x}_c$, that is, $\mathbf{s} = \mathbf{s}(\mathbf{x}_c)$.

The component shape parameters occur at both the vehicle level and the component level and simultaneously affect the measures of vehicle packing quality optimized in (4.2) and the measures of component performance optimized in (4.3). Thus, the change of the component geometry during the design phase should be considered when defining the vehicle layout and, on the other hand, the available space in the vehicle should be considered when the components are designed. This relationship would lead to a design problem (4.4) that is the collection of design problems (4.2) and (4.3).

$$\begin{aligned} \min_{\mathbf{x}_l, \mathbf{x}_c} \quad & [\mathbf{f}_v(\mathbf{x}_l, \mathbf{s}(\mathbf{x}_c)), \mathbf{f}_c(\mathbf{x}_c)] \\ \text{s.t.} \quad & \mathbf{x}_l \in X_l, \quad \mathbf{x}_c \in X_c. \end{aligned} \tag{4.4}$$

Problem (4.4) is nothing but a multiobjective optimizaton problem whose efficient designs can be computed once all of the problem data are known. In practice, problem (4.4) is never solved directly because vehicle and component designs are typically assigned to independent engineering teams with complementary background and expertise. This distinction among the design problems and their respective designing teams is in concert with the properties of problems (4.2) and (4.3), which have different optimization domains and characteristics, belong to different disciplines, and require different solution algorithms.

In the absence of the coupling parameter $\mathbf{s}(\mathbf{x}_c)$, the efficient points of (4.4) could be computed as the Cartesian product of efficient points for the individual subproblems (4.2) and (4.3), as shown in [44]. This decomposition methodology cannot be applied to problem (4.4) due to the presence of coupling between the constituent subproblems (4.2) and (4.3) in the form of $\mathbf{s}(\mathbf{x}_c)$.

The following variable definitions are useful to model this design scenario involving separate design groups working on their subproblems in the presence of intersubproblem coupling.

$$\begin{aligned} &\mathbf{x}_v = [\mathbf{x}_l, \mathbf{x}_s] \\ &\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{x}_s - \mathbf{s}(\mathbf{x}_c) = \mathbf{0}. \end{aligned} \tag{4.5}$$

The introduction of the variable $\mathbf{x}_s \in X_s$ gives the vehicle level designers the possibility to modify the component shape to improve the vehicle layout, while the equality constraint $\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c)$ enforces the consistency between the component shapes at both levels. The shape variable $\mathbf{x}_s$ is taken from a set of feasible shapes X_s, and the aggregation $\mathbf{x}_v$ of the position and shape variables is taken from the set defined as $X_v = X_\ell \times X_s$.

Consequently, the integration of design problems (4.2) and (4.3) considering definitions (4.5) is formulated as problem (4.6)

$$\begin{aligned} \min_{\mathbf{x}_v, \mathbf{x}_c} \quad & [\mathbf{f}_v(\mathbf{x}_v), \mathbf{f}_c(\mathbf{x}_c)] \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c \\ & \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}. \end{aligned} \tag{4.6}$$

The paradigm of decomposition must be considered in this multidisciplinary optimization problem to really picture the design process with tasks assigned to different teams at different levels. The reformulation of subproblems (4.2) and (4.3) induced from problem (4.6) reads

$$\begin{aligned} \min_{\mathbf{x}_v, \mathbf{x}_c} \quad & \mathbf{f}_v(\mathbf{x}_v) \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c \\ & \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0} \end{aligned} \tag{4.7}$$

and

$$\begin{aligned} \min_{\mathbf{x}_v, \mathbf{x}_c} \quad & \mathbf{f}_c(\mathbf{x}_c) \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v \quad \mathbf{x}_c \in X_c \\ & \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}. \end{aligned} \tag{4.8}$$

In [37], it is shown that weakly efficient solutions for problem (4.6) may be obtained by computing the efficient solutions of subproblems (4.7) and (4.8) separately. However, in the current study, the information available to subproblems (4.7) and (4.8) is limited in the following sense. The vehicle layout designer solving problem (4.7) has knowledge of a specific value $\overline{\mathbf{x}}_c$ communicated from the component designer, but no knowledge of the set X_c and no control over the value of $\mathbf{x}_c$. Similarly, the component designer solving problem (4.8) has knowledge of a specific value $\overline{\mathbf{x}}_v$ communicated from the vehicle layout designer, but no knowledge of the set X_v and no control over the value of $\mathbf{x}_v$. Problems (4.7) and (4.8) are restated under these assumptions as follows.

$$\begin{aligned} \min_{\mathbf{x}_v} \quad & \mathbf{f}_v(\mathbf{x}_v) \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v \\ & \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) = \mathbf{0} \end{aligned} \tag{4.9}$$

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & \mathbf{f}_c(\mathbf{x}_c) \\ \text{s.t.} \quad & \mathbf{x}_c \in X_c \\ & \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) = \mathbf{0}. \end{aligned} \tag{4.10}$$

Thus, efficient points for the all-in-one (AiO) problem (4.6) must be computed using only references to subproblems (4.9) and (4.10), and so the above mentioned approach referred to in [37] is not applicable.

4.3 Decomposition and coordination

Section 4.2 motivated and developed the formulation of the vehicle design problem (4.6) in terms of the coordinated subproblems (4.9) and (4.10). By suitably reformulating subproblems (4.9) and (4.10), a coordinated decomposition procedure may be stated for computing efficient points of (4.6). This procedure respects the autonomous qualities of each subproblem, while incorporating essential information about their relations to the all-in-one problem (4.6).

The use of the weighted-sum method for computing efficient points for subproblems (4.9)

and (4.10) leads to the statement of single objective problems

$$\begin{aligned} \min_{\mathbf{x}_v} \quad & f_v(\mathbf{x}_v) \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v \\ & \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) = \mathbf{0} \end{aligned} \tag{4.11}$$

and

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & f_c(\mathbf{x}_c) \\ \text{s.t.} \quad & \mathbf{x}_c \in X_c \\ & \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) = \mathbf{0}, \end{aligned} \tag{4.12}$$

where $f_v = \mathbf{w}_v \cdot \mathbf{f}_v$ and $f_c = \mathbf{w}_c \cdot \mathbf{f}_c$ are computed using positive weight vectors $\mathbf{w}_v \in \mathbb{R}^p_>$ and $\mathbf{w}_c \in \mathbb{R}^p_>$ normalized so that their vector components sum to one.

Fixing weight vectors $\mathbf{w}_v$ and $\mathbf{w}_c$, the underlying AiO problem becomes the following biobjective problem

$$\begin{aligned} \min_{\mathbf{x}_v, \mathbf{x}_c} \quad & [f_v(\mathbf{x}_v), f_c(\mathbf{x}_c)] \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c \\ & \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}. \end{aligned} \tag{4.13}$$

The weighted-sum scalarization applied to (4.13) takes the form

$$\begin{aligned} \min_{\mathbf{x}_v, \mathbf{x}_c} \quad & \alpha_v f_v(\mathbf{x}_v) + \alpha_c f_c(\mathbf{x}_c) \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c \\ & \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}, \end{aligned} \tag{4.14}$$

for weights $\alpha_v > 0, \alpha_c > 0$ with $\alpha_v + \alpha_c = 1$.

Although weights for problem (4.6) could be given by the AiO weight vector $\mathbf{w} = [\alpha_v \mathbf{w}_v, \alpha_c \mathbf{w}_c]$, the distinction between *intra*-subproblem weights, $\mathbf{w}_v$ and $\mathbf{w}_c$, and *inter*-subproblem weights, α_v and α_c, is maintained in line with the assumption that subproblem (4.9) has direct knowledge of its intra-subproblem weight $\mathbf{w}_v$, and (4.10) has direct knowledge of its intra-subproblem weight $\mathbf{w}_c$,

but neither subproblem has direct knowledge of either α_v or α_c. The implementation of this idea and its motivation are made clear in Section 4.5.

Weighted-sum at both the intra-subproblem and inter-subproblem levels is used initially due to its analytical simplicity. The use of weighted-sum may later be replaced with more elaborate scalarization techniques as the problem properties warrant.

The equality constraints of problems (4.11) and (4.12) are relaxed and incorporated into the objective function as Lagrange term and penalty term, resulting in the subproblems

$$\begin{aligned} \min_{\mathbf{x}_v} \quad & \alpha_v \, f_v(\mathbf{x}_v) + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) + \frac{\mu}{2} \left\| \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v \end{aligned} \tag{4.15}$$

and

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & \alpha_c \, f_c(\mathbf{x}_c) + \mathbf{v}^T \cdot \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) + \frac{\mu}{2} \left\| \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_c \in X_c, \end{aligned} \tag{4.16}$$

where $\mathbf{v}$ denotes the vector of Lagrange multipliers associated with the consistency equality constraints and $\mu > 0$ is a penalty coefficient.

The resulting AiO augmented Lagrangian problem that is to be solved using only references to problems (4.15) and (4.16) is given by

$$\begin{aligned} \min_{\mathbf{x}_v, \, \mathbf{x}_c} \quad & \alpha_v f_v(\mathbf{x}_v) + \alpha_c f_c(\mathbf{x}_c) \\ & + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) + \frac{\mu}{2} \left\| \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c, \end{aligned} \tag{4.17}$$

where the manner in which information about α_v and α_c is communicated to subproblems (4.15) and (4.16) is described in Section 4.5.

The distributed and coordinated computation of designs efficient for problem (4.6) using subproblems (4.15) and (4.16) is based on the Block Coordinate Descent (BCD) method [11, 49, 114], and the method of multipliers [10, 11], both of which have previously been developed in the single objective optimization setting. The BCD method and the method of multipliers are described in Section 4.3.1.

4.3.1 Block Coordinate Descent (BCD) and the method of multipliers

The BCD method is applied to subproblems (4.15) and (4.16) with a Lagrange multiplier $\mathbf{v}$ and a penalty coefficient μ treated as fixed parameters. In turn, the method of multipliers is applied for updating $\mathbf{v}$ and μ.

BCD is a block nonlinear Gauss-Seidel approach to solving, in a distributed manner, the single objective optimization problem having the form

$$\begin{aligned} \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \quad & f(\mathbf{x}_1,\ldots,\mathbf{x}_m) \\ \text{s.t} \quad & \mathbf{x}_i \in X_i \quad \text{for} \quad i = 1,\ldots,m, \end{aligned} \tag{4.18}$$

where each X_i, $i = 1,\ldots,m$ is a closed and convex set. The writing of the feasible set X as a Cartesian product of local feasible sets X_i, $i = 1,\ldots,m$ corresponds to a partition $\mathbf{x} = [\mathbf{x}_1,\ldots,\mathbf{x}_m]$ of the design space in the sense that $\mathbf{x} \in X$ if and only if $\mathbf{x}_i \in X_i$ for each $i = 1,\ldots,m$. Thus, each constraint set X_i is local to block coordinate $\mathbf{x}_i$ for each $i = 1,\ldots,m$.

BCD is applied to a problem of form (4.18) in the following manner. Each block coordinate $\mathbf{x}_i$ is updated separately in a fixed cyclic order using the computation

$$\mathbf{x}_i^k = \arg\min_{\mathbf{x}_i \in X_i} f_{i,k}(\mathbf{x}_i), \tag{4.19}$$

where the function $f_{i,k} : X_i \to \mathbb{R}$ is defined by evaluating f at $\mathbf{x}_j = \mathbf{x}_j^k$ for $j < i$, and at $\mathbf{x}_j = \mathbf{x}_j^{k-1}$ for $j > i$, while leaving $\mathbf{x}_i$ as a vector variable. In other words, the minimization of $f_{i,k}$ over $\mathbf{x}_i \in X_i$ amounts to the minimization of f over $\mathbf{x}_i \in X_i$ while treating every other block coordinate $\mathbf{x}_j$, $j \neq i$, as being fixed to its most recently updated value. Each instance (i,k) of problem (4.19) corresponds to a subproblem of (4.18).

The iterative application of update (4.19) to each block coordinate in a fixed cyclic order starting from an initial value $\mathbf{x}^0$ is stated in Algorithm 5. The algorithm takes as input the scalar-valued objective function f, the sets $X_1,\ldots,X_m$ containing the feasible values that each respective block coordinate $\mathbf{x}_1,\ldots,\mathbf{x}_m$ may take, and also the initial value $\mathbf{x}^0$. Termination of Algorithm 5 occurs when convergence within a specified tolerance is detected within the generated sequence $\{\mathbf{x}^k\}$. The returned output $\overline{\mathbf{x}}$ is taken as the value of $\mathbf{x}^k$ when the repeat loop terminates. The value $\overline{\mathbf{x}}$ thus approximates a limit point to the sequence $\{\mathbf{x}^k\}$.

Algorithm 5 BCD

function BCD(f, $X_1, \ldots, X_m$,$\mathbf{x}^0$)
  $k \leftarrow 0$
  repeat
    Update: $k \leftarrow k+1$
    for $i \in \{1, \ldots, m\}$ **do**
      Set:
      $f_{i,k}(\mathbf{x}_i) \leftarrow f(\mathbf{x}_1^k, \ldots, \mathbf{x}_{i-1}^k, \mathbf{x}_i, \mathbf{x}_{i+1}^{k-1}, \ldots, \mathbf{x}_m^{k-1})$
      Compute:
      $\mathbf{x}_i^k \approx \arg\min_{x_i \in X_i} \ f_{i,k}(\mathbf{x}_i)$
    end for
  until $\left(||\mathbf{x}^k - \mathbf{x}^{k-1}|| < tol\right)$
  $\overline{\mathbf{x}} \leftarrow \mathbf{x}^k$
  return $\overline{\mathbf{x}}$
end function

The BCD method is applied to problem (4.17) with two blocks corresponding to subproblems (4.15) and (4.16). During the application of BCD, the Lagrange multiplier $\mathbf{v}$ and the penalty coefficient μ are treated as fixed parameters. By fixing these values, the constraint set of problem (4.17) has the structure of problem (4.18) required for the application of BCD when the following identifications

$$\begin{aligned} &\mathbf{x}_1 = \mathbf{x}_v, \quad \mathbf{x}_2 = \mathbf{x}_c \\ &X_1 = X_v, \quad X_2 = X_c \\ &f = \alpha_v \mathbf{w}_v^T \cdot \mathbf{f}_v(\mathbf{x}_v) + \alpha_c \mathbf{w}_c^T \cdot \mathbf{f}_c(\mathbf{x}_c) \\ &\qquad + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) + \tfrac{\mu}{2} \|\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c)\|_2^2 \end{aligned}$$

are made. For fixed $\mathbf{v}$ and μ, the BCD subproblems are identified with the instantiations of subproblems (4.15) and (4.16) resulting from the updates of $\overline{\mathbf{x}}_c$ and $\overline{\mathbf{x}}_v$, respectively.

For any given fixed values of the Lagrange parameters $\mathbf{v}$ and μ, the application of BCD typically yields optimal solutions of (4.17) that are not feasible solutions for problem (4.14). Thus, a process for adjusting $\mathbf{v}$ and μ needs to be included in addition to the updates of the BCD method. The updates of $\mathbf{v}$ and μ leading to optimal solutions of problem (4.17) that are also feasible for problem (4.14) are obtained using the method of multipliers.

The method of multipliers is applied to problems having the form

$$\begin{aligned} \min_{\mathbf{x} \in X} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \end{aligned} \tag{4.20}$$

with the use of the augmented Lagrangian reformulation given by

$$\min_{\mathbf{x}\in X} f_{\mathbf{v},\mu}(\mathbf{x}), \tag{4.21}$$

where the augmented Lagrange function $f_{\mathbf{v},\mu}$ is given by

$$f_{\mathbf{v},\mu}(\mathbf{x}) := f(\mathbf{x}) + \mathbf{v}^T \cdot \mathbf{h}(\mathbf{x}) + \frac{\mu}{2} \|\mathbf{h}(\mathbf{x})\|_2^2$$

for a vector $\mathbf{v}$ of Lagrange multipliers and a penalty parameter $\mu > 0$.

An iteration of the method of multipliers takes the form

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}\in X} f_{\mathbf{v}^k,\mu_k}(\mathbf{x}) \tag{4.22}$$

$$\mathbf{v}^{k+1} = \mathbf{v}^k + \mu_k(\mathbf{h}(\mathbf{x})) \tag{4.23}$$

$$\mu_{k+1} = c\mu_k \quad 2 \leq c \leq 10, \tag{4.24}$$

where update (4.22) is obtained by computing an optimal argument to problem (4.21). Updates (4.22), (4.23), and (4.24), when repeated, generate a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$. Under certain conditions, any limit points $(\overline{\mathbf{x}}, \overline{\mathbf{v}})$ satisfy the Karush-Kuhn-Tucker conditions (e.g., see [10] or [11]) for problem (4.20). The theory of convergence for the generated sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ to Lagrange stationary points $((\mathbf{x}^*, \mathbf{v}^*))$ is explored in [10] under both exact and inexact minimizations in (4.22).

Method of multiplier updates may be applied to problem (4.14) as it has the form of problem (4.20). Identifying problem (4.21) with problem (4.17), the BCD method is used to compute optimal arguments to (4.17) in a distributed manner with $\mathbf{v}$ and μ treated as fixed parameters. Once BCD terminates, $\mathbf{v}$ and μ are updated as in (4.23) and (4.24) with the necessary identifications with problems (4.14) and (4.17).

The integration of the method of multipliers with BCD described above is presented in Algorithm 6. This algorithm takes input f, X_i having the meaning given in problem (4.20), and $\mathbf{x}^0$ is the starting value for $\mathbf{x}$ as in Algorithm 5. Additionally, Algorithm 6 takes as input the vector of linear functions $\mathbf{h}(\mathbf{x})$ defining the global linear constraints; $\mathbf{v}^0$, the initial value for the Lagrange multiplier $\mathbf{v}$; and μ_0, the initial value for the penalty coefficient μ. The function $f_{v,\mu}$ passed into Algorithm BCD is updated whenever $\mathbf{v}$ and μ are updated. Once convergence of BCD is observed

Algorithm 6 COOR integrates BCD and method of multipliers for single objective optimization

function COOR(f, $X_1, \ldots, X_m$,$\mathbf{h}$,$\mathbf{x}^0$,$\mathbf{v}^0$,μ_0)
  $k \leftarrow 0$
  repeat
    $f_{v,\mu}(\mathbf{x}) \leftarrow f(\mathbf{x}) + \mathbf{v}^k \cdot \mathbf{h}(\mathbf{x}) + \frac{\mu_k}{2} \|\mathbf{h}(\mathbf{x})\|_2^2$
    $\mathbf{x}^{k+1} \leftarrow \text{BCD}(f_{v,\mu}, X_1, \ldots, X_m, \mathbf{x}^k)$
    $\mathbf{v}^{k+1} \leftarrow \mathbf{v}^k + \mu_k \mathbf{h}(\mathbf{x})$
    $\mu_{k+1} \leftarrow c\mu_k$ for $2 \leq c \leq 10$
    Update: $k \leftarrow k + 1$
  until $\left(\|\mathbf{h}(\mathbf{x})\| < tol \text{ AND } \left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| < tol\right)$
  $(\overline{\mathbf{x}}, \overline{\mathbf{v}}) \leftarrow \left(\mathbf{x}^k, \mathbf{v}^k\right)$
  return $(\overline{\mathbf{x}}, \overline{\mathbf{v}})$
end function

within the call to Algorithm 5, then updates of $\mathbf{v}$ and μ are performed using formulas (4.23) and (4.24). This update process is repeated until $\left\|\mathbf{h}(\mathbf{x}^k)\right\| < tol$ and $\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| < tol$ are observed for tolerance $tol > 0$. The values $\mathbf{x}^k$ and $\mathbf{v}^k$ at termination are stored as the limiting values $(\overline{\mathbf{x}}, \overline{\mathbf{v}})$ of the generated sequence $\left\{\left(\mathbf{x}^k, \mathbf{v}^k\right)\right\}$. The pair $(\overline{\mathbf{x}}, \overline{\mathbf{v}})$ is then returned as the output.

The decomposition and coordination approach stated in Algorithm 6 is the main engine behind the decomposition and coordination approach in the multiobjective setting. Algorithm 6 may be applied to each instance of problem (4.14) specified by intra-subproblem weight vectors, $\mathbf{w}_v$ and $\mathbf{w}_c$, and inter-subproblem weights, α_v and α_c. The repeated application of Algorithm 6 over a set $\mathcal{W}$ of scalarizations results in the Multiobjective Decomposition Algorithm (MODA) presented in Algorithm 7. In addition to the inputs of Algorithm 6, Algorithm 7 takes set $\mathcal{W}$ containing scalarizing functionals (determined by weight vectors) used to reformulate an MOP into a single objective optimization problem. Each value $(\overline{\mathbf{x}}, \overline{\mathbf{v}})$ returned by Algorithm 7 is stored in set E. The set E is returned as the output at the conclusion of Algorithm 7. The $\overline{\mathbf{x}}$ values of each element of E are candidate efficient points for problem (4.6).

The solution approach developed in this section is intended to model the design process in which two engineering teams work independently on their design problems while sharing a limited amount of information taking the form of subproblem specific targets that appear in the Lagrange and penalty terms. The conditions under which this shared information is adequate for the computation of feasible efficient designs for the all-in-one problem (4.6) is discussed in Section 4.3.2.

Algorithm 7 Multiobjective Decomposition Algorithm (MODA)

function MODA($\{\mathbf{f}_i\}$, $X_1, \ldots, X_m$,$\mathbf{h}$,$\mathbf{x}^0$,$\mathbf{v}^0$,μ_0,$\mathcal{A}$,$\mathcal{W}$)
  $E \leftarrow \emptyset$
  for $([\mathbf{w}_i] \in \mathcal{W}$ **do**
    for $([\alpha_i] \in \mathcal{A}$ **do**
      $f_w \leftarrow \sum_{i=1}^{m} \alpha_i \mathbf{w}_i \cdot \mathbf{f}_i$
      $(\overline{\mathbf{x}}, \overline{\mathbf{v}}) \leftarrow$ COOR(f_w,$X_1, \ldots, X_m$,$\mathbf{h}$,$\mathbf{x}^0$,$\mathbf{v}^0$,μ_0)
      $E \leftarrow E \cup \{(\overline{\mathbf{x}}, \overline{\mathbf{v}})\}$
    end for
  end for
  return E
end function

4.3.2 Convergence of MODA

The convergence analysis of MODA follows from the integration of the convergence analysis of its constituent parts, the BCD method and the method of multipliers.

Proposition 12 states conditions under which the sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ generated with Algorithm 6 converges to $(\mathbf{x}^*, \mathbf{v}^*)$, where $\mathbf{x}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*)$ is a global minimum for an optimization problem of the form

$$\begin{aligned} \min_{\mathbf{x}_1, \mathbf{x}_2} \quad & f(\mathbf{x}_1, \mathbf{x}_2) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0} \\ & \mathbf{x}_1 \in X_1, \quad \mathbf{x}_2 \in X_2, \end{aligned} \tag{4.25}$$

and $\mathbf{v}^*$ is the multiplier associated with constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ that, together with $\mathbf{x}^*$, satisfies the necessary conditions of optimality over convex set X for problem (4.25) given by

1. Stationary point condition:

$$\left[(\nabla_{\mathbf{x}} f(\mathbf{x}^*) + (\mathbf{v}^*)^T \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}^*)\right]^T (\mathbf{x} - \mathbf{x}^*) \geq 0 \tag{4.26}$$

   for all $\mathbf{x} \in X$, and

2. Feasibility condition: $\mathbf{h}(\mathbf{x}^*) = \mathbf{0}$.

Proposition 12. *For problem* (4.25)*, let* $X_1 \subset \mathbb{R}^{n_1}$ *and* $X_2 \subset \mathbb{R}^{n_2}$ *be closed, convex sets; let*

$f : X_1 \times X_2 \to \mathbb{R}$ be continuously differentiable with bounded level sets

$$\mathcal{X}^0 = \left\{(\mathbf{x}_1, \mathbf{x}_2) \in X_1 \times X_2 \,:\, f(\mathbf{x}_1, \mathbf{x}_2) \leq f(\mathbf{x}_1^0, \mathbf{x}_2^0)\right\}$$

for a starting point $(\mathbf{x}_1^0, \mathbf{x}_2^0)$; and let $\mathbf{h} : X_1 \times X_2 \to \mathbb{R}^\ell$ be continuously differentiable. Let a sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ be generated using Algorithm 6 under the following assumptions:

1. *Each $\mathbf{x}^k$ computed by update (4.22) using BCD is a global minimizer for the augmented Lagrangian problem*

$$\begin{aligned} \min_{\mathbf{x}_1, \mathbf{x}_2} \quad & f(\mathbf{x}_1, \mathbf{x}_2) \\ & + (\mathbf{v}^k)^T \cdot \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) + \frac{\mu^k}{2} \left\|\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2)\right\|_2^2 \\ s.t. \quad & \mathbf{x}_1 \in X_1, \quad \mathbf{x}_2 \in X_2; \end{aligned} \tag{4.27}$$

2. *The sequence $\left\{\mathbf{v}^k\right\}$ generated using the method of multipliers is bounded;*

3. *The penalty parameter $\mu > 0$ grows arbitrarily large.*

Then any limit point $(\mathbf{x}^*, \mathbf{v}^*)$ of the generated sequence $\left\{(\mathbf{x}^k, \mathbf{v}^k)\right\}$ satisfies the necessary condition for optimality stated in (4.26), and $\mathbf{x}^*$ is a global minimizer for problem (4.25) satisfying $\mathbf{h}(\mathbf{x}^*) = \mathbf{0}$.

Proof. The continuous differentiability of f and the
boundedness of X^0 are sufficient for each limit point $\mathbf{x}^k$ obtained from the sequence generated with BCD to satisfy the stationary point condition for problem (4.27) with $\mathbf{v} = \mathbf{v}^k$, which is given by

$$\left[(\nabla_{\mathbf{x}} f(\mathbf{x}^k) + (\mathbf{v}^k)^T \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}^k)\right]^T \left(\mathbf{x} - \mathbf{x}^k\right) \geq 0 \quad \text{for all } \mathbf{x} \in X$$

(see Corollary 2 of [49]). When these limit points are furthermore global minima for (4.27), then assumptions 2 and 3 of this proposition are sufficient for the application of Proposition 2.1 of [10], by which it follows that the limit point $\mathbf{x}^*$ of the sequence $\left\{\mathbf{x}^k\right\}$ generated by Algorithm 6 is a global minimizer for problem (4.25) satisfying $h(\mathbf{x}^*) = \mathbf{0}$. By Proposition 1 of [120], the pair $(\mathbf{x}^*, \mathbf{v}^*)$ satisfies the stationary condition (4.26). □

The computation of $(\mathbf{x}^*, \mathbf{v}^*)$ satisfying the stationary condition (4.26) and the feasibility

requirement $\mathbf{h}(\mathbf{x}^*) = \mathbf{0}$ for problem (4.25) allows for μ to grow more slowly, thus improving the conditioning of the computations.

Using the foundation laid with Proposition 12, Proposition 13 establishes conditions under which MODA computes efficient points for an AiO bilevel design problem

$$\begin{aligned} \min_{\mathbf{x}_1,\mathbf{x}_2} \quad & [\mathbf{f}_1(\mathbf{x}_1), \mathbf{f}_2(\mathbf{x}_2)] \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0} \\ & \mathbf{x}_1 \in X_1, \quad \mathbf{x}_2 \in X_2, \end{aligned} \tag{4.28}$$

while using references to the nonintegrable subproblems

$$\begin{aligned} \min_{\mathbf{x}_1} \quad & \mathbf{f}_1(\mathbf{x}_1) \\ \text{s.t.} \quad & \mathbf{x}_1 \in X_1 \\ & \mathbf{h}(\mathbf{x}_1, \overline{\mathbf{x}}_2) = \mathbf{0} \end{aligned} \tag{4.29}$$

and

$$\begin{aligned} \min_{\mathbf{x}_2} \quad & \mathbf{f}_2(\mathbf{x}_2) \\ \text{s.t.} \quad & \mathbf{x}_2 \in X_2 \\ & \mathbf{h}(\overline{\mathbf{x}}_1, \mathbf{x}_2) = \mathbf{0}, \end{aligned} \tag{4.30}$$

where X_1 and X_2 are closed convex sets and $\mathbf{h}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{0}$ are linear consistency constraints.

Proposition 13. *Let $\mathcal{W}$ be a set of intra-subproblem weight vectors and $\mathcal{A}$ be a set of inter-subproblem weight vectors for problems* (4.28)-(4.30)*. For subproblems* (4.29) *and* (4.30)*, define the weighted-sum scalarizations $f_1 : X_1 \to \mathbb{R}$ and $f_2 : X_2 \to \mathbb{R}$ by*

$$f_1(\mathbf{x}_1) = \mathbf{w}_1 \cdot \mathbf{f}_1(\mathbf{x}_1)$$

and

$$f_2(\mathbf{x}_2) = \mathbf{w}_2 \cdot \mathbf{f}_2(\mathbf{x}_2),$$

respectively, for each $[\mathbf{w}_1, \mathbf{w}_2] \in \mathcal{W}$.

Furthermore, let the assumptions of Proposition 12 apply to each instance of problem (4.25)

with objective function $f : X_1 \times X_2 \to \mathbb{R}$ *defined by* $f(\mathbf{x}_1, \mathbf{x}_2) = \alpha_1 f_1(\mathbf{x}_1) + \alpha_2 f_2(\mathbf{x}_2)$ *for each* $(\alpha_1, \alpha_2) \in \mathcal{A}$. *Then MODA generates efficient points for the AiO problem* (4.28).

Proof. The proof follows from noting that MODA is just the repeated application of Algorithm 6 on scalarized AiO problems of form (4.25) constructed for each $\mathbf{w} \in \mathcal{W}$ and $\alpha \in \mathcal{A}$. Each BCD block update corresponds to solving subproblems of forms (4.29) and (4.30). □

Propositions 12 and 13 state assumptions on the AiO problem (4.28) under which the use of MODA is justified. However, in applying MODA to engineering design problems such as the vehicle design problem, the assumptions stated in Propositions 12 and 13 may be difficult to verify beforehand. The application of MODA to the vehicle design problem therefore requires a deeper understanding of the problem. The specific meaning that the design variables $\mathbf{x}_v$, $\mathbf{x}_c$; the feasible sets X_v, X_c; the objective function vectors $\mathbf{f}_v$, $\mathbf{f}_c$; and the consistency constraint $\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}$ have for the vehicle design problem is described in Section 4.4.

4.4 Models and algorithms for packaging optimization

The modeling and computational methodologies proposed in this paper are applied to vehicle packaging optimization represented by problem (4.6). In this section models for subproblems (4.2) and (4.3) are presented along with the utilized solution algorithms.

The vehicle layout problem is to place the components of the underhood compartment (see Fig. 4.1). Besides the battery, other five components are considered, the engine, the radiator, the coolant reservoir, the air filter, and the brake booster. Since the focus is on the battery design, only the battery is considered as morphable component according to its functionality. As shown in Fig. 4.1, the components are modeled using tessellated representation, i.e., the STL format. This representation is used to conveniently generate voxel models of the components and the underhood enclosure. The voxel representation allows the computation volume of intersecting components and the enclosure, which have irregular and complex shapes. The intersection constraint is in fact a peculiar characteristic of packaging optimization which subjects the objective and constraint functions to geometric considerations.

In the vehicle layout problem, the vector

$$\mathbf{x}_v = (x, y, z, L_v, W_v)$$

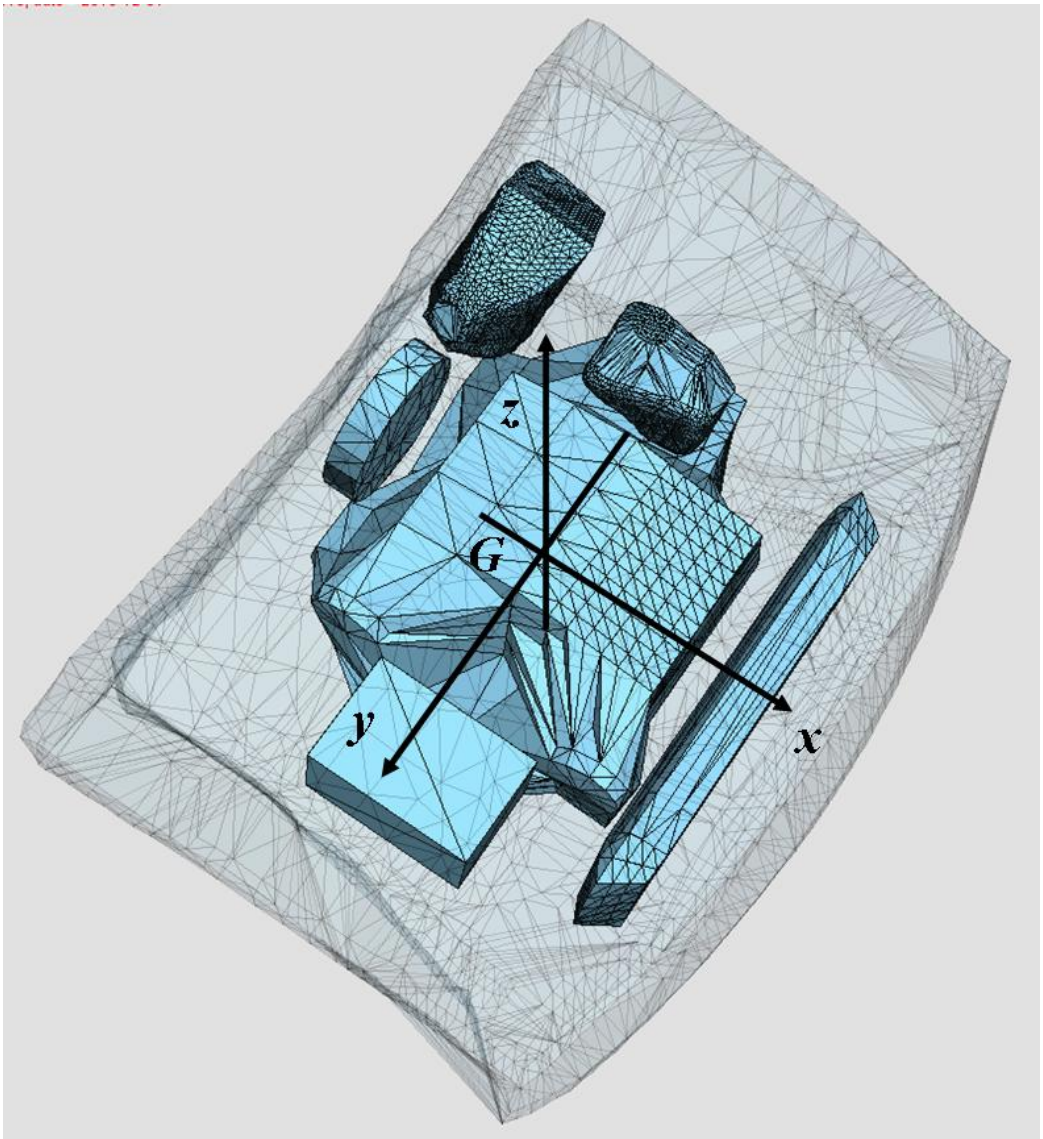

Figure 4.1: Vehicle underhood components

collects the locations of the components (three coordinates, x, y and z) as well as the battery component shape parameters L_v and W_v being, respectively, the length and width (in mm) of the battery box. (Orientations of the components are not considered.) The vector $\mathbf{f}_v = (J_G, A, U)$ has three objectives, layout compactness J_G, accessibility A, and vehicle survivability U. The compactness (also referred as compacity) is measured by the moment of inertia of the components with respect to the vertical axis passing through the center of gravity G, with coordinates x_G and y_G, of the underhood enclosure. Objective J_G is to be minimized and is given by

$$J_G(\mathbf{x}_v) = \sum_{i=1}^{6} \left(J_i + \left((x_i - x_G)^2 + (y_i - y_G)^2 \right) m_i \right), \tag{4.31}$$

where index i identifies all the six components under the hood and m_i's and J_i's are their masses and moments of inertia, respectively.

The accessibility of a component in a given layout is measured by the number of components that have to be removed before accessing it from one direction. The layout accessibility is the sum of all the component accessibilities. For the layout in Fig. 4.2, the component accessibilities are reported in Table 4.1. A weight is associated with each component to determine its importance. The components that require frequent access for maintenance are associated with larger weights. For a given layout solution $\mathbf{x}_v$, the accessibility to be minimized is

$$A(\mathbf{x}_v) = \sum_{i=1}^{6} A_i(\mathbf{x}_v). \tag{4.32}$$

Table 4.1: Accessibility values for the example in Fig. 4.2 (left)

Component	Weight	To be removed	Accessibility A_i
1	3	2	6
2	3	0	0
3	1	1	1
4	9	1	9
5	1	0	0
6	1	0	0

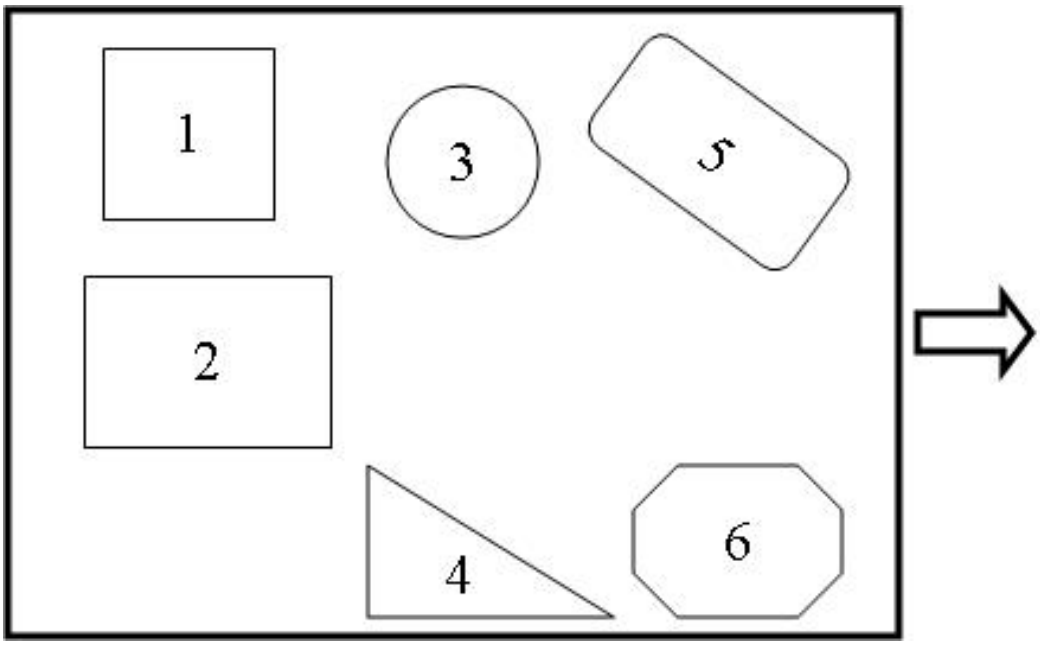

Figure 4.2: Example for accessibility computation

Vehicle survivability is intended here as the degree of protection offered to the components by the other components from the impact with external bodies. The degree of protection is computed by the overlapping area among components. For layout in Fig. 4.3, component 1 is partially protected by components 2 and 3 due to the overlapping areas O_{12} and O_{13}, respectively. The overlap area for component 1 is

$$O_1 = O_{12} + O_{13}, \tag{4.33}$$

which accounts for additional protection due to the double overlap offered by O_{123}. Weights p_i are introduced considering that some components are more crucial than others. The overlap area O_i is then normalized with respect to the area of the ith component surface P_i. The vehicle survivability is given by

$$U(\mathbf{x}_v) = \sum_{i=1}^{6} p_i \frac{O_i(\mathbf{x}_v)}{P_i} \tag{4.34}$$

and is to be maximized.

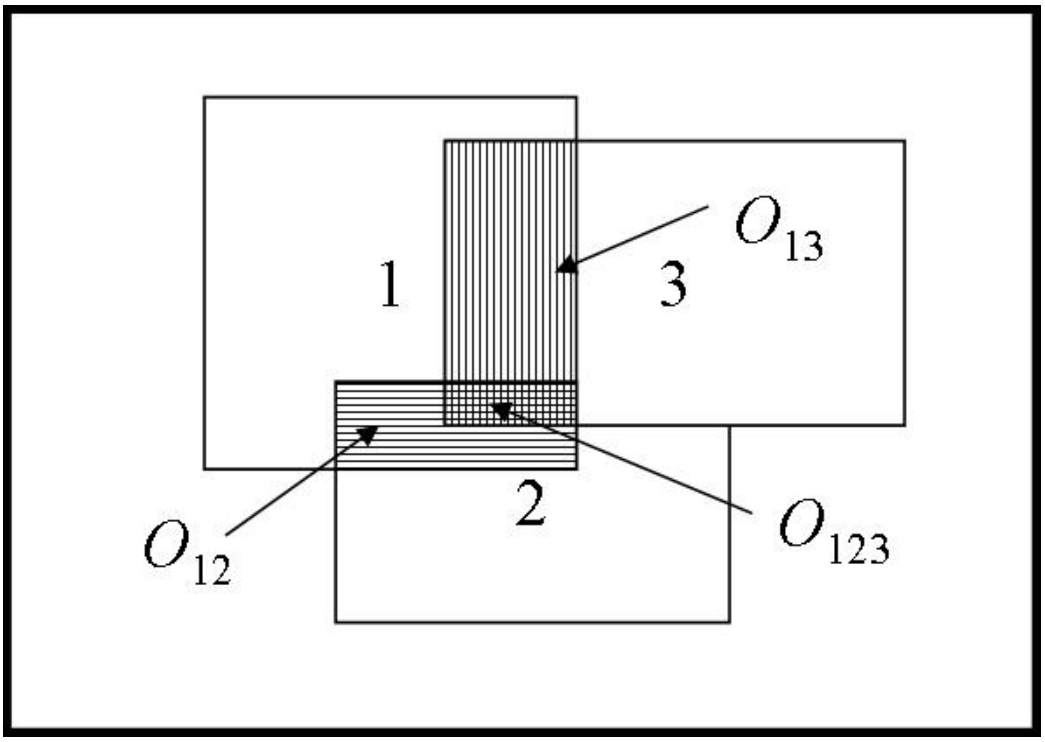

Figure 4.3: Example for survivability computation

The constraint set X_v for the vehicle design problem is given by

$$X_v := \left\{ \begin{array}{c} \mathbf{x}_v = (\mathbf{x}, \mathbf{y}, \mathbf{z}, L_v, W_v) \quad \text{s.t.} \\ \hline \ell_{x,i} < x_i < u_{x,i} \\ \ell_{y,i} < y_i < u_{y,i} \\ \ell_{z,i} < z_i < u_{z,i} \\ h_{overlap}(x_i, y_i, z_i, L_v, W_v) = 0 \\ \text{for } i = 1, \ldots, N_{comps} \end{array} \right\}.$$

The values $\ell_{x,i}$, $\ell_{y,i}$, and $\ell_{z,i}$ are lower bounds on the location parameters x_i, y_i, and z_i, while $u_{x,i}$, $u_{y,i}$, and $u_{z,i}$ are the corresponding upper bounds. The components are indexed by i, and N_{comps} is the number of components. The equality constraint $h_{overlap} = 0$ prevents the overlapping placement of components.

As already mentioned, only the battery is considered at the component level. The driving

design criterion for the layout of cells is the thermal behavior of the battery, as reflected in the distribution of battery cell temperatures during the operation of the vehicle. A thermal model of the battery simulates heat rejection through a liquid coolant at steady state conditions and provides the cell temperatures T_i, for $i = 1, \ldots, N$, as functions of certain layout variables, with the assumption that the battery cells are arranged into a triangular lattice having N_{cols} columns and N_{rows} rows (see Fig. 4.4 in which a column is highlighted in a layout with 12 columns and 6 cells per column). Cell spacing parameter p determines the distance d_{center} between the centers of two adjacent cells by the equation $d_{center} = p \cdot d_{diameter}$, where $d_{diameter}$ denotes the diameter of a cell. The battery level requirements on the battery box length and width, L_c and W_c, are determined by the following geometric formulas influenced by the values of p, N_{cols}, and N_{rows}.

$$L_c = \left(p \cdot \cos(\tfrac{\pi}{6}) \cdot (N_{cols} - 1) + 3\right) \cdot d_{cell},$$

$$W_c = p \cdot d_{cell} \cdot \left(N_{rows} + \tfrac{1}{2}\right)$$

These layout variables are collected into the vector denoted by $\mathbf{x}_c = (p, N_{col}, N_{row}, L_c, W_c)$.

Uniform heat rejection leads to uniform temperature distributions with beneficial effect on the cell durability. Thermal unevenness may cause electrical unbalance with consequent reduced lifetime and cell failure. Obtaining a uniform temperature distribution can be seen as the minimization of the deviations of the $N = 72$ cell temperatures T_i, for $i = 1, \ldots, N$, from the ideal operating cell temperature T_o, as stated in the following problem with N objective functions,

$$\min_{\mathbf{x}_c \in X_c} \left[\left(T_i(\mathbf{x}_c) - T_o\right)^2 \right]_{i=1,\ldots,N}, \tag{4.35}$$

where X_c is the component level constraint set defined by

$$X_c := \left\{ \begin{array}{c} \mathbf{x}_c = (p, N_{cols}, N_{rows}, L_c, W_c) \quad \text{s.t.} \\ \hline 1 < p \leq 2, \\ N_{cols} \cdot N_{rows} = N, \\ N_{cols} \in \{9, 12, 18\} \\ L_c = \left(p \cdot \cos(\frac{\pi}{6}) \cdot (N_{cols} - 1) + 3\right) \cdot d_{cell}, \\ W_c = p \cdot d_{cell} \cdot \left(N_{rows} + \frac{1}{2}\right) \end{array} \right\}.$$

As reported in [24], these objectives are conflicting, so that problem (4.35) requires an optimality concept such as Pareto efficiency. Due to peculiar characteristics of the objectives, three properties apply to problem (4.35): comparability and anonymity of the objectives, and the Pigou-Dalton principle of transfer. The objectives are comparable because they measure the same physical quantity, deviations from T_o. The objectives are anonymous because the temperature is important but the cell that specifically exhibits this temperature is not. The third property, the Pigou-Dalton principle of transfer, is satisfied because evenly distributed deviations, resulting in uniform cell temperatures, are preferred. Due to these properties, the equitability preference, a refinement of the Pareto efficiency, can be applied to the cell layout optimization problem [72]. It is also shown in [72] that the equitable efficient solutions of MOP (4.1) may be computed as the Pareto efficient solutions of the following reformulated multiobjective problem

$$\min_{\mathbf{x} \in X} \quad \left[\overline{\theta}_q(\mathbf{x})\right]_{q=1,\ldots,N}, \tag{4.36}$$

where $\overline{\theta}_q(\mathbf{x})$ is the sum of the q largest components of the vector $\mathbf{f}(\mathbf{x})$ evaluated for some $\mathbf{x} \in X$. The objectives $\overline{\theta}_q(\mathbf{x})$ may be computed as an optimal value of the following linear program (see

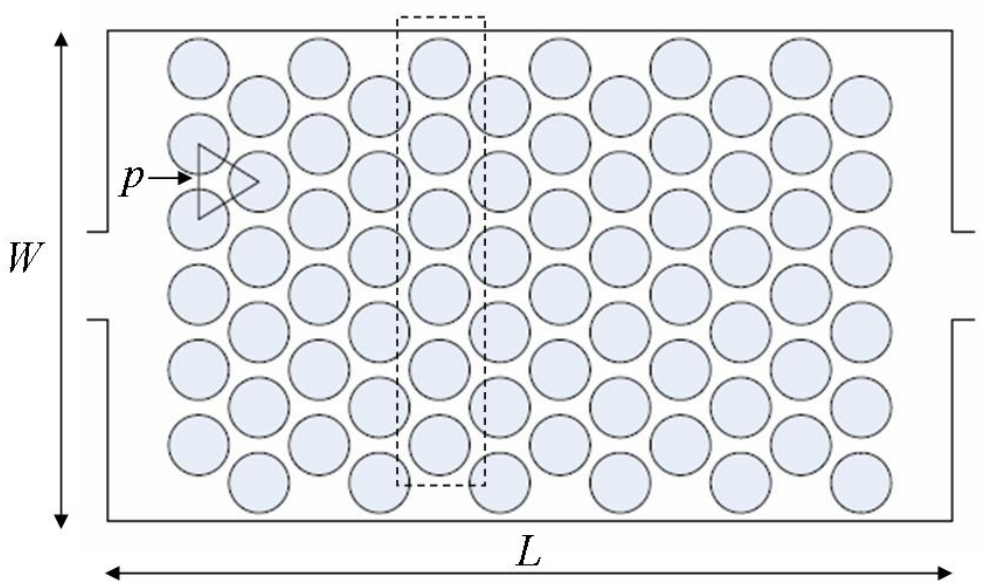

Figure 4.4: Cell layout in the battery module

[24, 72] for details)

$$\begin{aligned}
\overline{\theta}_q(\mathbf{x}) = \min_{z_q, t_q, d_{q,i}} \quad & z_q \\
\text{s.t.} \quad & z_q \in \mathbb{R}, \quad t_q \in \mathbb{R}, \quad d_{q,i} \geq 0 \\
& \qquad\qquad\qquad\qquad i = 1, \ldots, N \\
& z_q = q\, t_q + \sum_{i=1}^{N} d_{q,i} \\
& t_q + d_{q,i} \geq \left(T_i(\mathbf{x}_c) - T_o\right)^2, \\
& \qquad\qquad\qquad\qquad i = 1, \ldots, N
\end{aligned} \tag{4.37}$$

where t_q and $d_{q,i}$, $q, i = 1, \ldots, N$, are auxiliary variables. Multiobjective problem (4.35) is thus transformed into multiobjective problem (4.36), whose Pareto solutions are equitable for the original problem (4.35) with the identifications $f_i(\mathbf{x}) = \left(T_i(\mathbf{x}_c) - T_o\right)^2$, $X = X_c$, and $\mathbf{x} = \mathbf{x}_c$. In effect, equitable efficient solutions can be obtained by the algorithms traditionally used for finding Pareto efficient solutions.

To summarize, the bilevel formulation of the vehicle design problem is given by

$$\begin{aligned} \min_{\mathbf{x}_v \in X_v} \quad & \begin{bmatrix} J_G, & A, & -U \end{bmatrix} \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) = \mathbf{0} \end{aligned} \tag{4.38}$$

(U is negated because it is to be maximized) and

$$\begin{aligned} \min_{\mathbf{x}_c \in X_c} \quad & \left[\overline{\theta}_q(\mathbf{x}_c) \right]_{q=1,\ldots,N} \\ \text{s.t.} \quad & \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) = \mathbf{0}, \end{aligned} \tag{4.39}$$

where

$$\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) := \begin{bmatrix} L_c - L_v \\ W_c - W_v \end{bmatrix}.$$

Problems (4.38) and (4.39) belong to different disciplines of automotive design. The objective functions of problem (4.38) come from dynamics and safety while the objective functions of (4.39) come thermodynamics. Two independent research efforts have been undertaken to propose algorithms for solving these problems separately. Subproblem (4.38) is solved using a genetic algorithm since, as known from the literature, packaging problems are highly multimodal and not suitable for gradient-based algorithms [68]. An archive-based micro genetic algorithm (AMGA) implemented in JAVA is used. The voxelization required for the intersection constraint is conveniently implemented in JAVA as well. The battery design problem (4.39) instead is solved using a sequential quadratic programming (SQP) algorithm since the layout is forced over a triangular grid. The problem is implemented in Matlab which provides the SQP algorithm that is tied to the battery model developed in Matlab/Simulink [24].

In the current work, we make use of these available methods and their implementations to address the packaging problem with the proposed bilevel optimization approach accounting for multiple design disciplines and teams. While the conditions under which MODA generates efficient designs are stated in Section 4.3.2, the application of MODA to the vehicle design problem requires problem-specific fine tuning as described in Section 4.5.

4.5 Application of MODA to the packaging optimization problem

Algorithm MODA is applied to the vehicle design problem with the following identifications.

$$X = X_1 \times X_2 \quad (m = 2)$$

$$X_1 = X_v, \quad X_2 = X_c$$

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2], \quad \mathbf{x}_1 = \mathbf{x}_v, \quad \mathbf{x}_2 = \mathbf{x}_c$$

$$f(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{w}_v^T \cdot \mathbf{f}_v(\mathbf{x}_v) + \mathbf{w}_c^T \cdot \mathbf{f}_c(\mathbf{x}_c)$$

$$\mathbf{f}_v(\mathbf{x}_v) = \Big[J_G(\mathbf{x}_v), \quad A(\mathbf{x}_v), \quad U(\mathbf{x}_v) \Big]$$

$$\mathbf{f}_c(\mathbf{x}_c) = [\, z_q(\mathbf{x}_c) \,]_{q=1,\ldots,N}$$

$$\mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \begin{bmatrix} L_c - L_v \\ W_c - W_v \end{bmatrix}$$

The intra-subproblem weight vectors, $\mathbf{w}_v$ and $\mathbf{w}_c$, and the inter-subproblem weights, α_v and α_c, are specified later.

The application of MODA to the vehicle design problem needs to be subject to the following considerations.

1. The use of metaheuristic approaches such as genetic algorithms for solving the vehicle level problem introduces uncertainty and inexactness. Although this problem is never perfectly addressed, its ill-effect on the performance of the BCD method can be mitigated by accepting vehicle level updates under the condition that such updates result in AiO improvement of the objective function for problem (4.17). Otherwise, the update is rejected and the vehicle-level solver is called repeatedly until this condition is met.

2. AiO improvement for (4.17) resulting from a vehicle-level update cannot be meaningfully tested immediately after a method of multipliers update of $\mathbf{v}$ and μ. Therefore, updates of $\mathbf{v}$ and μ are followed immediately with a battery level update.

3. The presence of inexact minimization also requires that the global equality constraints $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ and the penalty parameter μ be carefully scaled to avoid feasible but non-efficient convergence

for problem (4.6). Rescaling of the constraints takes the form $s \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) = \mathbf{0}$ where $s > 0$ is a real-valued scalar. The use of this rescaled consistency constraint in problem (4.17) results in the formulation of the equivalent problem

$$\begin{aligned} \min \quad & \alpha_v \mathbf{w}_v^T \cdot \mathbf{f}_v(\mathbf{x}_v) + \alpha_c \mathbf{w}_c^T \cdot \mathbf{f}_c(\mathbf{x}_c) \\ & + \mathbf{v}^T \cdot s \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) \\ & + \frac{\mu}{2} \left[s \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) \right]^T \left[s \cdot \mathbf{h}(\mathbf{x}_v, \mathbf{x}_c) \right] \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v, \quad \mathbf{x}_c \in X_c. \end{aligned} \tag{4.40}$$

Subproblem solvers do not need direct knowledge of this rescaling. The statement of problem (4.40) suggests that the constraint rescaling may be affected by passing $\mathbf{v}$ and μ to each subproblem in the form $\mathbf{v} \leftarrow s\,\mathbf{v}$, $\mu \leftarrow s^2\,\mu$.

4. If subproblems (4.15) and (4.16) are assumed to have no direct knowledge of α_v and α_c, respectively, then the effect of α_v and α_c can be encoded into the parameters $\mathbf{v}$ and μ. This is accomplished through the reformulations of (4.15) and (4.16) taking the form

$$\begin{aligned} \min_{\mathbf{x}_v} \quad & f_v(\mathbf{x}_v) + \mathbf{v}_v^T \cdot \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) + \frac{\mu_v}{2} \left\| \mathbf{h}(\mathbf{x}_v, \overline{\mathbf{x}}_c) \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_v \in X_v \end{aligned} \tag{4.41}$$

and

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & f_c(\mathbf{x}_c) + \mathbf{v}_c^T \cdot \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) + \frac{\mu_c}{2} \left\| \mathbf{h}(\overline{\mathbf{x}}_v, \mathbf{x}_c) \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_c \in X_c, \end{aligned} \tag{4.42}$$

where $\mathbf{v}_v = \frac{\mathbf{v}}{\alpha_v}, \mathbf{v}_c = \frac{\mathbf{v}}{\alpha_c}$ and $\mu_v = \frac{\mu}{\alpha_v}, \mu_c = \frac{\mu}{\alpha_c}$.

An implementation of these ideas is depicted in Fig. 4.5.

4.5.1 Numerical results

Pareto efficient designs of the bilevel problem (4.6) are computed under the following scenarios implied by the assumption that the design variable $N_{cols} \in \{9, 12, 18\}$ in subproblem (4.10) is treated as a fixed parameter. (Thus, N_{rows} is also fixed due to the constraint

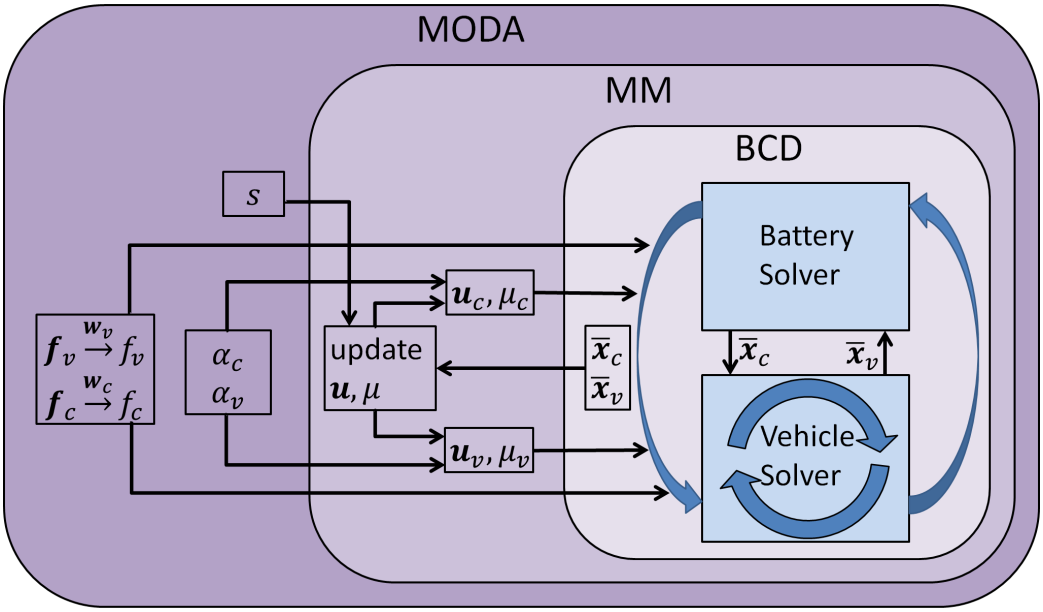

Figure 4.5: Applying MODA to the vehicle design problem

$N_{cols}N_{rows} = 72$.) Three scenarios (referred to as N_{cols}-scenarios) are considered corresponding to battery box aspect ratios induced by the arrangement of battery cells into 9 columns, 12 columns, and 18 columns.

Weights are specified as the *intra*-subproblem weight vectors, $\mathbf{w}_v$ and $\mathbf{w}_c$, and a scalar $\beta > 0$ is used to construct the *inter*-subproblem weights

$$\alpha_v = \frac{1}{1+\beta}, \quad \alpha_c = \frac{\beta}{1+\beta},$$

whose purpose is to allocate relative importance between the two subproblems. The inter-subproblem weights, α_v and α_c, may be constructed to reflect the designer's preference.

For each N_{cols}-scenario, one vector of weights

$$\mathbf{w}_v = [0.01, 50, 50]$$

is considered at the vehicle level. This value for $\mathbf{w}_v$ is for the purposes of favorably scaling the problem and does not correspond to a designer's preference.

For each N_{cols}-scenario, three vectors of weights $\mathbf{w}_c$ of length N_{cols} are considered having

Table 4.2: Solutions computed using MODA (9 columns, 8 rows)

Battery level weight: $\mathbf{w}_c^1$

β	1	10	100	1000
f_v	-32.822	-376.019	-255.698	-207.100
f_c	35.906	5.920	0.864	0.059
J_G	92823.050	77430.460	80928.470	89032.060
A	0.000	1.000	0.000	1.000
U	19.221	24.006	21.300	22.947
p	1.236	1.057	1.095	1.117
L_v	391.754	351.954	364.546	360.201
L_c	393.115	350.982	360.032	365.067
W_c	358.274	304.648	312.963	322.615
W_v	357.160	305.468	316.572	322.749
u_1	-6.001	28.950	47.504	-0.985
u_2	1.348	5.867	51.840	52.069
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^2$

β	1	10	100	1000
f_v	-660.469	-360.716	-293.274	260.857
f_c	57.098	5.467	2.390	0.078
J_G	74841.670	95503.230	79943.660	95781.840
A	1.000	0.000	2.000	0.000
U	29.177	26.315	23.852	13.939
p	1.015	1.078	1.150	1.118
L_v	339.914	351.809	370.395	364.494
L_c	341.122	355.902	372.958	365.315
W_v	294.315	314.821	336.940	322.606
W_c	293.372	311.505	332.430	323.054
u_1	24.893	-69.704	-18.705	-10.833
u_2	5.083	95.716	-85.468	98.262
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^3$

β	1	10	100	1000
f_v	-439.028	-182.589	6.555	-249.299
f_c	2.123	1.863	0.116	0.125
J_G	78804.310	83953.760	89331.580	95793.220
A	1.000	0.000	1.000	1.000
U	25.540	20.442	18.734	25.144
p	1.097	1.099	1.121	1.116
L_v	358.957	359.492	365.037	361.614
L_c	360.476	360.801	366.107	364.868
W_v	318.358	318.415	322.891	317.770
W_c	317.116	317.515	324.025	322.504
u_1	-12.982	-57.624	-154.377	45.549
u_2	13.311	66.812	34.555	67.426
μ	128.000	128.000	128.000	128.000

Table 4.3: Solutions computed using MODA (12 columns, 6 rows)

Battery level weight: $\mathbf{w}_c^1$

β	1	10	100	1000
f_v	-563.563	-5.141	-357.208	-331.942
f_c	40.688	3.286	1.085	0.102
J_G	70667.080	72971.800	80529.730	91776.540
A	0.000	1.000	1.000	1.000
U	25.405	15.696	24.249	25.993
p	1.246	1.177	1.094	1.120
L_v	499.188	485.767	455.456	468.999
L_c	505.605	483.326	456.335	464.810
W_v	284.799	256.578	241.882	251.612
W_c	275.389	260.188	241.771	247.554
u_1	36.349	46.953	46.711	20.438
u_2	-57.962	-85.208	24.411	2.739
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^2$

β	1	10	100	1000
f_v	-325.756	-458.535	-267.731	-303.221
f_c	4.676	7.703	0.289	0.177
J_G	99526.140	71234.520	82173.020	76344.370
A	1.000	1.000	1.000	1.000
U	27.419	24.417	22.788	22.332
p	1.085	1.076	1.123	1.119
L_v	453.996	450.329	469.991	465.319
L_c	453.584	450.614	465.785	464.448
W_v	239.254	238.006	243.579	249.584
W_c	239.894	237.868	248.219	247.307
u_1	4.002	30.085	-45.957	6.261
u_2	-2.012	17.036	36.253	-15.480
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^3$

β	1	10	100	1000
f_v	-492.996	-299.608	-391.409	-486.777
f_c	9.905	4.555	0.410	0.219
J_G	105874.700	69555.240	82402.070	69050.710
A	1.000	1.000	1.000	1.000
U	32.034	20.902	25.308	24.545
p	1.166	1.092	1.113	1.118
L_v	482.218	457.171	463.061	466.849
L_c	479.759	455.549	462.337	464.184
W_v	254.149	238.650	242.595	242.803
W_c	257.754	241.235	245.866	247.127
u_1	-2.585	8.915	26.695	-19.723
u_2	-2.761	42.123	10.898	44.862
μ	128.000	128.000	128.000	128.000

Table 4.4: Solutions computed using MODA (18 columns, 4 rows)

Battery level weight: $\mathbf{w}_c^1$

β	1	10	100	1000
f_v	-381.987	-178.959	-198.840	-210.174
f_c	9.443	4.222	6.089	0.217
J_G	82449.230	88616.670	94997.310	90771.990
A	1.000	1.000	2.000	0.000
U	25.129	22.302	24.974	22.358
p	1.046	1.181	1.060	1.116
L_v	626.270	691.875	633.499	661.222
L_c	625.697	693.344	632.358	660.779
W_v	158.204	185.554	159.469	164.043
W_c	160.071	180.748	162.107	170.794
u_1	18.119	-5.603	171.838	43.624
u_2	-51.428	-5.726	26.132	33.805
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^2$

β	1	10	100	1000
f_v	453.512	-306.231	-145.122	-470.798
f_c	7.039	4.141	0.631	0.589
J_G	91212.830	79380.660	89841.730	89010.340
A	1.000	0.000	1.000	2.000
U	10.171	22.001	21.870	29.216
p	1.156	1.144	1.120	1.116
L_v	681.656	674.345	663.654	658.613
L_c	680.403	674.686	662.524	660.785
W_v	172.769	176.173	170.839	170.734
W_c	176.792	175.045	171.327	170.796
u_1	37.650	-9.438	0.237	27.354
u_2	-128.764	-27.447	6.704	2.046
μ	128.000	128.000	128.000	128.000

Battery level weight: $\mathbf{w}_c^3$

β	1	10	100	1000
f_v	8.105	-315.012	-56.439	-137.478
f_c	59.288	0.749	0.892	0.750
J_G	83808.680	77789.670	82140.090	80704.000
A	1.000	0.000	1.000	1.000
U	17.599	21.858	18.556	19.889
p	1.047	1.117	1.122	1.117
L_v	627.401	660.511	665.146	658.588
L_c	625.990	661.347	663.430	661.042
W_v	155.530	173.699	172.065	167.233
W_c	160.161	170.968	171.605	170.875
u_1	7.872	5.933	18.798	-40.661
u_2	28.897	-19.857	-39.989	-40.121
μ	128.000	128.000	128.000	128.000

Table 4.5: Trajectory of computation for 9 columns, 8 rows, $\mathbf{w}_c = \mathbf{w}_c^1$, $\beta = 1$

μ	p	f_v	f_c	L_v	L_c	W_v	W_c
0.000	1.119	-253.285	0.043	499.205	365.528	341.483	323.314
2.000	1.369	-253.285	69.552	499.205	424.481	341.483	395.643
2.000	1.205	-141.505	28.118	425.145	385.920	323.356	348.333
8.000	1.220	-141.505	31.805	425.145	389.339	323.356	352.528
8.000	1.236	-32.822	35.892	391.754	393.103	358.274	357.145

the form

$$\mathbf{w}_c^1 = \begin{bmatrix} 1 & 2^{-1} & 2^{-2} & \dots & 2^{-(N_{cols}-2)} & 2^{-(N_{cols}-1)} \end{bmatrix}$$

$$\mathbf{w}_c^2 = \begin{bmatrix} 1 & 1 & \dots & \dots & 1 & 1 \end{bmatrix}$$

$$\mathbf{w}_c^3 = \begin{bmatrix} 2^{-(N_{cols}-1)} & 2^{-(N_{cols}-2)} & \dots & 2^{-2} & 2^{-1} & 1 \end{bmatrix},$$

where $\mathbf{w}_c^i$ are subsequently normalized so that $\left\|\mathbf{w}_c^i\right\|_1 = 1$ for $i = 1, 2, 3$. The weight vectors $\mathbf{w}_c^i$, $i = 1, 2, 3$, applied to problem (4.39) are used to compute the equitable efficient battery designs and may reflect the designer's preference.

For each N_{cols}-scenario and each battery level weight vector $\mathbf{w}_c^i$, $, i = 1, 2, 3$, four scalar values β_j, $j = 1, 2, 3, 4$, are considered: $\beta_1 = 1$, $\beta_2 = 10$, $\beta_3 = 100$, and $\beta = 1000$. These β values determine the relative importance between vehicle subproblem (4.9) and battery component subproblem (4.10) in the following manner.

$$\mathbf{w}^{i,j} = \left[\frac{1}{1+\beta_j}\mathbf{w}_v,\ \frac{\beta_j}{1+\beta_j}\mathbf{w}_c^i\right], \qquad \begin{matrix} i = 1, 2, 3, \\ j = 1, 2, 3, 4 \end{matrix} .$$

There are thus twelve all-in-one weight vectors $\mathbf{w}^{i,j}$ corresponding to (i, j) pairs for $i = 1, 2, 3$ and $j = 1, 2, 3, 4$. The inter-subproblem weights determined by β may also be chosen to reflect the designer's preference.

With the above initializations, the results obtained using MODA (Fig. 4.5) are given in Tabs. 4.2–4.4. Each table corresponds to an N_{cols}-scenario and battery level weight $\mathbf{w}_c^i$, $i = 1, 2, 3$, combination. In each table, the four right-most columns correspond to the use of a β_j value for $j = 1, 2, 3, 4$.

The rows of each table report the following computed values.

1. $f_v = \mathbf{w}_v^T \mathbf{f}_v$ is the weighted-sum objective value of problem (4.15).

2. $f_c = \mathbf{w}_c^T \mathbf{f}_c$ is the weighted-sum objective value of problem (4.16).

3. J_G, A, and U are the vehicle-level objective values.

4. p is the battery subproblem spacing factor.

5. L_v and L_c are the battery box component lengths in the vehicle subproblem and the battery subproblem, respectively.

6. W_v and W_c are the battery box component widths in the vehicle subproblem and the battery subproblem, respectively.

7. u_1, u_2, μ are the values for the Lagrange multiplier $\mathbf{v}$ and penalty coefficient μ.

Some interesting considerations can be made in support of the application of MODA for the distributed computation of efficient solutions of the bilevel packaging problem. The algorithm captures the negotiation between the design teams in the design process. This negotiation is demonstrated by different values of the aggregated objectives f_v and f_c that are obtained as the inter-subproblem weights are varied: as β increases from 1 to 1000, f_v increases and f_c decreases (check Tabs. 4.2–4.4). This is possible due to the consistency constraint that is enforced and satisfied, within engineering tolerances, at optimality (compare L_v and L_c to W_v and W_c in Tabs. 4.2–4.4), which reflects the agreement between the teams that is required by the design of the decomposed problem.

The results obtained for $\beta = 1000$, that is, when maximum importance is given to the battery level, show the tradeoff exploration capabilities of MODA. In this case, each of the optimal solutions present the same cell spacing $p = 1.118$, with some oscillations due to numerical tolerances. This is the optimal (equitable) value that is reported for *each aspect ratio* considered in [24], where the battery problem was formulated and solved without considering the vehicle level.

The effect of the bilevel interaction on the vehicle layouts obtained with a 9×8 aspect ratio and $\mathbf{w}_c = \mathbf{w}_c^2$ is shown in Figs. 4.6 and 4.7 for $\beta = 1$ and $\beta = 1000$, respectively. Two different solutions in terms of cell spacing p resulted in different battery box dimensions, 340×294 mm and 365×322 mm, and consequently in two distinct vehicle layouts. Expected tradeoffs are noticed, with deterioration in compactness and survivability and improvement in accessibility and

cell temperature uniformity (the latter occurs because for $\beta = 1000$ the battery solution approaches the equitable one).

The effect of the aspect ratio also demonstrates the capability of MODA to handle bilevel tradeoffs. The layout with the 18×4 battery, $\mathbf{w}_c = \mathbf{w}_c^2$, and $\beta = 1000$ represented in Fig. 4.8 can be compared to the one in Fig. 4.7 with an aspect ratio of 9×8. As the number of cell columns increases the cell temperature becomes less uniform. The different battery dimensions, 365×322 mm and 660×170 mm, affect the vehicle layout; in this case, the accessibility worsens but compactness and accessibility improve.

When analyzing the computations resulting from MODA, the following properties of the scalarized vehicle level problem (4.15) should be considered due to their contradiction of the assumptions stated in Proposition 12:

- lack of convexity of the feasible set X_v;
- multimodality of f_v implying that:
  - the tradeoffs cannot be properly explored through the weighted-sum scalarization;
  - the method of multipliers updates (4.22) computed with BCD are not guaranteed to be globally optimal;
- optimality that is not guaranteed when genetic algorithms are utilized.

Although MODA has been successfully applied to modeling the bilevel negotiation and to achieving satisfaction of the consistency constraints within engineering tolerances, future improvement in the computed designs may be obtained by adapting the scalarized vehicle level problem (4.15) so that its properties align more closely with the assumptions stated in Proposition 12.

4.6 Conclusion

The Multiobjective Decomposition Algorithm (MODA) proposed in this paper is motivated by the design of engineering systems involving different disciplines and subsystems which require a high level of specialization of the collaborating teams. MODA reflects the features of this design scenario, i.e., the decomposition and multiple criteria, in the case of bilevel problems. The negotiation

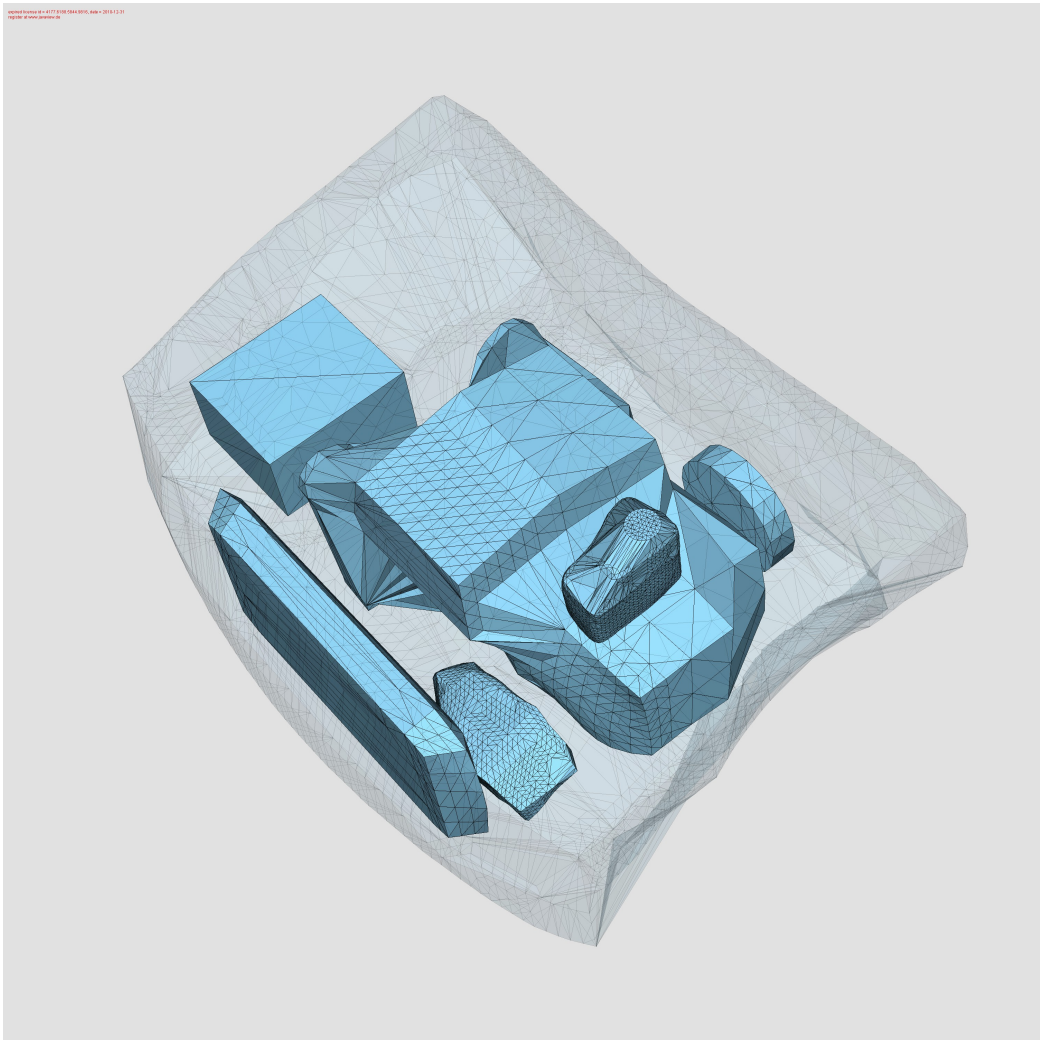

Figure 4.6: Vehicle layout with 9×8 aspect ratio, $\mathbf{w}_c = \mathbf{w}_c^2$, and $\beta = 1$

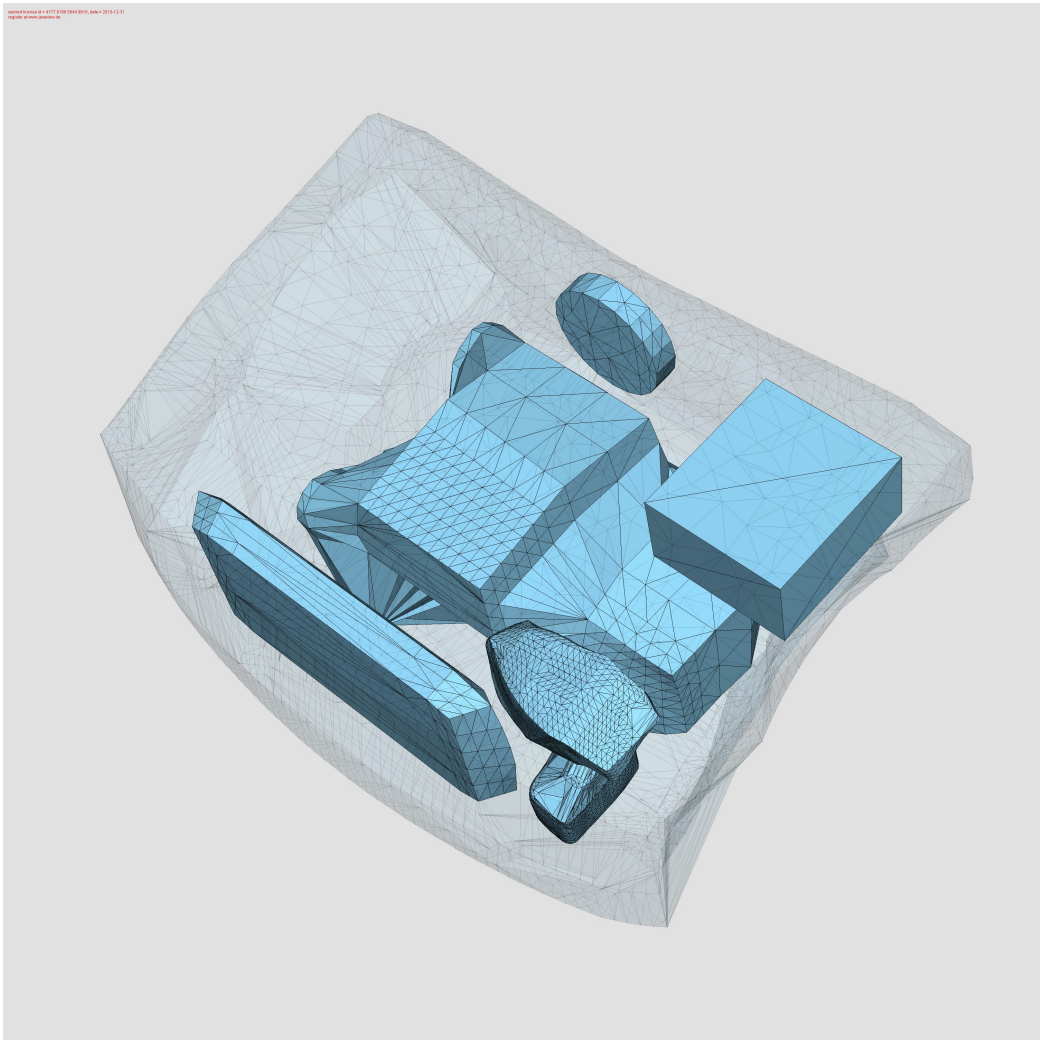

Figure 4.7: Vehicle layout with 9×8 aspect ratio, $\mathbf{w}_c = \mathbf{w}_c^2$, and $\beta = 1000$

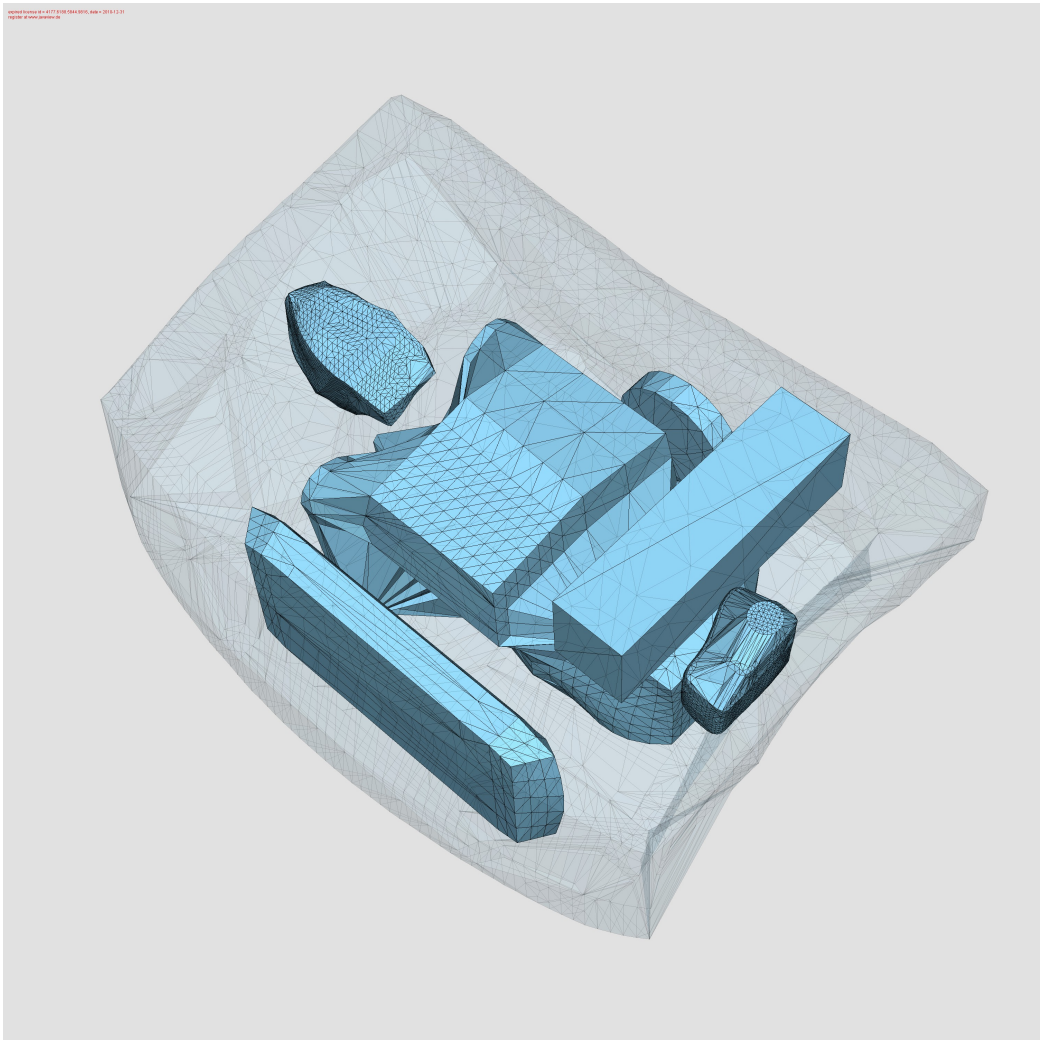

Figure 4.8: Vehicle layout with 18×4 aspect ratio, $\mathbf{w}_c = \mathbf{w}_c^2$, and $\beta = 1000$

between the two design teams generates tradeoffs that MODA captures as demonstrated by the engineering example. The design of a hybrid vehicle layout is addressed at both the vehicle and battery levels. The vehicle layout is designed considering that the battery, one of the main components of hybrid vehicles, is optimized with respect to its design criteria. The numerical results demonstrate the capability of MODA to obtain Pareto efficient solutions through distributed computation and decomposition. The tradeoff exploration occurs by means of the weighted-sum method that, due to the use of intra- and inter-subproblem weights, captures the tradeoffs within the subproblems (i.e., internal tradeoffs occurring at the vehicle and battery problems) and between the subproblems (i.e., tradeoffs between the vehicle and battery subproblems).

In future research, improvement in the bilevel computations using MODA will be realized with the use of a scalarization method that addresses multimodality in the objective functions while maintaining the decomposable structure of the problem. Extensions to multilevel problems, and possibly to nonhierarchically decomposable problems, are encouraged by the results presented in this paper and would broaden the application of MODA to other MDO problems.

Acknowledgments

The views presented in this work do not necessarily represent the views of our sponsors, the Automotive Research Center, a center of excellence of the US Army TACOM, and the National Science Foundation, grant number CMMI-1129969, whose support is greatly appreciated.

Chapter 5

A comparison of t-domain and s-domain least squares parameter estimation for modeling properties of viscoelastic materials

[This chapter contains the contents of a paper submitted for review in the journal: *Problems of Machine Building and Automatization* on May 3rd, 2013 titled "A comparison of t-domain and s-domain least squares parameter estimation for modeling properties of viscoelastic materials"; authors are B. Dandurand, I. Viktorova, and S. Alekseeva.]

5.1 Introduction

Let $m(\mathbf{p}, t) : \mathbf{R}^n \times [0, \infty) \to \mathbb{R}$ be a parameterized time dependent stress model, where $t \in [0, \infty)$ denotes the passage of time in hours, and $\mathbf{p}$ is a vector of real-valued parameters $\mathbf{p} = [p_1, \ldots, p_n]$ taking values from some set $P \subseteq \mathbb{R}^n$. Let the time-dependent experimental stress observations be denoted by $\mathbf{y} = [y_1, \ldots, y_N]$, where each y_i is the experimental stress observation for some time $t_i > 0$ for $i = 1, \ldots, N$. The problem of computing optimal-fitting parameters may

be formulated as the following least squares problem (LSP)

$$\min_{\mathbf{p}\in P} \sum_{i=1}^{N} \left(m(\mathbf{p}, t_i) - y_i\right)^2 \tag{5.1}$$

or alternately by

$$\min_{\mathbf{p}\in P} \sum_{i=1}^{N} \left(\frac{m(\mathbf{p}, t_i) - y_i}{y_i}\right)^2,$$

Certain models $m(\mathbf{p}, t)$, such as the model described in Section 5.3 do not yield computationally stable problems of form (5.1) [25]. This motivates the exploration of another approach for computing optimal parameters $\mathbf{p}$ when the model $m(\mathbf{p}, t)$ simplifies under certain integral transforms such as the Laplace transform. To develop this approach, first compute a regression function $r(t) : [0, \infty) \to \mathbb{R}$ for the experimental data $\mathbf{y}$. Regression function $r(t)$ is assumed to have a Laplace transform $R(s) := \mathcal{L}\{r(t)\}$ that is given in closed form. Once the Laplace transform is also computed for the model $M(\mathbf{p}, s) := \mathcal{L}\{m(\mathbf{p}, t)\}$, the Laplace domain least squares problem (LDLSP) is given by

$$\min_{\mathbf{p}\in P} \sum_{s\in S_N} |M(\mathbf{p}, s) - R(s)|^2 \Delta(s) \tag{5.2}$$

where $S_N \subset S$ is a finite set of complex-valued sample points taken from a sample region $S \subset H := \{s \in \mathbb{C} : Re(s) > 0\}$. $\Delta : S_N \to \mathbb{R}_>$ is a function defined on S_N that assigns a weight to each term of (5.2) corresponding to each $s \in S_N$.

This work contributes a mathematical foundation for the approach described in [115] by which to compare the use of the LSP and the LDLSP. Each of the minimization problems (5.1) and (5.2) is viewed as the minimization of distance between two functions from the same underlying space $C[a, b]$ of continuous functions $f : [a, b] \to \mathbb{R}$. However, the notions of distance on the space $C[a, b]$ are different for the two problems. Section 5.2 defines these two notions of distances in terms of norms on $C[a, b]$. It is shown that the norm associated with problem (5.1) can serve as a bound for the norm associated with problem (5.2), but not vice versa, and so the two norms are not equivalent in the sense described in [92]. The implication of this fact for using problem (5.2) to compute the optimal parameters for problem (5.1) is briefly discussed. Section 5.3 describes the use of problem (5.2) for the computation of optimal parameter estimates for modeling certain properties of viscoelastic nanocomposite materials. The specification of S_N and $\Delta(s)$ for problem (5.2) in this application is motivated from the development of Section 5.2. Section 5.4 summarizes the work and

its relation to the results computed in Section 5.3.

5.2 Two optimization problems for obtaining optimal parameter estimates

The norm associated with the distance being minimized in problem (5.1) is identified with the standard norm for the $L^2[a,b]$ space induced from the inner product

$$\langle f, g\rangle = \int_a^b f(t)\overline{g(t)}dt,$$

defined for each $f, g \in L^2[a,b]$. The induced norm is given by

$$\|f\|_2 = \sqrt{\langle f, f\rangle},$$

The minimization of discrepancies between model $m(\mathbf{p},t)$ and regression $r(t)$ may thus be stated as the minimization problem

$$\min_{\mathbf{p}\in P} \|m(\mathbf{p},t) - r(t)\|_2^2, \tag{5.3}$$

of which problem (5.1) is viewed as a computationally convenient approximation.

In order to describe the distance being minimized between continuous functions $f, g \in C[a,b]$ in problem (5.2), the following inner product

$$\langle f, g\rangle_S := \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d F_{a,b}\left(\rho\, \mathrm{e}^{\mathrm{i}\theta}\right) \overline{G_{a,b}\left(\rho\, \mathrm{e}^{\mathrm{i}\theta}\right)} \rho d\rho \right) d\theta \tag{5.4}$$

is defined on $C[a,b] \times C[a,b]$ where

$$F_{a,b}(s) := \int_a^b e^{-st} f(t)dt$$
$$G_{a,b}(s) := \int_a^b e^{-st} g(t)dt$$

are "truncated" Laplace transforms of f and g defined on

$$H := \{s \in \mathbb{C} : Re(s) > 0\},$$

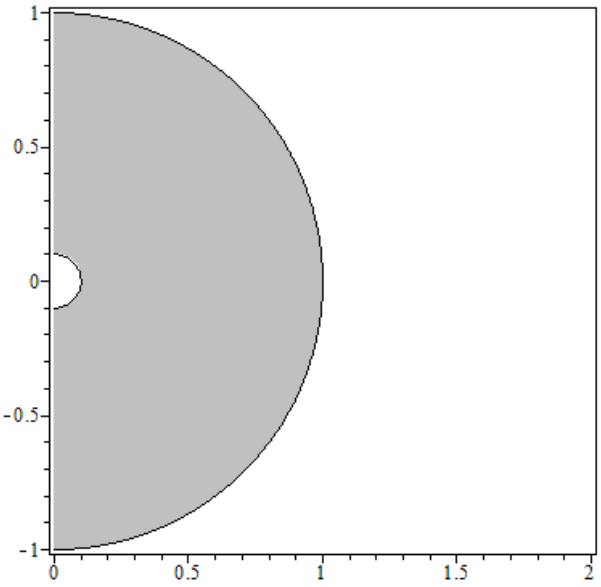

Figure 5.1: Region $S \subset H$

and $\rho\, e^{i\theta}$ is the polar form of s with the region of integration

$$S := \left\{ s \in H \ : \ c < Re(s) < d, \ -\frac{\pi}{2} < arg\ s < \frac{\pi}{2} \right\}$$

depicted in Figure 5.1. This region S is a specification of the region S mentioned in the discussion surrounding the introduction of problem (5.2). The restriction of the integration over $S \subset H$ is motivated by the theory of analytical continuation, which specializes to the present context to imply that if $F : H \to \mathbb{C}$ and $G : H \to \mathbb{C}$ are analytic functions defined on H, and if $F(s) = G(s)$ for all $s \in S \subset H$, then $F = G$ holds over all of H [19, 21].

Proposition 14 verifies that the functional defined in (5.4) is, in fact, an inner product.

Proposition 14. *The functional* $\langle \cdot, \cdot \rangle_S : C[a,b] \times C[a,b] \to \mathbb{C}$ *defined in equation* (5.4) *is an inner product on* $C[a,b]$.

Proof. The homogeneous property $\langle \mu f, g \rangle_S = \mu \langle f, g, \rangle_S$ and the additivity property $\langle f, g + h \rangle_S = \langle f, g \rangle_S + \langle f, h \rangle_S$ follow easily from elementary additive and multiplicative properties of integrals. For showing the remaining inner product properties, we view $F_{a,b}$ as the Laplace transform of the

extended domain function $\hat{f} : [0, \infty) \to \mathbb{R}$ defined by

$$\hat{f}(t) = \begin{cases} f(t) & \text{if} \quad a \leq t \leq b \\ 0 & \text{otherwise} \end{cases},$$

and $G_{a,b}$ is viewed as the Laplace transform of a similarly extended function $\hat{g}$ of g. Since $F_{a,b}$ and $G_{a,b}$ are the Laplace transforms of the piecewise continuous function $\hat{f}$ and $\hat{g}$, respectively, so $F_{a,b}$ and $G_{a,b}$ are analytic in H (see Chapter 7, Theorem 1 of [19]), and therefore continuous.

Writing $s = \rho\, \mathrm{e}^{\mathrm{i}\theta}$, then for each fixed $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$, the expressions $\sqrt{\rho} F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})$ and $\sqrt{\rho} G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})$ may be used as definitions for continuous, complex-valued functions of ρ where $\rho \in [c, d]$ is real-valued. Applying the standard inner product for the space $(C_{\mathbb{C}}[c, d], \|\cdot\|_2)$ on these functions, it follows that

$$\int_c^d F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho d\rho = \overline{\int_c^d G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho d\rho} \tag{5.5}$$

for each fixed $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$. Substituting the right-hand side of (5.5) for the inner integral in (5.4), it immediately follows that

$$\begin{aligned} \langle f, g \rangle_S &= \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho d\rho \right) d\theta \\ &= \overline{\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho d\rho \right) d\theta} \\ &= \overline{\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d G_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho d\rho \right) d\theta} \\ &= \overline{\langle g, f \rangle}_S \end{aligned}$$

and so the Hermitian/Conjugate symmetry is established.

The nonnegativity property $\langle f, f \rangle_S \geq 0$ for each $f \in C[a, b]$ follows from the nonnegativity of the integrand

$$F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta}) \overline{F_{a,b}(\rho\, \mathrm{e}^{\mathrm{i}\theta})} \rho.$$

To show the implication $\langle f, f \rangle_S = 0$ if and only if $f \equiv 0$, consider the continuity of the integrand of (5.4), which follows from the continuity of $F_{a,b}$. With the nonnegativity of the same integrand, the right-hand side of (5.4) is zero if and only if $F_{a,b}$ is the zero function. Since $F_{a,b} \equiv 0$ if and only if $\hat{f} \equiv 0$ (and thus also $f \equiv 0$) (see Chapter 6, Theorem 7 of [19]), then $\langle f, f \rangle_S = 0$ if and only if $f \equiv 0$

as was to be shown. Thus, $\langle f, g\rangle_S$ is an inner product on the space $C[a, b]$. □

Let $L^S := (C[a, b], \|\cdot\|_S)$ be the topology defined on $C[a, b]$ with the norm $\|\cdot\|_S : C[a, b] \to \mathbb{R}$ defined by $\|f\|_S = \sqrt{\langle f, f\rangle_S}$. The LDLSP is then approximated with the following minimization problem

$$\min_{\mathbf{p}\in P} \|m(\mathbf{p}, \cdot) - r\|_S^2 \tag{5.6}$$

corresponding to the minimization of distance between functions $m(\mathbf{p}, \cdot)$ and r in L^S.

Equivalence between problem (5.6) and problem (5.3) is understood in terms of the norm equivalence, as defined in [92], of the two norms $\|\cdot\|_2$ and $\|\cdot\|_S$ associated with L^2 and L^S, respectively. The norm equivalence is characterized by the existence of $\ell, u > 0$ satisfying $0 < \ell < u < \infty$ such that

$$\ell \, \|f\|_2 \leq \|f\|_S \leq u \, \|f\|_2 \quad \text{for all} \quad f \in C[a, b] \tag{5.7}$$

The values of bounding parameters $\ell > 0$, $u > 0$, when they exist, provide a measure for how well the optimal solutions for problem (5.3) approximate the optimal solutions for problem (5.6), and vice versa.

In Proposition 15, the existence of the upper bound u is established, while the lack of lower bound $\ell > 0$ is demonstrated with a counterexample. Thus, the norms are not equivalent. The lack of a lower bound ℓ in (5.7) means that local minimizers $\hat{\mathbf{p}}$ for problem (5.2) may have no connection with minimizers $\mathbf{p}^*$ for problem (5.1). Nevertheless, the existence of upper bound $u \in (0, \infty)$ and the continuity of the norms $\|\cdot\|_2$ and $\|\cdot\|_S$ when viewed as functionals of $\mathbf{p}$ in the context of problems (5.3) and (5.6) suggest that a minimizer $\mathbf{p}^*$ for problem (5.3) lies within some bounded neighborhood of a local minimizer $\hat{\mathbf{p}}$ for problem (5.6).

Proposition 15. *For each closed bounded interval $[a, b]$ and closed bounded region S, there exists $u \in (0, \infty)$ satisfying the upper bound inequality* (5.7).

Proof. Write

$$\begin{aligned}
\|f\|_S^2 &= \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d \left| \int_a^b e^{-\rho\, \mathrm{e}^{\mathrm{i}\theta} t} f(t) dt \right|^2 \rho d\rho \right) d\theta \\
&\leq \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d \left(\int_a^b \left| e^{-\rho\, \mathrm{e}^{\mathrm{i}\theta} t} f(t) \right| dt \right)^2 \rho d\rho \right) d\theta \\
&\qquad\qquad \text{(by Cauchy-Schwartz inequality)} \\
&\leq \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d \left(\int_a^b \left| e^{-\rho\, \mathrm{e}^{\mathrm{i}\theta} t} \right|^2 dt \right) \left(\int_a^b |f(t)|^2 \, dt \right) \rho d\rho \right) d\theta \\
&= \|f\|_2^2 \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d \left(\int_a^b \left| e^{-\rho\, \mathrm{e}^{\mathrm{i}\theta} t} \right|^2 dt \right) \rho d\rho \right) d\theta.
\end{aligned}$$

The upper bound is established by identifying $u = \sqrt{\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \left(\int_c^d \left(\int_a^b \left| e^{-\rho\, \mathrm{e}^{\mathrm{i}\theta} t} \right|^2 dt \right) \rho d\rho \right) d\theta}$, which is finite in value by the compactness of the sets S and $[a, b]$. □

To show there is no $\ell > 0$ satisfying (5.7), consider functions $f \in C[0, 1]$ of form $f(t) = e^{-\gamma t}$, where $\gamma > 0$, and let $S = [c, d]$ only take real values where $0 < c < d < \infty$. We have

$$\|f\|_2^2 = \int_0^1 e^{-2\gamma t} dt = \frac{1 - e^{-2\gamma}}{2\gamma}. \tag{5.8}$$

Furthermore, the value of the norm $\|f\|_S^2$ can be shown to satisfy the bounds

$$\frac{d - c - 2e^{-(\gamma+c)} + e^{-2(\gamma+d)}}{(\gamma + c)(\gamma + d)} \leq \|f\|_S^2 \leq \frac{d - c - 2e^{-(\gamma+d)} + e^{-2(\gamma+c)}}{(\gamma + c)(\gamma + d)}. \tag{5.9}$$

Using the value computed in (5.8) and the value for either bound in (5.9), taking the limit

$$\lim_{\gamma \to \infty} \frac{\|f\|_S}{\|f\|_2} = 0$$

shows that there is no $\ell > 0$ satisfying (5.7) for all $f \in C[0, 1]$.

For computational purposes, problem (5.2) serves to approximate (5.6). In motivating

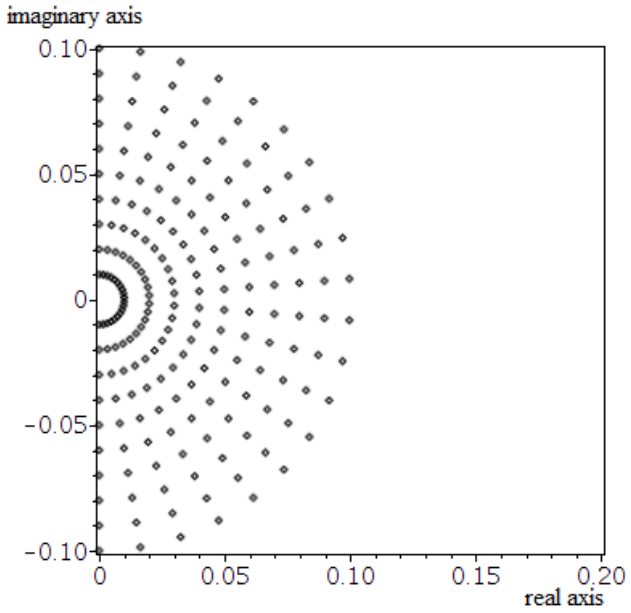

Figure 5.2: Sample region $S_N \subset S$. The individual $s \in S_N$ are depicted with dots.

problem (5.2) as a Riemann sum approximation of problem (5.6), the use of S_N taking the form

$$S_N = \left\{ \begin{array}{c} s \in S \,:\, s = \epsilon + \rho\, \mathrm{e}^{\mathrm{i}\theta}, \quad \rho = c + \frac{j}{n-1}(d-c), \quad \theta = -\frac{\pi}{2} + \frac{k}{m-1}\pi \\ \text{for} \quad \epsilon > 0, \quad j = 0, \ldots, n-1, \quad k = 0, \ldots, m-1 \end{array} \right\} \tag{5.10}$$

together with the use of $\Delta(s) = \rho = |s|$ serves to aptly approximate the role of differential's product $\rho\, d\rho\, d\theta$ in (5.4). Figure 5.2 illustrates the appearance of set S_N having form (5.10).

Additional approximations appear in the use of the actual Laplace transforms $M(\mathbf{p}, s)$ and $R(s)$ in problem (5.2), replacing the truncations $M_{a,b}(\mathbf{p}, s)$ and $R_{a,b}(s)$ as defined for problem (5.6).

5.3 Application for modeling time dependent properties of viscoelastic materials

Viscoelastic materials have memory in the sense that stress applied in the past can affect strain in the present time t. The introduction of time dependence or memory effect leads to the analysis of Volterra's equation of second type [116, 96] that models the dependence of stress as a

functional of strain

$$\varphi\left(\varepsilon(t)\right) = m(\mathbf{p}, t) \tag{5.11}$$

where $\varphi\left(\varepsilon(t)\right) := E\varepsilon$ is a linear response functional of ε (the so-called instantaneous loading diagram) and $m(\mathbf{p}, t)$ models the material stress resulting from the memory effect and is taken to have form

$$m(\mathbf{p}, t) := \sigma(t) + \int_0^t K(\mathbf{p}, t-\tau)\sigma(\tau)d\tau \tag{5.12}$$

In practice, $m(\mathbf{p}, t)$ as defined in (5.12) models the relations between time, stress, and strain successfully for a wide range of materials such as polymers, metals, and composites [96, 111, 122].

One of the most effective and universal kernels $K(\mathbf{p}, t)$ is based on the exponential of arbitrary order function [96]

$$K(\mathbf{p}, t) := \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n \ t^{n(1-\alpha)}}{\Gamma\left[(1-\alpha)(n+1)\right]} \tag{5.13}$$

where $\mathbf{p} = [\alpha, \beta, \lambda]$ is the vector of parameters. The exponential of arbitrary order operators combine several important features [96, 15].

- The initial moment singularity at $t = 0$ is integratable.
- The asymptotic exponential behavior with $t \to \infty$.
- The resolvent operator is the same type of exponential of arbitrary order with a different set of defining parameters.

Using the kernel given in (5.13) together with the assumption that $\sigma(t) := \sigma$ is a fixed known constant, the integral in (5.12) can be evaluated, and so (5.12) becomes

$$m(\mathbf{p}, t) := \sigma\left[1 + \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n \ t^{(1-\alpha)(n+1)}}{\Gamma\left[(1-\alpha)(n+1)+1\right]}\right] \tag{5.14}$$

The use of model $m(\mathbf{p}, t)$ as given in (5.14) requires the specification of material-specific kernel parameter values $\mathbf{p} = [\alpha, \beta, \lambda]$. The parameter α is determined from the first term of the series (5.13) expansion [122]. Thus, α is assumed to be a known constant, while β and λ need to be treated as unknown values to be determined through optimization techniques. Therefore, the set P

from which $\mathbf{p}$ takes its values is given by

$$P = \{\mathbf{p} : \alpha = \overline{\alpha},\, \beta \in \mathbb{R},\, \lambda \in \mathbb{R}\}$$

for some fixed value $\overline{\alpha}$.

In setting up the application of problem (5.2), compute

$$M(\mathbf{p}, s) := \mathcal{L}\{m(\mathbf{p}, t)\} = \frac{\sigma}{s}\left[1 + \frac{\lambda}{s^{1-\alpha} + \beta}\right]$$

for $m(\mathbf{p}, t)$ as defined in (5.14) and $R(s)$ as the Laplace transform of the regression function $r(t)$ computed to fit experimental observations $\varphi(\varepsilon_i)$.

The creep experimental data used to generate the observations $\varphi(\varepsilon_i)$ are described in detail in [115]. These experiments were performed for three types of composites with nanofillers:

1. Pure polyamide (PA),
2. Polyamide with ultra-dispersed diamonds (PA+UDD),
3. Polyamide with carbon nanotube fillers (PA+CNT).

For each material, the tests with the corresponding three loading levels $\sigma_{0.3}$, $\sigma_{0.4}$, and $\sigma_{0.5}$ are performed, where the subscript of σ indicates that the stress applied to the materials is 30%, 40%, and 50%, respectively, of the ultimate stress, which was taken equivalent to the yielding stress of each of the tested materials.

Problem (5.2) is now formulated for each of the nine data sets with the following setup.

- $\Delta(s) = |s|$.
- For each data set, regression functions having the form

  $$r(t) = \sum_i c_i\, t^{a_i} + \sum_j c_j\, e^{-a_j}$$

  are used (see [115] for more detail), where fixed values of $a_i \in (0, 1)$, $a_j \in (0, 1)$ are customized for each data set. The regression functions $r(t)$ used for each data set are given in Tab. 5.1.
- S_N is of form (5.10) where $\epsilon = 10^{-6}$, $n = 10$, $m = 20$, $c = 0.01$, and $d = 0.1$.

PA	$R(t)$
$\sigma_{0.3}$	$36.03955\,t^{\frac{1}{10}} - 30.81646\,t^{\frac{1}{15}} + 6.13278\,t^{\frac{1}{20}}$
$\sigma_{0.4}$	$2.47000\,t^{\frac{1}{5}} + 31.31428\,t^{\frac{1}{10}} - 19.69854\,t^{\frac{1}{20}}$
$\sigma_{0.5}$	$9.47225\,t^{\frac{1}{5}} + 19.43801\,t^{\frac{1}{10}} - 11.84635\,t^{\frac{1}{20}}$
PA+UDD	
$\sigma_{0.3}$	$2.91939\,t^{\frac{1}{5}} + 16.79958\,t^{\frac{1}{10}} - 9.30827\,t^{\frac{1}{20}}$
$\sigma_{0.4}$	$26.28460\,t^{\frac{1}{8}} - 13.06354\,t^{\frac{1}{16}}$
$\sigma_{0.5}$	$34.46006\,t^{\frac{1}{8}} - 18.58995\,t^{\frac{1}{16}}$
PA+CNT	
$\sigma_{0.3}$	$5.68600\,t^{\frac{1}{5}} + 3.11726\,t^{\frac{1}{10}}$
$\sigma_{0.4}$	$113.44063\,t^{\frac{1}{15}} - 102.36318\,t^{\frac{1}{20}}$
$\sigma_{0.5}$	$7.13975 - 7.13368\,e^{-0.05\,t} + 141.34010\,t^{\frac{1}{16}} - 129.24869\,t^{\frac{1}{20}}$

Table 5.1: Regression functions obtained for creep experiments

- $\varphi(\varepsilon)$ is of the form $\varphi(\varepsilon) := E\varepsilon$ where E is stated for each material in Tab. 5.2.
- $\alpha = 0.83$ for each material is estimated by the method described in [25].
- $\sigma_{0.3}$, $\sigma_{0.4}$, and $\sigma_{0.5}$ are also stated in Tab. 5.2.

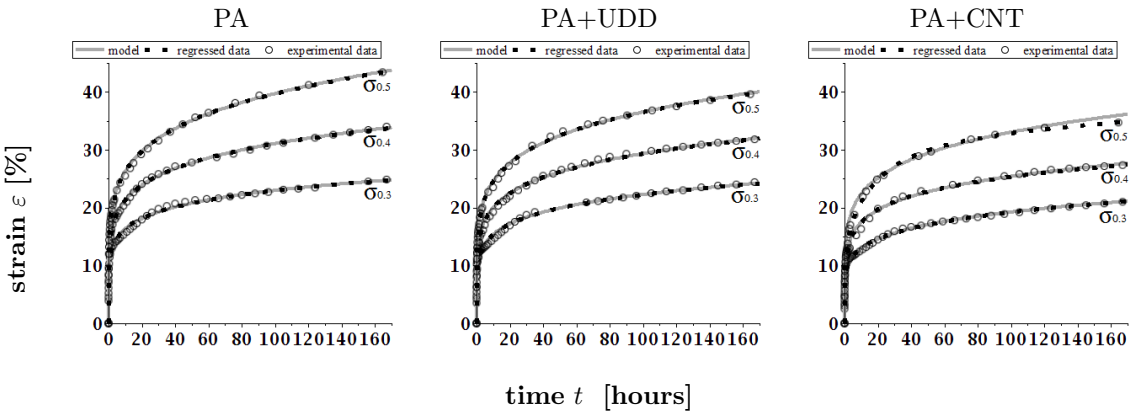

Figure 5.3: Wellness of fit plots

The results of computing optimal parameter estimates using problem (5.2) are given in Tab. 5.3. Trends in the optimal parameter estimates may be observed, for example, along the following patterns.

1. For the materials PA and PA+UDD, the increase of loading level σ results in decrease for both β and λ. (This is not the case for PA+CNT).

	α	$\sigma_{0.3}$	$\sigma_{0.4}$	$\sigma_{0.5}$	E
PA	0.83	16.20	21.60	27.00	955
PA+UDD	0.83	15.90	21.20	26.50	1008
PA+CNT	0.83	18.72	24.96	31.20	1320

Table 5.2: Setup parameters

	PA		PA+UDD		PA+CNT	
load	β	λ	β	λ	β	λ
$\sigma_{0.3}$	0.077	673.526	0.027	633.473	-0.008	563.191
$\sigma_{0.4}$	0.018	605.099	0.012	606.490	0.028	604.076
$\sigma_{0.5}$	-0.028	556.018	-0.006	579.757	0.002	593.923

Table 5.3: Optimal parameter estimates

2. One may observe an increasing trend in both parameters β and λ for the progression PA, PA+UDD, PA+CNT for fixed loading level $\sigma_{0.5}$.

Observations such as these may provide insight into the microstructure analysis.

Figure 5.3 gives plots depicting wellness of fit for the model $m(\mathbf{p}, t)$ against the experimental data $\mathbf{y}$ and the regression function $r(t)$ with the use of the parameters given in Tab. 5.3.

5.4 Conclusion

This work contributes a mathematical foundation for the comparison between the time domain least squares parameter estimation problem given in (5.1) and the Laplace domain least squares parameter estimation problem (5.2) introduced in [115, 122]. In developing this comparison, problems (5.1) and (5.2) are viewed as approximations of problems (5.3) and (5.6), respectively. The latter two problems allow for a comparison between the minimization of distance for two normed function spaces defined on the same underlying set $C[a, b]$ but with different norms. Using the language of normed spaces, equivalence between the minimization of (5.3) and the minimization of (5.6) are identified with the equivalence of the corresponding norms.

In exploring this equivalence, the existence of the necessary upper bound coefficient u, $0 < u < \infty$ was shown to exist in Proposition 15, but the non-existence of the corresponding lower bound coefficient ℓ, $0 < \ell < u$ was demonstrated through the subsequent counterexample. Thus, minimization problems (5.3) and (5.6) are not equivalent.

Although minimization problems (5.3) and (5.6) are not equivalent in terms of the above-stated norm equivalence, a connection between the two minimization problems may be obtained from the existence of upper bound coefficient u, $0 < u < \infty$, and from the observation that the objective functions of problems (5.3) and (5.6) are both continuous functions of β and λ. Due to this continuity, the existence of an upper bound coefficient u implies that for any parameters (β^*, λ^*) optimal for the t-domain problem (5.3), there correspond parameters $(\hat{\beta}, \hat{\lambda})$ that are (locally) optimal for the s-domain problem (5.6) for which $(\hat{\beta}, \hat{\lambda}) \in N((\beta^*, \lambda^*); \epsilon)$, $\epsilon > 0$; that is, $(\hat{\beta}, \hat{\lambda})$ is bounded within some ϵ-ball neighborhood of (β^*, λ^*), where ϵ depends on u. Due to the lack of a lower bound coefficient $\ell > 0$, the reverse claim does not hold; parameters $(\hat{\beta}, \hat{\lambda})$ (locally) optimal for the s-domain problem do not have any analogous bounding influence on parameters (β^*, λ^*) optimal for the t-domain problem. Thus, due to the *non*-existence of lower bounding coefficient $\ell > 0$, each computed parameter $(\hat{\beta}, \hat{\lambda})$ that is (locally) optimal for the s-domain problem (5.6) *may or may not* serve as an approximation for some (β^*, λ^*) optimal for the t-domain problem (5.3). However, due to the existence of u, $0 < u < \infty$, there is *at least one* such locally optimal parameter $(\hat{\beta}, \hat{\lambda})$serving as an approximation for (β^*, λ^*).

The application in Section 5.3 demonstrates an effective application of the s-domain problem (5.6) for computing high-quality approximations to optimal parameter estimates for the t-domain problem (5.3). An effective determination of $\Delta(s)$ and S_N is motivated from the theoretical developments in Section 5.2. This, together with the fact that problem (5.2) was formulated without using the linear least squares approximation used in [115], led to the computation of better fitting parameter estimates as compared with those obtained in [115], especially for the PA+CNT $\sigma_{0.5}$ model.

5.5 Acknowledgments

The authors are thankful for the helpful insights, comments and suggestions provided by Taufiquar Khan and Daniel Warner of the Clemson University Department of Mathematical Sciences. This work was done in support of the project no. 8640 awarded by the Russian Ministry of Education and Science.

Chapter 6

Initial exploration of estimating model parameters in the t domain for modeling the properties of viscoelastic solids

[This chapter contains the contents of a paper titled "Nonlinear modeling and optimization of parameters for viscoelastic composites and nanocomposites"; the authors are B. Dandurand, I. Viktorova, S. Alekseeva, and S. Goodson; the paper is published in *Problems of Machine Building and Automatization*, No. 3, pp. 51-57, 2011.]

The hereditary mechanics accounting for the time dependent stress-strain relationship (also known as delay or memory effect) had started from Boltzman's work in the middle of 19th century and later was developed in fundamental research on integral equations by Volterra [116].The application of this mathematical theory to the modeling of deformation processes in the viscoelastic solids that are characterized by the memory of the history of loading had shown the tremendous potential for various engineering applications [96] involving ranging loading conditions like short/longterm creep, quasistatic loading, cyclic deformation for wide range of polymer based composites and as

the recent studies show for the polymer based nanocomposites [111]. Consider the relationships between the following properties of viscoelastic solids:

- ε-denotes strain [%].
- σ-denotes load stress [MPa].
- t-denotes elapsed time [hours].

In particular, these solids have memory in the sense that load stress applied in the past manifests as present load stress.

The materials in question respond to stress in such a manner that stress applied in the past can affect strain in the present time t. The introduction of time dependence or memory effect leads to the analysis of Volterra's equation of second type [15] to model the relationship between stress as a functional of strain

$$\varphi(\varepsilon(t)) = \sigma(t) + \int_0^t K(t-\tau)\sigma(\tau)d\tau \tag{6.1}$$

where $\varphi(\varepsilon(t))$ is a response functional of ε (the so-called instantaneous loading diagram). In practice, this models the relations between time, stress and strain successfully for a wide range of materials such as polymers, metals, and composites [122]. This relationship appears visually in the graphs in Figure 6.1 for fixed time values t.

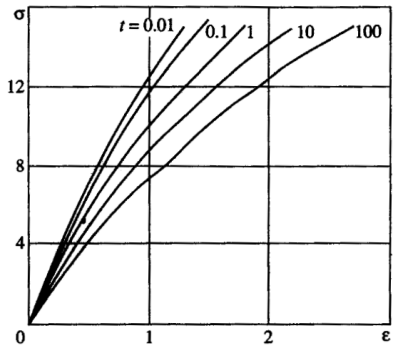

Figure 6.1: Isochronic creep diagrams.

The most suitable kernel $K(t)$ is based on the exponential of arbitrary order function and for our purposes takes form

$$K(t) = \lambda \sum_{n=0}^{\infty} \frac{\beta^n t^{n(1-\alpha)}}{\Gamma[(1-\alpha)(n+1)]} \tag{6.2}$$

The exponential of arbitrary order operators combine several important features

- The initial moment singularity at $t = 0$ is integratable.
- The asymptotic exponential behavior with $t \to \infty$.
- The resolvent operator is the same type of exponential of arbitrary order with different set of defining parameters.

Initially, the goal of our work is to find the best way to estimate the kernel parameters $p = \{\lambda, \alpha, \beta\}$ that most accurately models the relation between stress, strain, and time with the use of (6.1).

With the above kernel, the integral in equation (6.1) can be evaluated, and so (6.1) becomes

$$\varphi(\varepsilon(t)) = \sigma(t)\left[1 + \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n t^{(1-\alpha)(n+1)}}{\Gamma[(1-\alpha)(n+1)+1]}\right] \tag{6.3}$$

The parameter α is called the singularity parameter. Singularity reflects the rate of change for time $t \to 0$ of the stress-strain diagrams. As pointed out in [122], the parameter α can be determined readily from the first term of the above infinite series (6.3) given as

$$\varphi(\varepsilon(t)) = \sigma(t)\left[1 + \frac{\lambda t^{(1-\alpha)}}{\Gamma[(1-\alpha)+1]}\right] \tag{6.4}$$

and from information given in the isochronic creep curves of Figure 6.1. For different materials, α is different. The parameter α is estimated in this work in the context of optimization models.

Now consider finding the other two parameters λ and β for equation (6.3) that will model the relationship between stress, strain, and time as accurately as possible. Start by restricting to the low loading level of $\sigma = 5MPa$. (This corresponds to curve #1 in Figure 6.2.) For this low loading level σ, and for strain $\varepsilon < 1\%$, the working assumption is that material response is linear, and therefore, one can obtain $\varphi(\varepsilon) = E\varepsilon$ and $\sigma_0 = E\varepsilon_0$. Thus, substituting $\varphi(\varepsilon) = E\varepsilon$ and $\sigma = E\varepsilon_0$, we obtain

$$\varepsilon(t) = \varepsilon_0\left[1 + \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n t^{(1-\alpha)(n+1)}}{\Gamma[(1-\alpha)(n+1)+1]}\right] \tag{6.5}$$

If α is determined beforehand, then three parameters ε_0,λ, and β need to be determined. Formulate the optimal parameter estimate objective function as the following sum of squares to be minimized.

$$F(p) = \sum_{i=1}^{n}\left[\frac{\varepsilon[t_i, p] - \varepsilon[t_i]}{\varepsilon[t_i]}\right]^2 \tag{6.6}$$

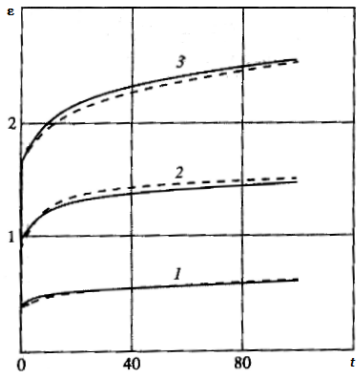

Figure 6.2: Progression of strain for three fixed levels of σ.

where $\varepsilon[t_i]$ are known data based on times t_i around which optimal parameters $p = \{\varepsilon_0, \lambda, \beta\}$ are to be obtained.

Two approaches are considered in obtaining these optimal parameter estimates.

1. **Direct Optimization:** Optimization occurs directly on the choice of parameters p. For example, in [122], the material Nylon 6 is considered, and parameter estimates of $\alpha = 0.85$, $\varepsilon_0 = 0.2$, $\lambda = 1.47$, and $\beta = 0.13$ are obtained. The wellness of fit for the model with obtained parameters is shown in Figure 6.2 [122], where solid lines indicate the experimental results, and dashed lines the predicted results from the model. Although the fit for Diagram #1 is good, there is room for improvement in getting parameters that will allow a better fit for Diagrams #2 and #3 and to predict the mechanical behavior of different types of loading regimes in general.

2. **Reparameterization:** For the same assumptions [122] of loading level about $\sigma = 5MPa$ and strain $\varepsilon < 1\%$, ε as a function of t can be well-approximated by $\varepsilon(t) = at^b$ for some values a and b obtained through a power regression. Thus, substitute for $\varepsilon = at^b$ on the left-hand side of (6.5) to obtain

$$at^b = \varepsilon_0 \left[1 + \lambda \sum_{n=0}^{\infty} \frac{(-\beta)^n t^{(1-\alpha)(n+1)}}{\Gamma[(1-\alpha)(n+1)+1]} \right]$$

Apply the Laplace-Carson transformation to both sides of the above equation to obtain

$$a \frac{\Gamma(1+b)}{s^b} = \varepsilon_0 \left[1 + \frac{\lambda}{s^{1-\alpha} + \beta} \right] \tag{6.7}$$

The question then becomes one of how parameters ε_0, λ, and β may be determined using the above integral transform (α is fixed).

One approach involves formulating a minimization model where the parameters ε_0, λ, and β that most closely enforces equality (6.7) for an appropriately chosen sample of values $s > 0$. This idea is appealing because by going from the t domain to the s domain, the need to evaluate an infinite sum is removed.

Ideally, enforcing the equation (6.7) over an interval of s values should produce unique parameter estimates. In particular, let s_1 denote the lower bound on the interval, s_3 the upper bound, and s_2 some generic s value within the interval, so that $0 < s_1 < s_2 < s_3$. It can be shown that any triple (s_1, s_2, s_3) satisfying $0 < s_1 < s_2 < s_3$ can produce unique parameter values ε_0, λ, and β by substituting each of s_1, s_2, and s_3 into (6.7) for s to get three equations that are uniquely solvable for the parameter values. Ideally, these parameter values should be the same or nearly the same regardless of the manifestation of s_2, $s_1 < s_2 < s_3$. (In other words, one interval should determine unique parameter estimates). But experiments readily verify that the parameters vary as s_2 varies. Thus, an interval must be defined by three s values rather than two.

With this in mind, the formulation of the optimization model (6.6) may be altered so that optimization occurs in terms of the s interval determined by s_1, s_2 and s_3 by reparameterizing the parameters in terms of s_1, s_2, and s_3. From a practical point of view, this approach is not desirable to get optimal parameter estimates because function evaluations are much more expensive and numerical instability arises for s_1 near zero or when the three s values otherwise get close to one another. From a theoretical point of view, the question of what kinds of intervals correspond with optimal parameter estimates is of interest, but the remainder of this paper focuses on obtaining the optimal parameter estimates by optimizing directly on the parameters ε_0, λ, β, and α.

6.1 Addressing convergence issues arising from the evaluation of a truncated infinite sum

In the expression for $\varepsilon(t)$ given in (6.5), the need to evaluate an infinite sum arises. In the absence of a closed form expression for this infinite sum, there is the need to determine a suitably chosen truncated version of this infinite sum. Where truncation can occur depends largely on the values that t, α, and β take.

The following approach is used to assess convergence. First, rewrite the infinite sum from (6.5) (with $t^{1-\alpha}$ factored out). For notational convenience, denote $\Gamma_n = \Gamma[(1-\alpha)(n+1)+1]$. Then the rewrite is given as

$$\begin{aligned}\sum_{n=0}^{\infty} \frac{(-\beta t^{1-\alpha})^n}{\Gamma_n} &= \sum_{n=0 \ by \ 2}^{\infty} \frac{(\beta t^{1-\alpha})^n}{\Gamma_n}\left(1-\beta t^{1-\alpha}\frac{\Gamma_n}{\Gamma_{n+1}}\right) \\ &= \sum_{n=0}^{\infty} \frac{(\beta t^{1-\alpha})^{2n}}{\Gamma_{2n}}\left(1-\beta t^{1-\alpha}\frac{\Gamma_{2n}}{\Gamma_{2n+1}}\right) \end{aligned} \tag{6.8}$$

The trailing term of (6.8) should be sufficiently close to zero to insure that the truncated sum is a sufficiently accurate approximation of the infinite sum. Thus, in coding the optimization model, a line of code checking whether the trailing term

$$\frac{(\beta t^{1-\alpha})^{2N}}{\Gamma_{2N}}\left(1-\beta t^{1-\alpha}\frac{\Gamma_{2N}}{\Gamma_{2N+1}}\right)$$

is close enough to zero within a specified tolerance is important to insure meaningful optimization results. The number of summation terms N need not be large ($N = 100$ is sufficient) for the trailing term to be very small if $|-\beta t^{1-\alpha}| < 1$.

If $|-\beta t^{1-\alpha}| \geq 1$, then consider the cases

$$\begin{aligned} -\beta t^{1-\alpha} &\leq -1 \\ -\beta t^{1-\alpha} &\geq 1 \end{aligned} \tag{6.9}$$

In the first case where $-\beta t^{1-\alpha} \leq -1$, the size of the trailing term as discussed above is the only thing to consider. In this case, if the trailing term is not small enough, summing more terms can give a sufficiently small trailing term. If, on the other hand, $-\beta t^{1-\alpha} \geq 1$, then the size of the trailing

term is not the only problem to consider.

If $-\beta t^{1-\alpha} \geq 1$ is too large, then the summation (6.8) involves a sum of large negative terms followed by a sum of large positive terms that get smaller as the summation progresses. If the summation (6.8) is supposed to take a value close to zero, then in finite precision calculation, a loss of precision in the resulting calculation known as catastrophic cancellation occurs. For this model, any use of sufficiently large β values will likely result in this and so appropriate bounds on β must be enforced to insure $-\beta t^{1-\alpha} < 1$. In this work, lower and upper bounds on the β values are determined separately. Typically, the lower bound will depend only on the available number of terms N that are computed in the truncated sum. The upper bound depends on this and also on the values of α and the largest value of t in the data. Namely, the upper bound on β must satisfy $-\beta t^{1-\alpha} < 1$ for the set α value and the largest t value in the data set.

From the above considerations, it follows that there is room to relax the lower bound on β by using as many terms in the infinite summation as possible. In a C++ programming context, double precision arithmetic limits how far the summation can be evaluated. There is more than one solution to this problem. The solution used here is to have ratios $\frac{\Gamma_n}{\Gamma_{n+1}}$ available to compute terms in the summations recursively by multiplication. For example, given the term $\frac{-\beta t^{1-\alpha}}{\Gamma_n}$, the next term is obtained by multiplication by $(-\beta t^{1-\alpha})\frac{\Gamma_n}{\Gamma_{n+1}}$.

In doing this, the question arises, will the successive multiplications cause an accumulation of roundoff error resulting from the nature of finite precision arithmetic. It will, but if N is the length of the truncated sum, the expected accumulated error is $\sqrt{N} * \epsilon$ where ϵ is machine epsilon. Thus, for say, $N = 10,000$ summation terms, an expected loss of 2 or 3 decimal points of precision occurs. This is satisfactory for double precision arithmetic. For this work, $N = 1,000$ terms are used (finite precision only allows about $N = 150$ when $\alpha = 0.85$ if the above approach is not used). Using MAPLE 14, this problem does not arise, as the maximum magnitudes and precisions allowable with MAPLE arithmetic computation are large enough for the optimization needs in this work.

6.2 Results

In finding optimal parameter estimates in the formulation (6.6), there is readily available gradient and Hessian information. This makes the optimization model given by (6.6) solvable using standard deterministic nonlinear optimization techniques presented in many textbooks, such

as [83, 7]. This is a departure from previous works that normally use metaheuristic approaches such as simulated annealing.

Optimization occurs on the functional (6.6) with respect to ε_0, λ, β and α. However, the cost of gradient and Hessian evaluations when α is included as an optimization variable is prohibitive, so the model only considers ε_0, λ, and β as decision variables, while α is treated as a constant in the model that is manually varied before the running of the solver.

For each material and loading level, different α values are tried, e.g. $\alpha = 0.5, \ldots, 0.8$ and the results for the best α value are given.

6.2.1 Results for C-type polymer nanocomposite

Given below are the data sets, the parameter estimates obtained from the data using the MAPLE 14 command LSSolve, and the plots for each data set that shows how well the model fits the data.

Data Set 1 for $\sigma_{0.3}$

time (hours)	0.000	0.017	0.083	0.167	0.433	1.100	2.100	4.100
strain (%)	0.05	0.06	0.09	0.1	0.12	0.15	0.18	0.23
time (hours)	5.600	23.100	31.100	54.600	75.100	100.100	127.600	168.850
strain (%)	0.25	0.36	0.39	0.45	0.5	0.53	0.58	0.61

(6.10)

Data Set 2 for $\sigma_{0.3}$

time (hours)	0.000	0.017	0.083	0.167	0.400	1.067	2.067	4.067
strain (%)	0.08	0.09	0.12	0.13	0.15	0.18	0.2	0.25
time (hours)	5.567	23.067	31.067	54.567	75.067	100.067	127.567	168.817
strain (%)	0.27	0.34	0.37	0.42	0.46	0.51	0.55	0.6

(6.11)

Data Set 1 for $\sigma_{0.4}$

time (hours)	0.000	0.017	0.250	0.383	0.883	1.383	2.383	3.383	22.383
strain (%)	0.29	0.32	0.4	0.54	0.62	0.66	0.72	0.76	0.99
time (hours)	28.383	34.383	58.383	77.383	106.883	124.383	153.883	197.383	
strain (%)	1.04	1.17	1.24	1.29	1.35	1.48	1.53	1.6	

(6.12)

Data Set 2 for $\sigma_{0.4}$

time (hours)	0.000	0.017	0.250	0.333	0.833	1.333	2.333	3.333	22.333
strain (%)	0.36	0.42	0.54	0.67	0.77	0.82	0.87	0.92	1.18
time (hours)	28.333	34.333	58.333	77.333	106.833	124.333	153.833	197.333	
strain (%)	1.23	1.36	1.44	1.5	1.56	1.69	1.73	1.81	

(6.13)

Data Set 1 for $\sigma_{0.5}$

time (hours)	0.000	0.017	0.083	0.167	0.250	0.333	0.500	0.667	0.833
strain (%)	0	0.23	0.36	0.41	0.45	0.49	0.53	0.57	0.59
time (hours)	1.000	1.250	1.500	2.000	2.500	3.000	3.500	19.483	23.483
strain (%)	0.61	0.63	0.65	0.69	0.72	0.74	0.76	1.17	1.22
time (hours)	44.483	51.983	75.983	90.483	120.150	164.483	170.983	194.983	214.983
strain (%)	1.39	1.43	1.54	1.59	1.65	1.71	1.73	1.76	1.79
time (hours)	242.483	308.483	312.983	327.983	358.983	376.983	404.983	424.983	452.650
strain (%)	1.83	1.9	1.91	1.92	1.95	1.96	1.97	2.01	2.05

(6.14)

Data Set 2 for $\sigma_{0.5}$

time (hours)	0.000	0.017	0.083	0.167	0.250	0.333	0.500	0.667	0.833
strain (%)	0	0.27	0.37	0.43	0.46	0.49	0.53	0.56	0.58
time (hours)	1.000	1.250	1.500	2.000	2.500	3.000	3.500	19.550	23.550
strain (%)	0.6	0.63	0.64	0.68	0.7	0.72	0.74	1.07	1.11
time (hours)	44.550	52.050	76.050	90.550	120.217	164.550	171.050	195.050	215.050
strain (%)	1.27	1.3	1.4	1.44	1.5	1.55	1.56	1.6	1.63
time (hours)	242.550	308.550	313.050	328.050	359.050	377.050	405.050	425.050	452.717
strain (%)	1.67	1.75	1.77	1.78	1.8	1.82	1.83	1.88	1.91

(6.15)

Surface plots of equation (6.6) for fixed ε_0 and α values

In Figure 6.3, 3-dimensional plots are given around the parameter estimates that are optimal. The parameters ε_0 and α are fixed at the obtained optimal value and λ and β are allowed to vary. The following observations are warranted from the plots.

1. For each plot, the objective function surface, although not necessarily convex, seems to show one local minimum located inside of the inner most elliptical contour depicted in the plot.

Thus, the optimal solution obtained by the solver should be a globally optimal solution.

2. The elongation of the elliptical contours containing the optimal parameter estimates may lead to the optimal parameter estimates being sensitive to small changes in the experimental data.

Parameter Estimates

	Data Set 1			Data Set 2		
	$\sigma_{0.3}$	$\sigma_{0.4}$	$\sigma_{0.5}$	$\sigma_{0.3}$	$\sigma_{0.4}$	$\sigma_{0.5}$
α	0.59	0.66	0.70	0.67	0.69	0.76
ε_0	4.697×10^{-2}	2.688×10^{-1}	7.548×10^{-2}	7.476×10^{-2}	3.406×10^{-1}	6.977×10^{-2}
λ	2.077×10^{0}	1.159×10^{0}	7.217×10^{0}	1.226×10^{0}	1.164×10^{0}	7.457×10^{0}
β	7.431×10^{-2}	1.015×10^{-1}	1.453×10^{-1}	1.728×10^{-2}	1.108×10^{-1}	8.300×10^{-2}
obj val	2.664×10^{-3}	4.328×10^{-2}	1.354×10^{-2}	6.289×10^{-3}	3.410×10^{-2}	7.330×10^{-3}

(6.16)

The objective values in the above table gives some measure of how well the model (6.5) with the obtained parameter values fits the given data. The wellness of fit is also illustrated by the plots given in Figures 6.4, 6.5, and 6.6.

6.3 Conclusion

Earlier, a classic formulation for the modeling of creep is given in terms of loading condition σ and time t. From this formulation arises the need to obtain optimal parameter estimates. This work presents a modeling framework using deterministic gradient-based optimization to obtain these parameters. At the solver stage, three variables, ε_0, λ and β are chosen along with α before the solver stage, to obtain a function of form (6.5) that best fits the data.

These parameter estimates allow for the potential usability of functions of the form (6.5). With this potential comes the following potential limitation. The usefulness of the model will be limited by the value of β. A given value of $\beta > 0$ imposes an upper bound on the usable values of t for a given availability of computational precision. Values of t that are too large given β and a given availability of precision will result in catastrophic cancellation occurring in the summation of (6.5) and the severe loss of precision in output resulting from this. This issue calls attention to the need to use high enough precision arithmetic to insure that the loss of precision resulting from evaluating

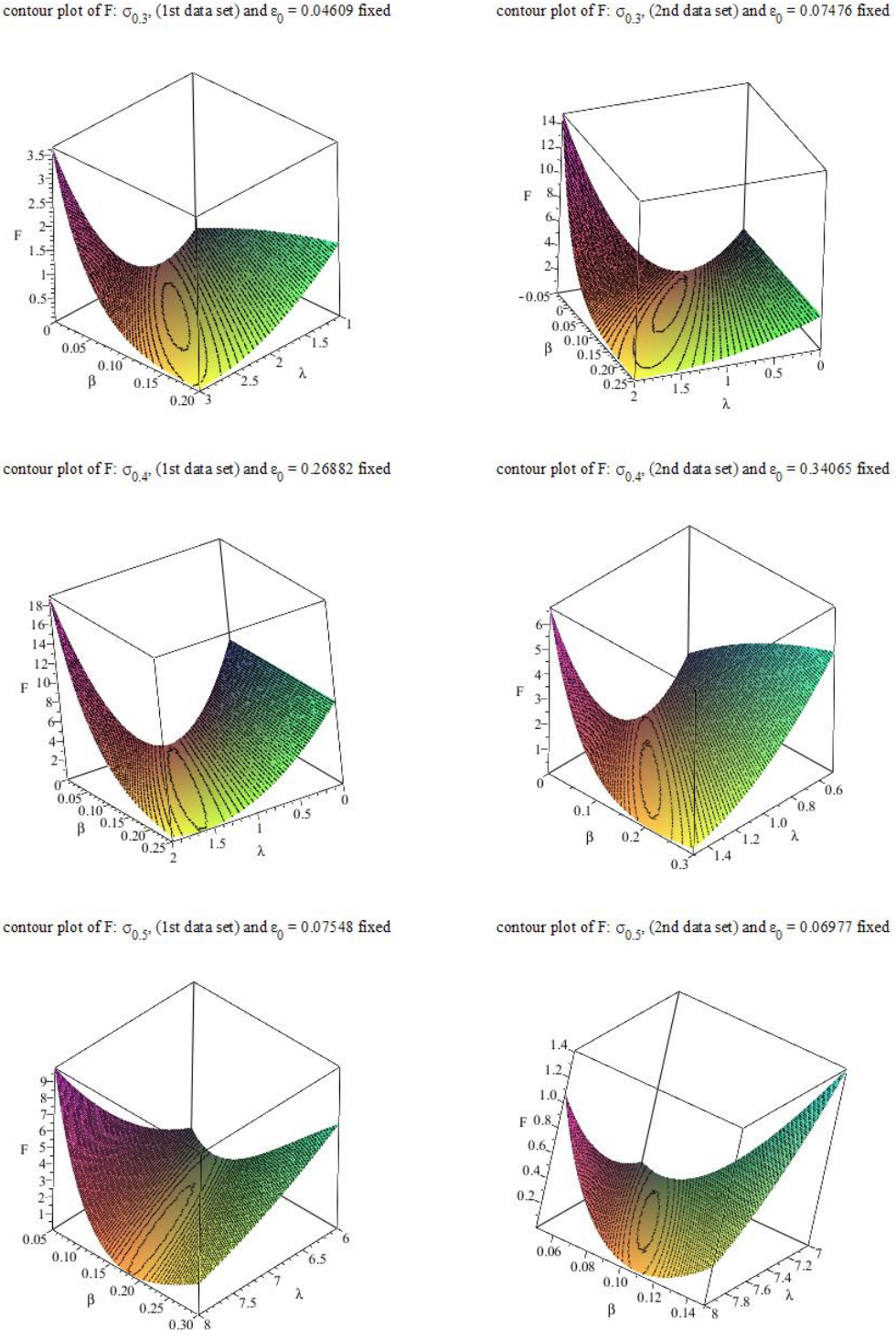

Figure 6.3: Contour plots illustrate surface over which optimization occurs for fixed ε_0 and α.

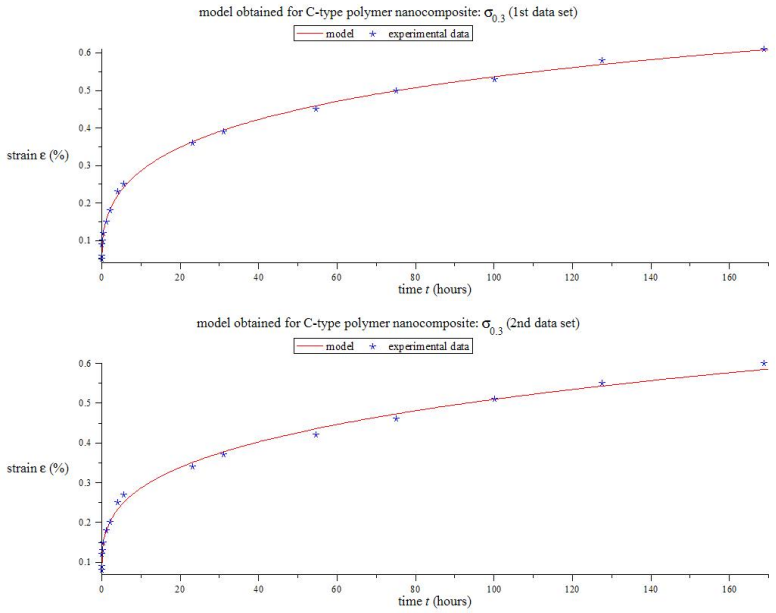

Figure 6.4: Plots illustrating wellness of fit for obtained parameter estimates ($\sigma_{0.3}$)

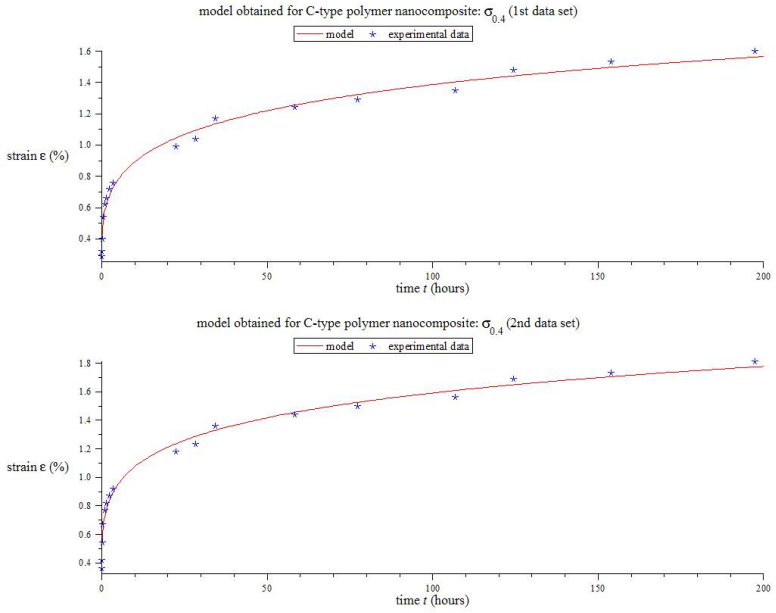

Figure 6.5: Plots illustrating wellness of fit for obtained parameter estimates ($\sigma_{0.4}$)

the infinite sum is within toleration.

For the purposes of parameter estimation, the use of appropriate transforms on the function (6.5) set equal to a suitably good regression of the experimental data the and the use of appropriate s intervals is an area of future work. In the formulation of the parameter estimation problem, an

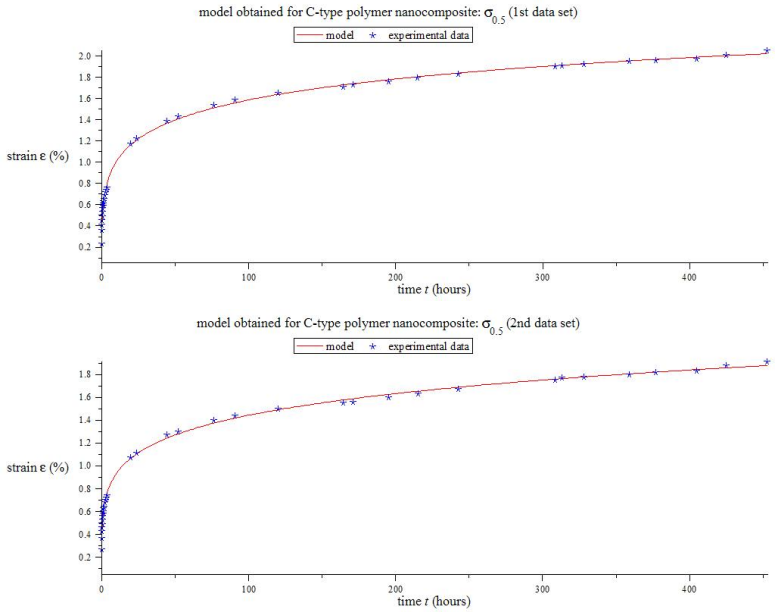

Figure 6.6: Plots illustrating wellness of fit for obtained parameter estimates ($\sigma_{0.5}$)

approach using Laplace-Carson transformation is attempted as outlined above. If future research in this approach can yield improvement in either the type of transform used or in knowledge of what types of intervals to use, then a more elegant approach to obtaining optimal parameter estimates will be available that does not rely on the evaluation of truncated infinite sums.

Appendices

Appendix A Maple Code for Chapter 2

The contents of Appendix A include the Maple code for implementing the algorithms described in Chapter 2. These include BCD, BCDMM (referred to as Coor in Chapter 2), ADMM, two weight vector generating methods, and the main procedures, DSDA and OSDA. The implementation is specific to a decomposable example MOP with four objective functions and a constraint $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ of global scope.

A.1 Implementation of two weight vector generating schemes

Both DSDA and OSDA require the use of a collection of weight vectors. The methods `genURandWeights()` and `genSierpinskiWeights()` provide two such means of generating the required sample of weights based on the ideas of Section 2.2.3. The input `p` is the length of the weight vector to be generated, and in the case of `genURandWeights()`, `N` is a positive integer parameter whose increase corresponds to increased refinement of a finite discrete representation of the sample space.

```
genURandWeights := proc(N, p)
 local R, N2, num, Weights, tWeights, tSum, l, i, j, c, M, m :
 M := combinat[binomial](N + p - 1, p - 1) :
 for m from 0 by 1 to M - 1 do
 Weights[m+1] := Vector(p) : num := (m mod M) : R := num : N2 := N :
 for i from 2 by 1 to p do
    for j from 0 by 1 to N2 do
       c := combinat[binomial](N2 - j + p - i, p - i) :
       if R >= c then
          R := R - c :next:
       else
          Weights[m+1][ i - 1 ] := j : N2 := N2 - j :break:
       end if:
    end do:
 end do:
 Weights[m+1][p] := N2 :for l from 1 by 1 to p do Weights[m+1]
   [l] := (Weights[m+1][l] + 0.100)/(N + 0.20) :end do:
 end do: print(M) :
   return [seq(Weights[k], k = 1 ..M)] :
 end proc:
```

```
genSierpinskiWeights := proc(p)
 local mPower, B, i, counter, Weights, P, j, M, k, l :
  mPower := 2 : B := Matrix(1, p) :for i from 1 by 1 to p do B[1, i] := 1/p : end do: B;
  counter := 1 : Weights_counter := B.IdentityMatrix(p) : counter := counter + 1 :
for i from 1 by 1 to p do P_i := 1/2 ·IdentityMatrix(p) :
for j from 1 by 1 to p do P_i[j, i] := 1/2 :end do:
 P_i[i, i] := 1 :
end do:
for i from 1 by 1 to p do
  M_1 := IdentityMatrix(p) :  M_1 := P_i.M_1 :
   Weights_counter := B.MatrixPower(M_1, mPower) : counter := counter + 1 :
 for j from 1 by 1 to p do
 M_2 := P_j.M_1 :
   Weights_counter := B.MatrixPower(M_2, mPower) : counter := counter + 1 :
 for k from 1 by 1 to p do
 M_3 := P_k.M_2 :
   Weights_counter := B.MatrixPower(M_3, mPower) : counter := counter + 1 :
 for l from 1 by 1 to p do
 M_4 := P_l.M_3 :
   Weights_counter := B.MatrixPower(M_4, mPower) : counter := counter + 1 :
 end do: end do: end do: end do:
  return [seq(Weights_k[1], k = 1 ..(counter − 1))] :
 end proc:
```

A.2 Implementation of DSDA

The first main method, `DSDA()`, and its helper methods `BCD()` and `BCDMM()` are given. The method `DSDA()` computes efficient points for an example MOP with four objective functions given by input `f1`, `f2`, `f3`, `f4` and a constraint function `h`. The method `BCD()` corresponds to Alg. 1 of Chapter 2. It takes as input a scalarized objective function `f`, starting values for the block coordinates `x1start`, `x2start`, `sStart`, and a small positive convergence tolerance value `tol`. The method `BCDMM()` corresponds to Alg. 2 of Chapter 2. In addition to the inputs taken by `BCD`, the method `BCDMM()` takes as inputs the coordinating constraint function `h`, the constraint function needed due to the use of the quadratic scalarization method `qsH`, and starting values for the augmented Lagrange parameters `v0` and `mu0`.

```
DSDA := proc(f1, f2, f3, f4, h)
 local S, UFac, U, OnesVec, Q, z, W, numWts, wIter, fVec, fVecT, F, qsH, XV :
 S := Matrix([[w1, -w2, -w3, -w4], [w2, w1, 0, 0], [w3, 0, w1, 0], [w4, 0, 0, w1]]) : UFac
     := QRDecomposition(eval(S, [w1 = 1, w2 = 1, w3 = 1, w4 = 1])) : U := (UFac[1])[1..4, 2..4] :
 OnesVec := Matrix([[1], [1], [1], [1]]) : Q := U.Transpose(U) : z := [0, 0, 0, 0] :
 W := genURandWeights(3, 4) : numWts := 20 :
 for wIter from 1 by 1 to numWts do
  fVec := Matrix([[W_wIter[1]·(f1 − z[1]) + s1], [W_wIter[2]·(f2 − z[2]) + s2], [W_wIter[3]·(f3
      − z[3]) + s3], [W_wIter[4]·(f4 − z[4]) + s4]]) :
  fVecT := Transpose(fVec) :
  F := ((fVecT.OnesVec)[1, 1]) :
  qsH := Transpose(U).fVec :
  XV_wIter := BCDMM(F, h, qsH, [0.25, 1], [0.25, 1], [0, 0, 0, 0], Matrix([0, 0]), 3, 10^−3) :
 end do:
 return [seq(XV_k, k = 1..numWts)] :
end proc:

BCD := proc(f, x1start, x2start, sStart, tol)
 local iterationCounter, sUB, X1Val, X2Val, SVal, X1Update, OldX1Val, X2Update, OldX2Val, SUpdate,
     OldSVal :
 iterationCounter := 1 : sUB := ∞:
 X1Val := x1start : X2Val := x2start : SVal := sStart :
 X1Update := NLPSolve(eval(f, [x2,1 = X2Val[1], x2,2 = X2Val[2], s1 = SVal[1], s2 = SVal[2], s3
     = SVal[3], s4 = SVal[4]]), x1,1 = 0..2, x1,2 = 0..2) :
 OldX1Val := X1Val : X1Val := [rhs(X1Update[2, 1]), rhs(X1Update[2, 2])] :
 X2Update := NLPSolve(eval(f, [x1,1 = X1Val[1], x1,2 = X1Val[2], s1 = SVal[1], s2 = SVal[2], s3
     = SVal[3], s4 = SVal[4]]), x2,1 = 0..2, x2,2 = 0..2) :
 OldX2Val := X2Val : X2Val := [rhs(X2Update[2, 1]), rhs(X2Update[2, 2])] :
 SUpdate := NLPSolve(eval(f, [x1,1 = X1Val[1], x1,2 = X1Val[2], x2,1 = X2Val[1], x2,2
     = X2Val[2]]), s1 = 0..sUB, s2 = 0..sUB, s3 = 0..sUB, s4 = 0..sUB) :
 OldSVal := SVal : SVal := [rhs(SUpdate[2, 1]), rhs(SUpdate[2, 2]), rhs(SUpdate[2, 3]),
     rhs(SUpdate[2, 4])] :
 while norm(Vector(OldX1Val − X1Val), 2) + norm(Vector(OldX2Val − X2Val), 2)
     + norm(Vector(OldSVal − SVal), 2) > tol do
  X1Update := NLPSolve(eval(f, [x2,1 = X2Val[1], x2,2 = X2Val[2], s1 = SVal[1], s2 = SVal[2], s3
     = SVal[3], s4 = SVal[4]]), x1,1 = 0..2, x1,2 = 0..2) :
 OldX1Val := X1Val : X1Val := [rhs(X1Update[2, 1]), rhs(X1Update[2, 2])] :
 X2Update := NLPSolve(eval(f, [x1,1 = X1Val[1], x1,2 = X1Val[2], s1 = SVal[1], s2 = SVal[2], s3
     = SVal[3], s4 = SVal[4]]), x2,1 = 0..2, x2,2 = 0..2) :
 OldX2Val := X2Val : X2Val := [rhs(X2Update[2, 1]), rhs(X2Update[2, 2])] :
 SUpdate := NLPSolve(eval(f, [x1,1 = X1Val[1], x1,2 = X1Val[2], x2,1 = X2Val[1], x2,2
     = X2Val[2]]), s1 = 0..sUB, s2 = 0..sUB, s3 = 0..sUB, s4 = 0..sUB) :
 OldSVal := SVal : SVal := [rhs(SUpdate[2, 1]), rhs(SUpdate[2, 2]), rhs(SUpdate[2, 3]),
     rhs(SUpdate[2, 4])] :
 iterationCounter := iterationCounter + 1 :
 if iterationCounter ≥ 1000 then print("BCD: Maximum number of iterations exceeded.") :break: end
     if
 end do:
 return [X1Val[1], X1Val[2], X2Val[1], X2Val[2], SVal[1], SVal[2], SVal[3], SVal[4]] :
end proc:
```

```
BCDMM := proc( f, h, qsH, x1start, x2start, sStart, v0, μ0, tol)
 local iterationCounter, MuVal, aVal, X1Val, X2Val, SVal, VVal, YVal, NewXVal, HVal, qsHVal :
 iterationCounter := 1 : MuVal := μ0 : aVal := 2 :
 X1Val := x1start : X2Val := x2start : SVal := sStart : VVal := v0 : YVal := Matrix( [ [0, 0, 0] ]) :
 NewXVal := BCD( f + (VVal.h) [1, 1] + MuVal/2 · (Transpose(h).h) [1, 1] + (YVal.qsH) [1, 1]
     + aVal/2 · (Transpose(qsH).qsH) [1, 1], X1Val, X2Val, SVal, tol ) :
 X1Val := [NewXVal[1], NewXVal[2]] : X2Val := [NewXVal[3], NewXVal[4]] :
 SVal := [NewXVal[5], NewXVal[6], NewXVal[7], NewXVal[8]] :
 HVal := eval( h, [x1,1 = NewXVal[1], x1,2 = NewXVal[2], x2,1 = NewXVal[3], x2,2 = NewXVal[4]] ) :
 qsHVal := eval( qsH, [x1,1 = NewXVal[1], x1,2 = NewXVal[2], x2,1 = NewXVal[3], x2,2
     = NewXVal[4], s1 = SVal[1], s2 = SVal[2], s3 = SVal[3], s4 = SVal[4]] ) :
while norm(HVal[1 ..2, 1], 2) + norm(qsHVal[1 ..3, 1], 2) > tol do
 VVal := VVal + MuVal·Transpose(HVal) :if MuVal ≤ 27 then MuVal := 3·MuVal :end if:
 YVal := YVal + aVal·Transpose(qsHVal) :if aVal ≤ 32 then aVal := 2· aVal :end if:
  NewXVal := BCD( f + (VVal.h) [1, 1] + MuVal/2 · (Transpose(h).h) [1, 1] + (YVal.qsH) [1, 1]
     + aVal/2 · (Transpose(qsH).qsH) [1, 1], X1Val, X2Val, SVal, tol ) :
 X1Val := [NewXVal[1], NewXVal[2]] : X2Val := [NewXVal[3], NewXVal[4]] :
 SVal := [NewXVal[5], NewXVal[6], NewXVal[7], NewXVal[8]] :
 HVal := eval( h, [x1,1 = NewXVal[1], x1,2 = NewXVal[2], x2,1 = NewXVal[3], x2,2
     = NewXVal[4]] ) :
 qsHVal := eval( qsH, [x1,1 = NewXVal[1], x1,2 = NewXVal[2], x2,1 = NewXVal[3], x2,2
     = NewXVal[4], s1 = SVal[1], s2 = SVal[2], s3 = SVal[3], s4 = SVal[4]] ) :
 iterationCounter := iterationCounter + 1 :
 if iterationCounter ≥ 500 then print("BCDMM: Maximum number of iterations exceeded.") :break:
    end if
end do:
 return evalf ( [X1Val[1], X1Val[2], X2Val[1], X2Val[2], SVal[1], SVal[2], SVal[3], SVal[4],
    VVal[1, 1], VVal[1, 2], YVal[1, 1], YVal[1, 2], YVal[1, 3], aVal] ) :
end proc:
```

A.3 Implementation of OSDA

The second main method, `OSDA()` is given together with its helper methods `testLegitPoint()` and `ADMM()`. The method `OSDA()` computes efficient points for an objective space decomposable MOP with four objective functions given by the input `f11`, `f12`, `f21`, `f22` and a constraint function `h`.

```
OSDA := proc(f11, f12, f21, f22, h)
 local i, numWts, WSP1, WSP2, AVals, PVec, Q, aIndex, aWt, w1Index, ySP1, fVec1, fVec1T, f1 :
 local w2Index, ySP2, fVec2, fVec2T, f2, XV, XVals, XEff, XEffInvalid, suffCond1, suffCond2 :
 i := 1 : numWts := 11 :
 WSP1 := genURandWeights(numWts − 1, 2) :
 WSP2 := genURandWeights(numWts − 1, 2) :
 AVals := genURandWeights(numWts − 1, 2) :
 PVec := evalf(Matrix([[⌊√2/2⌋, ⌊√2/2⌋]])) :
 Q := 4· Matrix([[0.5, −0.5], [−0.5, 0.5]]) :
 for aIndex from 1 by 1 to numWts do
 aWt := [AVals_aIndex[1], AVals_aIndex[2]] :
 for w1Index from 1 by 1 to numWts do
 ySP1 := 20·[WSP1_w1Index[1], WSP1_w1Index[2]] :
 fVec1 := Matrix([[f11 − ySP1[1]], [f12 − ySP1[2]]]) : fVec1T := Transpose(fVec1) :
 f1 := (1/2 · fVec1T.Q.fVec1 + fVec1T.PVec)[1, 1] :
  for w2Index from 1 by 1 to numWts do
  ySP2 := 20·[WSP2_w2Index[1], WSP2_w2Index[2]] :
  fVec2 := Matrix([[f21 − ySP2[1]], [f22 − ySP2[2]]]) : fVec2T := Transpose(fVec2) :
  f2 := (1/2 · fVec2T.Q.fVec2 + fVec2T.PVec)[1, 1] :
  XV := ADMM(aWt[1]· f1, aWt[2]· f2, h, [0, 0], [0, 0], Matrix([0, 0]), 2, 10^−4) :
  XVals_i := XV[1..4] :
  suffCond1 := testLegitPoint(Q, PVec, eval(fVec1, [x_1,1 = XVals_i[1], x_1,2 = XVals_i[2]]), 2, 10^−6) :
  suffCond2 := testLegitPoint(Q, PVec, eval(fVec2, [x_2,1 = XVals_i[3], x_2,2 = XVals_i[4]]), 2, 10^−6) :
  if suffCond1 and suffCond2 then
   XEff_i := XVals_i : XEffInvalid_i := [−1, −1, −1, −1] :
  else
   XEff_i := [−1, −1, −1, −1] : XEffInvalid_i := XVals_i :
  end if:
  i := i + 1 :
 end do:
 end do:
end do: print(i − 1) :
 return Matrix([seq(XEff_j, j = 1..i − 1)])
end proc:
```

The method `testLegitPoint()` tests for membership of an outcome given by input `fVec` in the critical set defined by (2.11) parameterized by the input matrix `Q`, vector `PVec`, and `p` specifying the number of objective functions.

```
testLegitPoint := proc(Q, PVec, fVec, p, tol)
 local T, i : T := Q.fVec + PVec :
  for i from 1 by 1 to p do
   if T[i, 1] ≤ 0 then return false : end if
  end do: return true :
end proc:
```

The method `ADMM()` used by `OSDA()` takes as input the subproblem scalarized objective functions `f1` and `f2`, with the other inputs having the same meaning as in `OSDA()`.

```
ADMM := proc( f1, f2, h, x1start, x2start, v0, μ0, tol)
 local iterationCounter, X1Val, X2Val, VVal, mu, HSP1, X1Update, OldX1Val, HSP2, X2Update,
    OldX2Val, HVal :
 iterationCounter := 1 :
 X1Val := x1start : X2Val := x2start : VVal := v0 : μ := μ0 :
 HSP1 := eval( h, [x2,1 = X2Val[1], x2,2 = X2Val[2]] ) :
 X1Update := NLPSolve( f1 + (VVal.HSP1)[1, 1] + (μ/2)·(Transpose(HSP1).HSP1)[1, 1], x1,1 = 0
    ..2, x1,2 = 0 ..2 ) :
 OldX1Val := X1Val : X1Val := [rhs(X1Update[2, 1]), rhs(X1Update[2, 2])] :
 HSP2 := eval( h, [x1,1 = X1Val[1], x1,2 = X1Val[2]] ) :
 X2Update := NLPSolve( f2 + (VVal.HSP2)[1, 1] + (μ/2)·(Transpose(HSP2).HSP2)[1, 1], x2,1 = 0
    ..2, x2,2 = 0 ..2 ) :
 OldX2Val := X2Val : X2Val := [rhs(X2Update[2, 1]), rhs(X2Update[2, 2])] :
 HVal := eval( h, [x1,1 = X1Val[1], x1,2 = X1Val[2], x2,1 = X2Val[1], x2,2 = X2Val[2]] ) :
 VVal := VVal + μ·Transpose(HVal) : if μ ≤ 128 then μ := 2·μ :end if:
 while norm(Vector(OldX1Val − X1Val), 2) + norm(Vector(OldX2Val − X2Val), 2) + norm(HVal[1
    ..2, 1], 2) > tol do
  HSP1 := eval( h, [x2,1 = X2Val[1], x2,2 = X2Val[2]] ) :
  X1Update := NLPSolve( f1 + (VVal.HSP1)[1, 1] + (μ/2)·(Transpose(HSP1).HSP1)[1, 1], x1,1 = 0
    ..2, x1,2 = 0 ..2 ) :
  OldX1Val := X1Val : X1Val := [rhs(X1Update[2, 1]), rhs(X1Update[2, 2])] :
  HSP2 := eval( h, [x1,1 = X1Val[1], x1,2 = X1Val[2]] ) :
  X2Update := NLPSolve( f2 + (VVal.HSP2)[1, 1] + (μ/2)·(Transpose(HSP2).HSP2)[1, 1], x2,1 = 0
    ..2, x2,2 = 0 ..2 ) :
  OldX2Val := X2Val : X2Val := [rhs(X2Update[2, 1]), rhs(X2Update[2, 2])] :
  HVal := eval( h, [x1,1 = X1Val[1], x1,2 = X1Val[2], x2,1 = X2Val[1], x2,2 = X2Val[2]] ) :
  VVal := VVal + μ·Transpose(HVal) : if μ ≤ 128 then μ := 2·μ :end if:
  iterationCounter := iterationCounter + 1 :
  if iterationCounter ≥ 3000 then print("Maximum number of iterations exceeded.") :break: end if
 end do:
 return [X1Val[1], X1Val[2], X2Val[1], X2Val[2], VVal[1, 1], VVal[1, 2]] :
end proc:
```

Appendix B Code for Chapters 3 and 4

Appendix B contains the Matlab code implementing the battery-level solver of the bilevel vehicle design problem described in Chapter 3 and the coordination between the two subproblems (vehicle level and battery level) described in Chapter 4. The vehicle-level solver is implemented in Java and has been compiled into a Java executable named `pack.jar`. The Java implementation of the vehicle-level solver is beyond the scope of this dissertation and so is not included. Also included at the end of this appendix are the contents of the main batch file `run_vehicle_design.bat`.

B.1 Initialization and the main Matlab method

The method `initializeMODA()` performs prerequisite initializations in preparation for the main iterations of the bilevel solution algorithm described in Chapter 4. The input variables are as follows: `nCols` is the number of columns in the battery cell arrangement; `weightIndex` specifies which battery-level weighted-sum weight to use (takes values 1,2, or 3 based on the three weights generated by the method `generateWeightsEquitable()`); `betaExp` specifies the weighting of importance of the battery-level problem, typically takes values 1,2,3, or 4; `scalCoeff` is used to rescale the consistency constraint for improved computational speed; and `scalType` specifies the type of scalarization used at the vehicle-level and its only role in the displayed code is to determine part of the output file's name.

```
function initializeMODA(nCols,weightIndex,betaExp,scalCoeff,scalType)
  global spacing_traverse t f w numCols numRows ...
    numberOfColumns numberOfRows cell_diameter...
    v penaltyWeight scalingCoefficient L_target W_target...
    inputVals tempVals
%%Many of these initializations are necessary when Simulink is used to
%%compute battery cell temperatures
  tau=0; %
  t=[0:1:100]';
  numberOfColumns=nCols;
  numberOfRows=72/numberOfColumns;
  numCols=[t numberOfColumns*ones(size(t))];
```

```
  numRows=[t numberOfRows*ones(size(t))];
  cell_diameter=0.034; %%Simulink model requires this be given in meters
%%%scaling is necessary for computational expediency
  scalingCoefficient=scalCoeff;
%%Augmented Lagrange parameters initially set to 0
  v=[0.0,0.0];
  penaltyWeight=0.0;
%%Initial battery box dimension targets
  L_target=300.0;
  W_target=300.0;
%%Reference indices for data in output file
  OI_PCELL=1;
  OI_BATT_L=2;
  OI_BATT_W=3;
  OI_VEH_L=4;
  OI_VEH_W=5;
  OI_V1=6;
  OI_V2=7;
  OI_MU=8;
  OI_TAU=9;
  OI_BATT_OBJ=10;
  OI_VEH_OBJ=11;
  OI_VEH_OBJ1=12;
  OI_VEH_OBJ2=13;
  OI_VEH_OBJ3=14;
  OI_AL_TERM=15;
  OI_AIO=16;
  outputVector=zeros(1,16);
%%Prepare for initial battery-level computation...
  dataFileName=sprintf('./TemperatureData/tempData%d.mat',numberOfColumns);
  load(dataFileName); %%%this initializes inputVals, tempVals
```

```
  weights=generateWeightsEquitable(numberOfColumns);
  w=weights(weightIndex,:);
  w=w*(10^betaExp);
  [pcell, battObjValAL]=fminbnd(@computeZ_Approx,1.01,2.0);
  [L W]=computeBoxDims(pcell);
%%Computing augmented Lagrange term
  battBoxDiscreps=computeBBDiscreps([L;W],[L_target;W_target],scalingCoefficient);
  ALTerm=v*battBoxDiscreps+(penaltyWeight/2)*norm(battBoxDiscreps,2)^2;
%%%%initialize the first row of the output vector as follows...
  outputVector(1,OI_PCELL)=pcell;
  outputVector(1,OI_BATT_L)=L;
  outputVector(1,OI_BATT_W)=W;
  outputVector(1,OI_VEH_L)=L;
  outputVector(1,OI_VEH_W)=W;
  outputVector(1,OI_V1)=v(1);
  outputVector(1,OI_V2)=v(2);
  outputVector(1,OI_MU)=4.0;
  outputVector(1,OI_TAU)=0;
  outputVector(1,OI_BATT_OBJ)=battObjValAL-ALTerm;
  outputVector(1,OI_VEH_OBJ)=3e7;
  outputVector(1,OI_VEH_OBJ1)=1e7;
  outputVector(1,OI_VEH_OBJ2)=1e7;
  outputVector(1,OI_VEH_OBJ3)=1e7;
  outputVector(1,OI_AL_TERM)=ALTerm;
  outputVector(1,OI_AIO)=battObjValAL+3e7;
%%Writing outputs...
  outputResultsFile=sprintf('./OutputFiles/%s%dCols/output%s%d_colsW%dBeta1e%d.txt',...
    scalType,numberOfColumns,scalType,numberOfColumns,weightIndex,betaExp);
  %outputResultsFile=sprintf('./Output/output%s%d_colsW%dBeta1e%d.txt',...
  %  scalType,numberOfColumns,weightIndex,betaExp);
  fidOut=fopen(outputResultsFile,'wt');
```

```
  fprintf(fidOut,'%s\n',num2str(outputVector,'%10.4f '));
  fclose(fidOut);
%%Prepare linking file to pass to vehicle-level subproblem
  linkingFileName='linking.dat';
  str=sprintf('%e %e %e %e %e %e %e',outputVector(1,OI_BATT_L),outputVector(1,OI_BATT_W),...
    scalingCoefficient*outputVector(1,OI_V1),scalingCoefficient*outputVector(1,OI_V2),...
    scalingCoefficient*outputVector(1,OI_MU)*scalingCoefficient,...
    outputVector(1,OI_VEH_L),outputVector(1,OI_VEH_W),outputVector(1,OI_TAU));
  fid=fopen(linkingFileName,'wt');
  fprintf(fid,'%s\n',str);
  fclose(fid);
  exit;
end
```

The method `solveBL()` manages the main loop iterations of the bilevel solution approach. Its main roles are to determine when to run the vehicle level solver and the battery level solver, and to manage the communication of information between the two. The inputs in common with `initializeMODA()` have the same meaning. The only new input is `status`: status 0 corresponds to a normal run, status 1 corresponds to update of multiplier `v` and penalty coefficient `penaltyWeight`.

```
function retval=solveBL(status,nCols,weightIndex,betaExp,scalCoeff,scalType)
%status:     0-regular run, 1-update augmented Lagrange parameters,
  global spacing_traverse t f w numCols numRows ...
    numberOfColumns numberOfRows cell_diameter...
    v penaltyWeight scalingCoefficient L_target W_target...
    inputVals tempVals
%%In practice, Simulink simulation of battery temperatures is used.
%%For computational testing, this feature is commented out,
%%and interpolated function evaluations
%%for the battery cell temperatures are used instead
  %addpath('../SimulinkModels/');
%%%%keep track of output column indices
```

```
  OI_PCELL=1;
  OI_BATT_L=2;
  OI_BATT_W=3;
  OI_VEH_L=4;
  OI_VEH_W=5;
  OI_V1=6;
  OI_V2=7;
  OI_MU=8;
  OI_TAU=9;
  OI_BATT_OBJ=10; %scalarized battery objective (equitable formulation)
  OI_VEH_OBJ=11; %scalarized vehicle objective
  OI_VEH_OBJ1=12; %three battery-level objectives
  OI_VEH_OBJ2=13;
  OI_VEH_OBJ3=14;
  OI_AL_TERM=15; %summed value augmented Lagrange terms
  OI_AIO=16; %all-in-one, augmented Lagrange, scalarized objective value
  newOutputVector=zeros(1,16);

%%indices referring to the vehicle-level battery box dimensions
%%from the output produced by the vehicle level
  VLBBDimIndices=[26 27];

%%Set number of columns in the battery cell configuration
%%(This is treated as a constant integer-valued parameter)
  numberOfColumns=nCols;
  numberOfRows=72/numberOfColumns;
  cell_diameter=0.034; %%Simulink model requires this be given in meters

%%This information is necessary when using Simulink
% t=[0:1:100]';
% numCols=[t numberOfColumns*ones(size(t))];
```

```
% numRows=[t numberOfRows*ones(size(t))];

%%%scaling is necessary for computational expediency
  scalingCoefficient=scalCoeff;

  tau=0; %this parameter is only used with the proximal variant of
         %the block coordinate descent method (set to zero if not using)

%output files
%%This is the path for storing the most recent pareto.dat file
%%At the end of the MODA process, the pareto%d_colsW%dBeta1e%d.dat contains
%%the final vehicle-level computations
  outputResultsFile=sprintf('./OutputFiles/%s%dCols/output%s%d_colsW%dBeta1e%d.txt',...
    scalType,numberOfColumns,scalType,numberOfColumns,weightIndex,betaExp);
  ParetoPathName=sprintf('./ParetoFiles/%s%dCols/pareto.dat',scalType,numberOfColumns);
  ParetoPathNameSaved=sprintf('./ParetoFiles/%s%dCols/pareto%s%d_colsW%dBeta1e%d.dat',...
    scalType,numberOfColumns,scalType,numberOfColumns,weightIndex,betaExp);

%%Initialize the latest design and objective data from
%%the last valid iteration of battery and vehicle-level computations
  outputData=load(outputResultsFile);
  [dm,dn]=size(outputData);
  mostRecentData=outputData(dm,:);

%%penalty associated with violation of consistency constraints
  penaltyWeight=mostRecentData(OI_MU);

%%multiplier values associated with consistency constraints
  v=mostRecentData([OI_V1, OI_V2]);

  if status==1
```

```
  %%Update Lagrange parameters v and penaltyWeight
  %%using method of multiplier updates

  %%Discrepancies between battery box dimensions determined at each level
  battBoxDiscreps=computeBBDiscreps(mostRecentData([OI_BATT_L,OI_BATT_W]),...
      mostRecentData([OI_VEH_L,OI_VEH_W]),scalingCoefficient);
  v=v+penaltyWeight*battBoxDiscreps;
  penaltyWeight=penaltyWeight*4;

  %%Prepare for battery level computation, use latest output
  %%information with with the AL terms and parameters updated
  newOutputVector=mostRecentData;
  newOutputVector(OI_V1)=v(1);
  newOutputVector(OI_V2)=v(2);
  newOutputVector(OI_MU)=penaltyWeight;
  ALTerms=v*battBoxDiscreps'+(penaltyWeight/2)*norm(battBoxDiscreps,2)^2;
  newOutputVector(OI_AL_TERM)=ALTerms;
  L_target=newOutputVector(OI_VEH_L);
  W_target=newOutputVector(OI_VEH_W);
  %%continue with battery-level computation
else
  %%Moving the pareto.dat file to another location is necessary to
  %%indicate when another vehicle-level computation needs to be performed
  movefile('pareto.dat',ParetoPathName);
  vehData=load(ParetoPathName);

  % select the global minimum (2nd column is the objective function)
  [underhoodObjValAL,i]=min(vehData(:,2));
  if(underhoodObjValAL < mostRecentData(OI_VEH_OBJ)+mostRecentData(OI_AL_TERM))
      %%Save this most recent improving vehicle-level file
      movefile(ParetoPathName,ParetoPathNameSaved);
```

```
        %augmented Lagrange parameters stay the same
        newOutputVector([OI_V1,OI_V2,OI_MU,OI_TAU])=...
            mostRecentData([OI_V1,OI_V2,OI_MU,OI_TAU]);

        %%Write vehicle-level information into new output, prepare for new
        %%battery-level computation

        newOutputVector([OI_VEH_L,OI_VEH_W])=vehData(i,VLBBDimIndices);
        L_target=newOutputVector(OI_VEH_L);
        W_target=newOutputVector(OI_VEH_W);
        newOutputVector([OI_VEH_OBJ1,OI_VEH_OBJ2,OI_VEH_OBJ3])=vehData(i,5:7);

        battBoxDiscreps=computeBBDiscreps(mostRecentData([OI_BATT_L,OI_BATT_W]),...
            newOutputVector([OI_VEH_L,OI_VEH_W]),scalingCoefficient);
        AL_Terms=v*battBoxDiscreps'+(penaltyWeight/2)*norm(battBoxDiscreps,2)^2;
        newOutputVector(OI_VEH_OBJ)=underhoodObjValAL-AL_Terms;

        %%Continue with battery-level computation
    else
        %%Redo vehicle-level run
        exit;
    end
  end
%%% NOW PERFORM OPTIMIZATION ON BATTERY LEVEL %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %loading the approximated temperature data that model would have generated
    %(this is to save time when testing MODA)
    dataFileName=sprintf('./TemperatureData/tempData%d.mat',numberOfColumns);
    load(dataFileName); %%%this initializes inputVals, tempVals
    %these are the weights for the battery level equitable,
    %scalarized reformulation
    weights=generateWeightsEquitable(numberOfColumns);
```

```
    w=weights(weightIndex,:);
    w=w*(10^betaExp);
    %%Run solver, obtain battery-level design updates
    [pcell, battObjValAL]=fminbnd(@computeZ_Approx,1.01,2.0);
    [L W]=computeBoxDims(pcell);
    %%Computing updated augmented Lagrange term
    battBoxDiscreps=...
        computeBBDiscreps([L;W],[L_target;W_target],scalingCoefficient);
    ALTerm=v*battBoxDiscreps+(penaltyWeight/2)*norm(battBoxDiscreps,2)^2;
    newOutputVector(OI_PCELL)=pcell;
    newOutputVector(OI_BATT_L)=L;
    newOutputVector(OI_BATT_W)=W;
    battObjVal=battObjValAL-ALTerm;
    %%Set new battery-level outputs, updated augmented Lagrange information
    newOutputVector(OI_BATT_OBJ)=battObjVal;
    newOutputVector(OI_AL_TERM)=ALTerm;
    newOutputVector(OI_AIO)= newOutputVector(OI_BATT_OBJ)...
        +newOutputVector(OI_VEH_OBJ)+ newOutputVector(OI_AL_TERM);
%%%%% DONE WITH Battery level OPTIMIZATION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Write/append to output files
  fidOut=fopen(outputResultsFile,'a');
  fprintf(fidOut,'%s\n',num2str(newOutputVector,'%10.4f '));
  fclose(fidOut);
%%Preparing linking file to be sent to vehicle level; containts battery
%%level linking information
  str=sprintf('%e %e %e %e %e %e %e',L,W,scalingCoefficient*v(1),...
    scalingCoefficient*v(2),scalingCoefficient*penaltyWeight*scalingCoefficient,...
    L_target,W_target,tau);
%linking.dat contains the linking variable information passed to the
%vehicle level
  linkingFileName='linking.dat';
```

```
  fid=fopen(linkingFileName,'wt');
  fprintf(fid,'%s\n',str);
  fclose(fid);
  retval=0;
  exit;
end
```

B.2 Helper methods:

The methods displayed below are invoked in either `initializeMODA()` or `solveBL()`. These methods are mostly related to function evaluations within the implementation of the battery-level solver.

The method `computeApproxTemps()` approximately evaluates the output that would have been produced by the Simulink model for generating battery cell temperatures. Approximations are based on a linear interpolation of a sample of output that are generated by a sample of input that are near the input `p_cell` value.

```
function  TValsApprox = computeApproxTemps( p_cell )
  global tempVals
%%Since p_cell is restricted to take value [1.01,2.0],
%%index should be a float ranging in value from 1 to 1000 (inclusive)
  index= (p_cell-1.01)/(2.0-1.01)*999+1;
%%At this point, index is a float, but we need to work with integer
%%approximants
  rv1=floor(index);
  rv2=ceil(index);
%%linear interpolation of temperature values sampled from Simulink output
%%where p_cell is between two sampled inputs
  alphaVal=(index-rv1); %%alphaVal takes values in [0,1]
  TValsApprox=(1-alphaVal)*tempVals(rv1,:)...
    +alphaVal*tempVals(rv2,:);
  return
```

```
end
```

The method `computeTemperatures()` is the interface through which battery cell temperatures are provided in the battery-level solver. This function may either use `computeApproxTemps()` or may invoke a Simulink model.

```
function TVals = computeTemperatures( x )
%%Simulink stuff is coded out

%Computes simulated temperatures of battery cell columns
%This function ultimately has to be embedded in a script with
%global variables declared as they are in this function.
  %global spacing_traverse t yout numRows numCols numberOfColumns cell_diameter...
  %spacing_traverse=[t cell_diameter*x*ones(size(t))];
  %sim('BTMS_MODEL_SJPARK_vML');
  %TVals=yout(1000,1:numberOfColumns)';

%%Using approximations to what Simulink would have generated for the
%%purpose of testing MODA
  TVals=computeApproxTemps(x);
%%%returns column vector
  return;
end
```

The method `computeBBDiscreps()` computes the discrepancies between the vehicle-level battery box measure and the battery-level battery box measure, taking into account the scaling of the battery box consistency constraints.

```
function battBoxDiscreps = computeBBDiscreps( LW_Batt, LW_Veh, scalCoeff )
  battBoxDiscreps=scalCoeff*(LW_Batt-LW_Veh);
  return;
end
```

The method `computeF()` computes the battery cell temperature deviations from a target temperature evaluated as least-square terms given an input `x`.

```
function fVal=computeF(x)
  global f numberOfColumns numberOfRows
  TARGET=40; %40 degrees centigrade target temperature
  temperatures=computeTemperatures(x);
  fVal=(temperatures-TARGET).^2; %%computing least square terms
  f=fVal; %%%%column vector
  return;
end
```

The method `computeBoxDims()` computes the dimensions of the battery box based on a spacing factor `p`, the aspect ratio layout of battery cells, and the battery cell diameters.

```
function [ L W ] = computeBoxDims( p )
  global numberOfColumns numberOfRows cell_diameter
  if p <=1.0
    p=1.01;
  end
%lgap=(p-1)*cell_diameter;
  L= cell_diameter*(numberOfColumns-1)*(sqrt(3)/2)*p+cell_diameter*3;
  W=cell_diameter*(numberOfRows+0.5)*p;
  L=L*1000; %convert to mm
  W=W*1000; %convert to mm
  return;
end
```

The method `computeZ_Approx()` evaluates the battery-level objective function summed with the augmented Lagrange terms. The objective function is the weighted-sum scalarization of the equitable reformulation of the battery-level multiobjective subproblem.

```
function zValWAL = computeZ_Approx( x )
  global f numberOfColumns w v L_target W_target ...
    penaltyWeight scalingCoefficient
  [L W]=computeBoxDims(x);
  f=computeF(x);
```

```
%%Evaluate the block-diagonal linear programs' optimal values
%%These are used in computing the objective values
%%of the equitable reformulation
  zVals=zeros(numberOfColumns,1);
  AEq=[];
  bEq=[];
  artificialBds=inf;
  lb=[-artificialBds zeros(1,numberOfColumns)];
  ub=artificialBds*ones(1,1+numberOfColumns);
  A=[-ones(numberOfColumns,1) -eye(numberOfColumns)];
  b=-f;
  options=optimset('LargeScale','off','Simplex','on');
  for i=1:numberOfColumns
   c=[i ones(1,numberOfColumns)];
   [~, zval]=linprog(c,A,b,AEq,bEq,lb,ub,[],options);
   zVals(i)=zval;
  end
%%Aggregate the LP optimal value evaluations using weighted-sum method
  zValW=w*zVals;
%%now add augmented Lagrangian term
  battBoxDiscreps=...
    computeBBDiscreps([L;W],[L_target;W_target],scalingCoefficient);
  zValWAL=zValW+v*battBoxDiscreps...
    +(penaltyWeight/2)*(battBoxDiscreps'*battBoxDiscreps);
return
end
```

The method `generateWeightsEquitable()` generates three weighted-sum weights for the equitable MOP formulation of the battery-level subproblem. The length of the weight vector is the same as the number of battery cell columns specified by the input `numberOfColumns`.

```
%%Generates weights for the weighted-sum scalarization of the equitable
```

```
%%reformulation of the battery-level subproblem
function weights = generateWeightsEquitable( numberOfColumns )
  weights=zeros(3,numberOfColumns);
  weights(1,1)=1;
  for i=1:numberOfColumns
    weights(1,i)=2^(-(i-1));
  end
  weights(1,:)=weights(1,:)/(sum(weights(1,:)));
  weights(2,:)=1;
  weights(2,:)=weights(2,:)/(sum(weights(2,:)));
  for i=1:numberOfColumns
    weights(3,i)=2^(-(numberOfColumns-i));
  end
  weights(3,:)=weights(3,:)/(sum(weights(3,:)));
  return;
end
```

B.3 MS Batch file for running MODA:

The batch script `run_vehicle_design.bat` is invoked to run the bilevel solution approach implementing MODA for the application of automotive design described in chapters 3 and 4. The Java executable `pack.jar` takes as argument a string, either "QS", indicating the vehicle-level use of quadratic scalarization, or otherwise any other string such as "WS" indicates the vehicle-level use of weighted-sum method.

```
set path=%path%;u:\profile.cu\Desktop\ARC_Battery\Bilevel_Vehicle_Design\MatlabCode
set NCOLS=12
set WT=2
set BETA=2
set SCALCOEFF=1/4
set SCALTYPE=WS
set JARFILE=.\JAR_FILES\pack.jar
```

```
set JARARGS=%SCALTYPE%
set MATLABARGS=-nojvm -nosplash -nodesktop -minimize -wait -r
set PATHTOMAINDIR=u:\profile.cu\Desktop\ARC_Battery\Bilevel_Vehicle_Design
set ADDMLPATH=addpath('%PATHTOMAINDIR%\MatlabCode')
set OUTPUTFILE='.\QSOutput%NCOLS%Cols\outputs%NCOLS%_colsW%WT%Beta1e%BETA%.txt'
set SOLVEBLARGS=%NCOLS%,%WT%,%BETA%,%SCALCOEFF%,'%SCALTYPE%'
set MATLABFNCT0="%ADDMLPATH%;solveBL(0,%SOLVEBLARGS%)"
set MATLABFNCT1="%ADDMLPATH%;solveBL(1,%SOLVEBLARGS%)"
del pareto.dat
matlab %MATLABARGS% "%ADDMLPATH%;initializeMODA(%SOLVEBLARGS%)"
for /l %%I in (1, 1, 8) do (
    ECHO Iteration %%I Round 0
    java -jar %JARFILE% %JARARGS%
    matlab %MATLABARGS% %MATLABFNCT0%
)
for /l %%K in (1, 1, 4) do (
ECHO Update v and mu
matlab %MATLABARGS% %MATLABFNCT1%
for /l %%I in (1, 1, 8) do (
    ECHO Iteration %%I Round %%K
    java -jar %JARFILE% %JARARGS%
    matlab %MATLABARGS% %MATLABFNCT0%
)
)
```

Appendix C Maple Code for Chapter 5

The Maple code presented in Appendix C implements the parameter estimation approaches for the modeling of viscoelastic materials carried out in Section 5.3.

C.1 Setup of experimental data:

The first segment of code displayed stores the experimental data for later use. This data includes strain measurements (in %) on the three types of viscoelastic materials and the three loading levels, along with the elapsed times (in hours) at which the strain measurements are taken.

```
with(VectorCalculus) : with(inttrans) : with(CurveFitting) : with(Statistics) :
with(plots) : with(LinearAlgebra) : with(Optimization) : with(ArrayTools) :
SumLim := 2000 : α := 0.83 :

DataSets: Indexed by (i,j), i=1,2,3, j=1,2,3.
i material: 1:PA   2:PA+UDD     3:PA+CNT; j loading level:  1:σ_0.3       2:σ_0.4       3:σ_0.5

T11a := 0.00, 0.01, 0.02, 0.06, 0.10, 0.20, 0.40, 0.70, 1.00, 2.00, 3.00, 4.00, 5.00, 6.00, 7.00, 9.00 :
 T11b := 11.00, 13.00, 15.00, 17.00, 20.00, 23.00, 26.00, 30.00, 35.00, 40.00, 46.00, 52.00, 59.00 :
 T11c := 66.00, 73.00, 80.00, 88.00, 96.00, 105.00, 114.00, 124.00, 156.00, 167.00 :
 S11a := 0.00, 3.90, 5.20, 7.42, 8.30, 9.40, 10.30, 11.40, 12.00, 12.95, 13.50, 14.00, 14.23, 14.51 :
 S11b := 14.79, 15.29, 15.80, 16.29, 16.78, 17.26, 17.93, 18.58, 19.17, 19.79, 20.27, 20.70, 21.00 :
 S11c := 21.30, 21.50, 21.60, 21.91, 22.18, 22.45, 22.70, 23.10, 23.27, 23.46, 24.50, 24.85 :
times[1, 1] := Vector(⟨T11a, T11b, T11c⟩) : strains[1, 1] := Vector(⟨S11a, S11b, S11c⟩) :
 T12a := 0.00, 0.01, 0.02, 0.06, 0.50, 0.80, 1.20, 1.50, 1.80, 2.50, 4.00, 5.00, 6.00, 7.00, 9.00, 11.00 :
 T12b := 13.00, 15.00, 17.00, 20.00, 23.00, 26.00, 30.00, 35.00, 40.00, 50.00, 65.00, 75.00, 85.00 :
 T12c := 95.00, 105.00, 124.00, 135.00, 145.00, 156.00, 167.00 :
 S12a := 0.00, 4.50, 6.10, 8.40, 12.00, 13.80, 15.00, 15.70, 16.20, 17.10, 18.03, 18.50, 19.05, 19.50 :
 S12b := 20.30, 21.00, 21.80, 22.62, 23.28, 24.07, 24.70, 25.23, 25.80, 26.40, 27.14, 27.80, 28.70 :
 S12c := 29.30, 30.00, 30.70, 31.20, 32.07, 32.60, 33.00, 33.42, 34.00 :
times[1, 2] := Vector(⟨ T12a, T12b, T12c⟩) : strains[1, 2] := Vector(⟨S12a, S12b, S12c⟩) :
 T13a := 0.00, 0.02, 0.08, 0.17, 0.25, 0.33, 0.50, 0.67, 0.83, 1.00, 1.25, 1.50, 2.00, 2.50, 3.00 :
 T13b := 3.50, 5.00, 7.00, 10.00, 13.00, 15.00, 19.58, 23.58, 30.00, 37.00 :
 T13c := 44.58, 52.08, 60.08, 76.08, 90.58, 120.25, 164.58, 171.08 :
 S13a := 0.00, 7.00, 9.60, 12.00, 13.20, 14.40, 15.40, 16.20, 17.00, 17.60, 18.20, 18.80 :
 S13b := 19.60, 20.40, 21.00, 21.40, 23.00, 24.30, 25.50, 26.80, 27.70, 29.20, 30.20 :
 S13c := 31.58, 33.10, 34.40, 35.60, 36.40, 38.10, 39.40, 41.20, 43.40, 43.70 :
times[1, 3] := Vector(⟨T13a, T13b, T13c⟩) : strains[1, 3] := Vector(⟨S13a, S13b, S13c⟩) :
```

$T21a$:= 0.00, 0.01, 0.02, 0.06, 0.10, 0.20, 0.40, 0.70, 1.00, 1.30, 1.70, 2.10, 2.60, 3.20, 4.00 :
$T21b$:= 5.00, 6.00, 7.00, 9.00, 11.00, 13.00, 15.00, 17.00, 20.00, 23.00, 26.00, 30.00, 35.00, 66.00 :
$T21c$:= 73.00, 80.00, 88.00, 96.00, 105.00, 114.00, 124.00, 135.00, 145.00, 156.00, 167.00 :
$S21a$:= 0.00, 3.50, 4.20, 6.35, 7.08, 8.25, 9.40, 10.63, 11.35, 11.45, 12.32, 12.48, 12.69, 12.73 :
$S21b$:= 12.95, 13.20, 13.47, 13.73, 14.23, 14.76, 15.26, 15.75, 16.21, 16.91, 17.55, 18.15, 18.75 :
$S21c$:= 19.23, 20.90, 21.12, 21.41, 21.74, 22.05, 22.48, 22.80, 23.00, 23.50, 23.80, 24.00, 24.40 :
$\text{times}_{2,1}$:= Vector(⟨$T21a$, $T21b$, $T21c$⟩) : $strains_{2,1}$:= $Vector$(⟨$S21a$, $S21b$, $S21c$⟩) :

$T22a$:= 0.00, 0.01, 0.02, 0.06, 0.10, 0.20, 0.40, 0.70, 1.00, 1.30, 1.70, 2.10, 3.00, 4.00, 5.00, 6.00 :
$T22b$:= 8.00, 10.00, 13.00, 16.00, 18.00, 20.00, 25.00, 30.00, 35.00, 40.00, 46.00, 51.00, 57.00, 65.00 :
$T22c$:= 73.00, 80.00, 90.00, 105.00, 114.00, 124.00, 135.00, 145.00, 156.00, 167.00 :
$S22a$:= 0.00, 4.70, 5.40, 8.00, 10.20, 10.70, 11.90, 12.80, 13.40, 14.00, 14.00, 15.00, 15.50, 16.10 :
$S22b$:= 17.02, 18.05, 18.50, 19.24, 20.32, 21.05, 21.50, 22.00, 23.00, 23.80, 24.70, 25.50, 25.88, 26.54 :
$S22c$:= 27.06, 27.68, 28.35, 28.80, 29.30, 29.80, 30.20, 30.53, 30.88, 31.22, 31.50, 31.80 :
$times_{2,2}$:= $Vector$(⟨ $T22a$, $T22b$, $T22c$⟩) : $strains_{2,2}$:= $Vector$(⟨$S22a$, $S22b$, $S22c$⟩) :

$T23a$:= 0.00, 0.02, 0.08, 0.17, 0.25, 0.33, 0.50, 0.67, 0.83, 1.00, 1.25, 1.50, 2.00 :
$T23b$:= 2.50, 3.00, 3.50, 7.00, 10.00, 13.00, 15.00, 19.43, 23.43, 33.43, 44.43 :
$T23c$:= 51.93, 66.93, 75.93, 90.43, 105.43, 120.10, 140.43, 164.43, 170.93 :
$S23a$:= 0.00, 6.20, 9.00, 11.15, 12.24, 13.10, 14.00, 14.86, 15.56, 16.05, 16.60 :
$S23b$:= 17.20, 18.30, 19.00, 19.70, 20.00, 22.70, 23.50, 25.00, 26.00, 27.20, 28.00 :
$S23c$:= 30.70, 32.30, 33.20, 34.40, 35.04, 35.94, 36.80, 37.50, 38.60, 39.60, 39.90 :
$times_{2,3}$:= $Vector$(⟨$T23a$, $T23b$, $T23c$⟩) : $strains_{2,3}$:= $Vector$(⟨$S23a$, $S23b$, $S23c$⟩) :

$T31a$:= 0.00, 0.01, 0.02, 0.06, 0.10, 0.20, 0.40, 0.70, 1.00, 2.60, 3.20, 4.00, 5.00, 6.00, 7.00, 9.00, 11.00 :
$T31b$:= 13.00, 15.00, 17.00, 20.00, 23.00, 26.00, 30.00, 35.00, 40.00, 46.00, 52.00, 59.00, 66.00 :
$T31c$:= 73.00, 80.00, 88.00, 96.00, 105.00, 114.00, 124.00, 135.00, 145.00, 156.00, 167.00 :
$S31a$:= 0.00, 2.50, 3.00, 4.70, 5.80, 6.90, 8.40, 9.72, 10.20, 11.05, 11.29, 11.50, 11.61, 11.81, 12.02 :
$S31b$:= 12.41, 12.80, 13.16, 13.55, 13.90, 14.45, 14.95, 15.41, 15.91, 16.39, 16.67, 16.90, 17.30 :
$S31c$:= 17.60, 17.80, 18.10, 18.35, 18.61, 18.90, 19.20, 19.55, 19.85, 20.15, 20.41, 20.71, 21.00 :
$times_{3,1}$:= Vector(⟨ $T31a$, $T31b$, $T31c$⟩) : $strains_{3,1}$:= $Vector$(⟨$S31a$, $S31b$, $S31c$⟩) :

$T32a$:= 0.00, 0.01, 0.02, 0.06, 0.10, 0.20, 0.40, 0.70, 1.00, 1.30, 1.70, 2.10, 2.60 :
$T32b$:= 3.20, 7.00, 10.00, 15.00, 20.00, 30.00, 35.00, 46.00, 65.00, 80.00 :
$T32c$:= 96.00, 105, 114.00, 124.00, 135.00, 145.00, 156.00, 167.00 :
$S32a$:= 0.00, 3.40, 4.00, 5.20, 6.40, 7.70, 9.20, 10.00, 10.90, 11.70, 12.00, 12.40 :
$S32b$:= 12.80, 13.00, 15.20, 16.30, 18.14, 19.80, 21.30, 21.94, 22.86, 23.95 :
$S32c$:= 24.70, 25.45, 25.8, 26.17, 26.54, 26.76, 26.98, 27.15, 27.35 :
$times_{3,2}$:= $Vector$(⟨$T32a$, $T32b$, $T32c$ ⟩) : $strains_{3,2}$:= $Vector$(⟨$S32a$, $S32b$, $S32c$⟩) :

$T33a$:= 0.00, 0.02, 0.08, 0.17, 0.25, 0.33, 0.50, 0.67, 0.83, 1.00, 1.25, 1.50, 2.00, 2.50, 3.00, 3.50 :
$T33b$:= 6.00, 11.00, 13.00, 19.48, 23.48, 44.48, 51.98, 75.98, 90.48, 120.15, 164.48, 170.98 :
$S33a$:= 0.00, 4.60, 7.20, 8.20, 9.00, 9.80, 10.60, 11.40, 11.80, 12.20, 12.60, 13.00, 13.80, 15.40 :
$S33b$:= 16.34, 17.00, 18.75, 21.90, 22.80, 24.86, 25.60, 28.90, 29.68, 31.80, 32.60, 33.80, 34.70, 34.80 :
$times_{3,3}$:= $Vector$(⟨$T33a$, $T33b$⟩) : $strains_{3,3}$:= $Vector$(⟨$S33a$, $S33b$⟩) :

C.2 Setup of regression function terms and other experimentally determined parameters

These are experimentally determined regression function terms on the experimental data, along with experimentally determined model parameters $\varepsilon 0$ for each test.

$$RegTerms_{1,1} := \left[t^{\frac{1}{10}}, t^{\frac{1}{15}}, t^{\frac{1}{20}} \right] : RegTerms_{1,2} := \left[t^{\frac{1}{5}}, t^{\frac{1}{10}}, t^{\frac{1}{20}} \right] : RegTerms_{1,3} := \left[t^{\frac{1}{5}}, t^{\frac{1}{10}}, t^{\frac{1}{20}} \right] :$$

$$RegTerms_{2,1} := \left[t^{\frac{1}{5}}, t^{\frac{1}{10}}, t^{\frac{1}{20}} \right] : RegTerms_{2,2} := \left[t^{\frac{1}{8}}, t^{\frac{1}{16}} \right] : RegTerms_{2,3} := \left[t^{\frac{1}{8}}, t^{\frac{1}{16}} \right] :$$

$$RegTerms_{3,1} := \left[t^{\frac{1}{5}}, t^{\frac{1}{10}} \right] : RegTerms_{3,2} := \left[t^{\frac{1}{15}}, t^{\frac{1}{20}} \right] : RegTerms_{3,3} := \left[1, e^{-0.05 \cdot t}, t^{\frac{1}{16}}, t^{\frac{1}{20}} \right] :$$

$$\varepsilon 0_{1,1} := \frac{16.20}{955} : \varepsilon 0_{1,2} := \frac{21.60}{955} : \varepsilon 0_{1,3} := \frac{27.00}{955} :$$

$$\varepsilon 0_{2,1} := \frac{15.90}{1008} : \varepsilon 0_{2,2} := \frac{21.20}{1008} : \varepsilon 0_{2,3} := \frac{26.50}{1008} :$$

$$\varepsilon 0_{3,1} := \frac{18.72}{1320} : \varepsilon 0_{3,2} := \frac{24.96}{1320} : \varepsilon 0_{3,3} := \frac{31.20}{1320} :$$

C.3 Time-domain least-squares parameter estimation

This code segment carries out time-domain least-squared optimal parameter estimates based on the experimental data.

```
for i from 1 by 1 to 3 do
 for j from 1 by 1 to 3 do
  ExamineIndex := i, j :
  εT_ExamineIndex := ε0_ExamineIndex · ( 1 + lambdaVal·t^(1 − α)
     ·evalf( Sum( ( (-betaVal·t^(1 − α))^(SumLim−n) / Γ( (1 − α)·(SumLim − n + 1) + 1 ), n = 0 ..SumLim ) ) ) :
  numData := Size( times_ExamineIndex )[1] :
  for k from 1 by 1 to numData do Term_k := ( eval( εT_ExamineIndex, [ t = times_ExamineIndex[k] ] )
     - strains_ExamineIndex[k] )^2 :end do:
  TEstimates_ExamineIndex := LSSolve( [ seq( Term_k, k = 1 ..numData ) ], initialpoint = { betaVal = 0.0,
     lambdaVal = 600 } ) :
  εEvalT_ExamineIndex := eval( εT_ExamineIndex, [ betaVal = rhs( TEstimates_ExamineIndex[2, 1] ), lambdaVal
     = rhs( TEstimates_ExamineIndex[2, 2] ) ] ) :
  PlotArg1 := [ εEvalT_ExamineIndex, t = 0 ..170, font = [ Times, bold, 16 ], linestyle = solid, thickness = 4,
     legend = typeset( "model" ), color = "DarkGrey" ] :
  PlotArg2PtStyle := style = point, symbol = circle, color = black, symbolsize = 18 :
  PlotArg2 := [ times_ExamineIndex, strains_ExamineIndex, font = [ Times, bold, 16 ], PlotArg2PtStyle, legend
     = typeset( "experimental data" ) ] :
  CreepPlotsT_ExamineIndex := multiple( plot, PlotArg1, PlotArg2, legendstyle = [ location = top ], resolution
     = 600 ) :
 end do:
end do:
display( [ CreepPlotsT_1,1, CreepPlotsT_1,2, CreepPlotsT_1,3 ] );
display( [ CreepPlotsT_2,1, CreepPlotsT_2,2, CreepPlotsT_2,3 ] );
display( [ CreepPlotsT_3,1, CreepPlotsT_3,2, CreepPlotsT_3,3 ] ) ;
```

C.4 Laplace-domain least-squares parameter estimation

The following segment of code initializes the sample of s values, sets plotting arguments, and defines the method `solveForBetaLambda` used for solving the Laplace-domain least squares problem.

```
i := 1 : numTiers := 10 : numSlices := 10 : ϵ := 10^-6 : c := 10^-2 : d := 0.1 :
 for k from 0 by 1 to numTiers − 1 do
   for j from 0 by 1 to numSlices − 1 do
     sVal_i := ϵ + ( c + k/(numTiers − 1) · (d − c) ) · evalf( e^( (π · j/(numSlices − 1) − π/2) · I ) ) : i := i + 1 :
   end do:
 end do:
numS := i − 1 :

 PlotArgs := style = point, color = black, axes = boxed, font = [Times, 12], scaling = constrained :
complexplot( [seq(sVal_i, i = 1 ..numS)], PlotArgs, view = [0 ..0.2, -0.1 ..0.1] ) :

solveForBetaLambda := proc(AMat, BVec, numS, SVals)
 local i, Factor1, T1, T2, T3, Term, SolveData
  for i from 1 by 1 to numS do
   Factor1
     := 1 / ( norm(SVals_i, 2) · ( x_1^2 + Re( conjugate( SVals_i^(1 − α) ) + SVals_i^(2 − α) ) · x_1
     + norm( SVals_i^(1 − α), 2 )^2 ) ) :
   T1 := norm(AMat[i, 1], 2)^2 · x_1^2 + Re(conjugate(AMat[i, 1]) · AMat[i, 2] + AMat[i, 1]
     · conjugate(AMat[i, 2]) ) · x_1 · x_2 :
   T2 := norm(AMat[i, 2], 2)^2 · x_2^2 + Re(AMat[i, 1] · conjugate(BVec[i, 1]) + conjugate(AMat[i, 1])
     .BVec[i, 1]) · x_1 :
   T3 := Re(AMat[i, 2] · conjugate(BVec[i, 1]) + conjugate(AMat[i, 2]).BVec[i, 1]) · x_2
     + norm(BVec[i, 1], 2)^2 :
   Term_i := Factor1 · (T1 + T2 + T3) :
  end do:
  SolveData := LSSolve( [seq( Term_i, i = 1 ..numS)], initialpoint = {x_1 = 0.0, x_2 = 600} ) :
  return SolveData :
end proc:

ALLS := Matrix(numS, 2) : bLLS := Matrix(numS, 1) :
 for i from 1 by 1 to numS do
   ALLS[i, 1] := (ϵ_0 − sL_i) : ALLS[i, 2] := ϵ_0 : bLLS[i, 1] := s_i^(1 − α) · (ϵ_0 − sL_i) :
 end do:
```

Finally, the Laplace-domain least-squares optimization is carried out for the 9 tests, computed parameters are stored, and plots are generated comparing the experimental data, the interpolated data, and the model obtained with the optimal parameter estimates.

```
for i from 1 by 1 to 3 do
for j from 1 by 1 to 3 do
  IF[i,j] := LinearFit(RegTerms[i,j], times[i,j], strains[i,j], t); LT[i,j] := s·laplace(IF[i,j], t, s);
  LVal[i,j] := Vector(numS) :
 for k from 1 by 1 to numS do
  LVal[i,j][k] := eval(LT[i,j], [s = sVal[k]]) :
 end do:

AValLLS[i,j] := eval(ALLS, [ε0 = ε0[i,j]]) : bValLLS[i,j] := eval(bLLS, [ε0 = ε0[i,j]]) :
 for k from 1 by 1 to numS do
   AValLLS[i,j] := eval(AValLLS[i,j], [s[k] = sVal[k], sL[k] = LVal[i,j][k]]) :
  bValLLS[i,j] := eval(bValLLS[i,j], [s[k] = sVal[k], sL[k] = LVal[i,j][k]]) :
 end do:
 solveStuff := solveForBetaLambda(AValLLS[i,j], bValLLS[i,j], numS, [seq(sVal[k], k = 1 ..numS)]);
 β[i,j] := rhs(solveStuff[2, 1]) : λ[i,j] := rhs(solveStuff[2, 2]) : print([β[i,j], λ[i,j]]) :
ε[i,j] := ε0[i,j]·(1 + λ[i,j]·t^(1 − α)·evalf(Sum((−β[i,j]·t^(1 − α))^(SumLim − n) / Γ((1 − α)·(SumLim − n + 1) + 1), n = 0 ..SumLim))) :
PlotArg1 := [ε[i,j], t = 0 ..170, font = [Times, bold, 16], linestyle = solid, thickness = 4, legend
    = typeset("model"), color = "Red"] :
PlotArg2 := [IF[i,j], t = 0 ..170, font = [Times, bold, 16], linestyle = dot, color = black, thickness = 5,
   legend = typeset("regressed data")] :
PlotArg3PtStyle := style = point, symbol = circle, color = black, symbolsize = 18 :
PlotArg3 := [times[i,j], strains[i,j], font = [Times, bold, 16], PlotArg3PtStyle, legend
    = typeset("experimental data")] :
CreepPlots[i,j] := multiple(plot, PlotArg1, PlotArg2, PlotArg3, legendstyle = [location = top], resolution
    = 600) :
end do:
end do:
display([seq(CreepPlots[1,j], j = 1 ..3)], labels = [" ", " "], view = [0 ..170, 0 ..45]);
display([seq(CreepPlots[2,j], j = 1 ..3)], labels = [" ", " "], view = [0 ..170, 0 ..45]);
display([seq(CreepPlots[3,j], j = 1 ..3)], labels = [" ", " "], view = [0 ..170, 0 ..45]);
```

Bibliography

[1] S. Al Hallaj and J.R. Selman. A novel thermal management system for electric vehicle batteries using phase-change material. *Journal of the Electrochemical Society*, 147(9):3231–3236, 2000.

[2] S. Al-Hallaj and J.R. Selman. Thermal modeling of secondary lithium batteries for electric vehicle/hybrid electric vehicle applications. *Journal of Power Sources*, 110(2):341 – 348, 2002.

[3] D. Baatar and M. M. Wiecek. Advancing equitability in multiobjective programming. *Computers & Mathematics with Applications*, 52(1-2):225–234, 2006.

[4] M. L. Balinski and P. (Eds.) Wolfe. Nondifferentiable optimization. In *In Mathematical Programming Study 3, Amsterdam: North-Holland*. 1975.

[5] T. M. Bandhauer, S. Garimella, and T. F. Fuller. A critical review of thermal issues in lithium-ion batteries. *Journal of the Electrochemical Society*, 158(3):R1, 2011.

[6] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms, 2nd Edition*. Wiley, 1993.

[7] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming Theory and Algorithms, 2nd Edition*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 1993.

[8] H.P. Benson. Existence of efficient solutions for vector maximization problems. *Journal of Optimization Theory and Applications*, 26(4):569–580, 1978.

[9] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation, Numerical Methods*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1989.

[10] D.P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.

[11] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[12] J.C. Bezdek, R.J. Hathaway, R.E. Howard, C.A. Wilson, and M.P. Windham. Local convergence analysis of a grouped variable version of coordinate descent. *Journal of Optimization Theory and Applications*, 54:471–477, 1987.

[13] V. J. Bowman. On the relationship of the Tchebycheff norm and the effcient frontier of multiple-criteria objectives. In H. Thieriez, editor, *Multiple Criteria Decision Making*, volume 130 of *Lecture Notes in Economics and Mathematical Systems*, pages 76–85. Springer Verlag, Berlin, 1976.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundation and Trends in Machine Learning*, 3(1):1–122, 2011.

[15] T.A. Burton. *Volterra Integral and Differential Equations*. Mathematics in Science and Engineering Series. Elsevier Science Limited, 2005.

[16] M.I. Campbel, C. H. Amon, and J. Cagan. Optimal three-dimensional placement of heat generating electronic components. *Journal of Electronic Packaging*, 119(2):106–113, 1997.

[17] V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making Theory and Methodology*. Elsevier Science, New York, 1983.

[18] A. Charnes and W. Cooper. *Management Models and Industrial Applications of Linear Programming*. John Wiley and Sons, New York, 1961.

[19] R. V. Churchill. *Operational Mathematics, 2nd Edition*. McGraw-Hill Book Company, Inc., 1958.

[20] F. Ciucci, T. Honda, and M. C. Yang. An information-passing strategy for achieving Pareto optimality in the design of complex systems. *Research in Engineering Design*, 23(1):71–83, 2012.

[21] J. B. Conway. *Functions of One Complex Variable I: Second Edition*. Springer, 1978.

[22] N. Cristello and I.Y. Kim. Multidisciplinary design optimization of a zero-emission vehicle chassis considering crashworthiness and hydroformability. In *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, volume 221, pages 511–526, 2007.

[23] B. Dandurand. *Mathematical Optimization for Engineering Design Problems*. PhD thesis, Clemson University, 2013.

[24] B. Dandurand, P. Guarneri, G. Fadel, and M. M. Wiecek. Equitable multiobjective optimization applied to the design of a hybrid electric vehicle battery. *Journal of Mechanical Design*, 135(4):041004, 2013.

[25] B. Dandurand, I. Viktorova, S. Alekseeva, and S. Goodson. Nonlinear modeling and optimization of parameters for viscoelastic composites and nanocomposites. *Problems of Machine Building and Automatization*, (3):51–57, 2011.

[26] I. Das. A preference ordering among various Pareto optimal alternatives. *Structural optimization*, 18:30–35, 1999.

[27] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, New York, NY, 2001.

[28] G. Dellino, P. Lino, C. Meloni, and A. Rizzo. Multidisciplinary design optimization of a pressure controller for CNG injection systems. In *Proceedings of the 2006 IEEE Conference on Computer Aided Control Systems Design, CACSD, October 4-6, 2006*, pages 2689–2694, 2007.

[29] H. Dong, P. Guarneri, and G. Fadel. Bi-level approach to vehicle component layout with shape morphing. *Journal of Mechanical Design*, 133(4):041008, 2011.

[30] J. Donndelinger, S. Ferguson, and K. Lewis. Exploring mass trade-offs in preliminary vehicle design using Pareto sets. In *Collection of Technical Papers*, volume 3 of *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages 1806–1815. 2006.

[31] L. M. Graña Drummond and B. F. Svaiter. A steepest descent method for vector optimization. *Journal of Computational and Applied Mathematics*, 175:395–414, 2005.

[32] J. Eckstein and D. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55:293–318, 1992.

[33] M. Ehrgott. Discrete decision problems, multiple criteria optimization classes and lexicographic max-ordering. In T. J. Stewart and R. C. van den Honert, editors, *Trends in Multicriteria Decision Making*, volume 465 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, 1998.

[34] M. Ehrgott. *Multicriteria optimization, 2nd Edition*. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, 2005.

[35] M. Ehrgott and E. A. Galperin. Min-max formulation of the balance number in multiobjective global optimization. *Computers & Mathematics with Applications*, 44(7):899–907, 2002.

[36] M. Ehrgott and M.M. Wiecek. Multiobjective programming. In J. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 667–722. Springer Verlag, Boston, Dordrecht, London, 2005.

[37] A. Engau and M. M. Wiecek. 2d decision-making for multicriteria design optimization. *Structural and Multidisciplinary Optimization*, 34(4):301–315, 2007.

[38] J. Fliege, L. M. Graña Drummond, and B. F. Svaiter. Newton's method for multiobjective optimization. *SIAM Journal on Optimization*, 20(2):602–626, 2009.

[39] J. Fliege and B. F. Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods for Operations Research*, 51:479–494, 2000.

[40] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers and Mathematics with Applications*, 2:17–40, 1976.

[41] E. A. Galperin. Nonscalarized multiobjective global optimization. *Journal of Optimization Theory and Applications*, 75(1):69–85, 1992.

[42] E. A. Galperin. Set contraction algorithm for computing Pareto set in nonconvex nonsmooth multiobjective optimization. *Mathematical and Computer Modelling*, 40(7-8):847–859, 2004.

[43] E. A. Galperin and M. M. Wiecek. Retrieval and use of the balance set in multiobjective global optimization. *Computers & Mathematics with Applications*, 37(4/5):111–123, 1999.

[44] M. Gardenghi, T. Gómez, F. Miguel, and M. M. Wiecek. Algebra of efficient sets for multiobjective complex systems. *Journal of Optimization Theory and Applications*, 149(2):385–410, 2011.

[45] Melissa Gardenghi and Margaret M. Wiecek. Efficiency for multiobjective multidisciplinary optimization problems with quasiseparable subproblems. *Optimization and Engineering*, 13:293–318, 2012.

[46] A. M. Geoffrion. Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22:618–630, 1968.

[47] R. Glowinski and A. Marrocco. Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualité, d'une classe de problems de dirichlet non lineares. *Revue Française d'Automatique, Informatique, et Recherche Opérationelle*, 9:41–76, 1975.

[48] G. Grimmett and D. Welsh. *Probability: An Introduction*. Oxford Science Publications, 1998.

[49] L. Grippo and M. Sciandrone. On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. *Operations Research Letters*, 26(3):127–136, 2000.

[50] S. Gunawan, S. Azarm, J. Wu, and A. Boyars. Quality-assisted multi-objective multidisciplinary genetic algorithms. *American Institute of Aeronautics and Astronautics*, 41(9):1752–62, 2003.

[51] S. Gunawan, A. Frahang-Mehr, and S. Azarm. Multi-level multi-objective genetic algorithm using entropy to preserve diversity. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 148–161. 2003.

[52] S. Gunawan, A. Frahang-Mehr, and S. Azarm. On maximizing solution diversity in a multiobjective multidisciplinary genetic algorithm for design optimization. *Mechanics Based Design of Structures and Machines*, 32(4):491–514, 2004.

[53] D. Han and X. Yuan. A note on the alternating direction method of multipliers. *Journal of Optimization Theory and Applications*, 155:227–238, 2012.

[54] B. He, Z. Peng, and X. Wang. Proximal alternating direction-based contraction methods for separable linearly constrained convex optimization. http://www.optimization-online.org/DB_HTML/2010/05/2628.html, 2010.

[55] B. He, M. Tao, and X. Yuan. Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM Journal on Optimization*, 22(2):313–340, 2012.

[56] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, pages 303–320, 1969.

[57] C. Hildreth. A quadratic programming procedure. *Naval Research Logistics Quarterly*, 4:79–85, 361, 1957.

[58] T. Honda, F. Ciucci, K. Lewis, and M.C. Yang. A comparison of information passing strategies in system level modeling. *International Design Engineering Technical Conferences, Montreal, Canada*, 2010.

[59] T. Honda, F. Ciucci, and M. C. Yang. Achieving Pareto optimality in a decentralized design environment. Aug. 24-27, 2009. Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 5, Design Methods and Tools (pt. 1), pp. 501-511, Palo Alto, CA.

[60] C.-H. Huang, J. Galuski, and C.L. Bloebaum. Multi-objective Pareto concurrent subspace optimization for multidisciplinary design. *American Institute of Aeronautics and Astronautics*, 45(8):1894–1906, 2007.

[61] Ch.-H. Huang. *Development of Multi-objective Concurrent Subspace Optimization and Visualization Methods for Multidisciplinary Design*. PhD thesis, State University of New York at Buffalo, Buffalo, NY, 2003.

[62] Ch.-H. Huang and C.L. Bloebaum. Incorporation of preferences in multi-objective concurrent subspace optimization for multidisciplinary design. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004. AIAA 2004-4548.

[63] Ch.-H. Huang and C.L. Bloebaum. Multi-objective Pareto concurrent subspace optimization for multidisciplinary design. In *Proceedings of the 42nd AIAA Aeropsace Sciences Meeting and Exhibit*, 2004. AIAA 2004-278.

[64] Ch.-H. Huang and C.L. Bloebaum. Visualization as a solution aid for multi-objective concurrent subspace optimization in a multidisciplinary design environment. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004. AIAA 2004-4464.

[65] Anthony Jarrett and Il Yong Kim. Design optimization of electric vehicle battery cooling plates for thermal performance. *Journal of Power Sources*, 196(23):10359 – 10368, 2011.

[66] C.D. Jilla and D.W. Miller. Multi-objective, multidisciplinary design optimization methodology for distributed satellite systems. *Journal of Spacecraft and Rockets*, 41(1):39–50, 2004.

[67] I. Kaliszewski. A modified weighted Tchebycheff metric for multiple objective programming. *Computers and Operations Research*, 14(4):315–323, 1987.

[68] R. T. Katrgadda, S. R. Gondipalle, P. Guarneri, and G. Fadel. Predicting the thermal perfomance of for the multi-objective vehicle underhood packing optimization problem. Aug. 12-15, 2012. Proceedings of ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference IDETC/CIE 2012, Chicago, IL, paper DETC2012-71098.

[69] S. Kontogiorgis and R. R. Meyer. A variable-penalty alternating directions method for convex optimization. *Mathematical Programming*, 83:29–53, 1998.

[70] G. C. Kopsidas. A new Pareto optimal solution in a Lagrange decomposable multi-objective optimization problem. *The Journal of the Operational Research Society*, 42(5):401–411, 1991.

[71] M.M. Kostreva and W. Ogryczak. Linear optimization with multiple equitable criteria. *RAIRO - Operations Research*, 33:275–297, 6 1999.

[72] M.M Kostreva, W. Ogryczak, and A. Wierzbicki. Equitable aggregations and multiple criteria analysis. *European Journal of Operations Research*, 158(2):362–377, 2004.

[73] J.B. Lassiter, M.M. Wiecek, and K.R. Andrighetti. Lagrangian coordination and analytical target cascading: Solving ATC-decomposed problems with Lagrangian duality. *Optimization and Engineering*, 6:361–381, 2005.

[74] D.S. Lee, L.F. Gonzalez, K. Srinivas, D.J. Auld, and J. Periaux. Multi-objective/multidisciplinary design optimisation of blended wing body UAV via advanced evolutionary algorithms. In *Collection of Technical Papers*, volume 1 of *45th AIAA Aerospace Sciences Meeting*, pages 296–316. 2007.

[75] D. Li. Convexification of a noninferior frontier. *Journal of Optimization Theory and Applications*, 88:177–196, 1996.

[76] D. Li and Y.Y. Haimes. Hierarchical generating method for large-scale multiobjective systems. *Journal of Optimization Theory and Applications*, 54:303–333, 1987.

[77] H. Liu, X. Yin, and W. Chen. Setting performance targets based on subsystem Pareto frontiers in multilevel optimization. In *Proceedings of IMECE 2006 ASME International Mechanical Engineering Congress and Exposition November 5-10, 2006, Chicago, Illinois, USA*, 2006.

[78] R. Mahamud and C. Park. Reciprocating air flow for li-ion battery thermal management to improve temperature uniformity. *Journal of Power Sources*, 196(13):5685 – 5696, 2011.

[79] R.A.E. Makinen, J. Periaux, and J. Toivanen. Multidisciplinary shape optimization in aerodynamics and electromagnetics using genetic algorithms. *International Journal for Numerical Methods in Fluids*, 30(2):149–159, 1999.

[80] Y. Miao, Y. B. Vincent, and G. M. Fadel. Multi-objective configuration optimization with vehicle dynamics applied to midsize truck design. In *Proc. ASME 2003 DETC03*, number DAC48735, page 319327, 2003.

[81] A. Mills and S. Al-Hallaj. Simulation of passive thermal management system for lithium-ion battery packs. *Journal of Power Sources*, 141(2):307 – 315, 2005.

[82] M. Mut and M. M. Wiecek. Generalized equitable preference in multiobjective programming. *European Journal of Operational Research*, 212(3):535 – 551, 2011.

[83] A. Ruszczyński. *Nonlinear Optimization*. Princeton University Press, 2006.

[84] Office of Naval Research. Special Notice 13-SN-0009 Special Program Announcement for 2013 Office of Naval Research Research Opportunity: Computational Methods for Decision Making. Accessed May 2013.

[85] Ramadass P., Haran B., White R., and Popov B.N. Capacity fade of Sony 18650 cells cycled at elevated temperatures - part i. cycling performance. *Journal of Power Sources*, 112(2):606–613, 2002.

[86] Ramadass P., Haran B., White R., and Popov B.N. Capacity fade of Sony 18650 cells cycled at elevated temperatures - part ii. capacity fade analysis. *Journal of Power Sources*, 112(2):614–620, 2002.

[87] S. Pan, S. He, and X. Li. Smoothing method for minimizing the sum of the r largest functions. *Optimization Methods and Software*, 22(2):267–277, 2007.

[88] S. Parashar and C.L. Bloebaum. Multi-objective genetic algorithm concurrent subspace optimization (MOGACSSO) for multidisciplinary design. 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Newport, RI, May 1-4, 2006.

[89] V. Pareto. *Manual d'économie politique*. F. Rouge, Lausanne, 1896.

[90] Y. G. Peng, A. Ganesh, J. Wright, W. L. Xu, and Y. Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 763–770, 2010.

[91] D. Peri and E.F. Campana. Multidisciplinary design optimization of a naval surface combatant. *Journal of Ship Research*, 47(1):1–12, 2003.

[92] S. Ponnusamy. *Foundations of Functional Analysis*. Alpha Science, 2002.

[93] M. J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*. New York: Academic Press, 1969.

[94] M.J.D. Powell. On search directions for minimization algorithms. *Mathematical Programming*, 4:193–201, 1973.

[95] S. Rabeau, P. Dépincé, and F. Bennis. Collaborative optimization of complex systems: a multidisciplinary approach. *International Journal on Interactive Design and Manufacturing*, 1:209–218, 2007.

[96] Yu. N. Rabotnov. *Creep Problems in Structural Members*. North-Holland Series in Applied Mathematics and Mechanics, Vol. 7. 1969.

[97] R.T. Rockafellar. Augmented Lagrange multiplier functions and duality in nonconvex programming. *SIAM Journal on Control*, 12:268–285.

[98] Rami Sabbah, R. Kizilel, J.R. Selman, and S. Al-Hallaj. Active (air-cooled) vs. passive (phase change material) thermal management of high power lithium-ion packs: Limitation of temperature rise and uniformity of temperature distribution. *Journal of Power Sources*, 182(2):630 – 638, 2008. Selected papers from the International Workshop on Degradation Issues in Fuel Cells.

[99] O. Schütze, K. Witting, S. Ober-Blöbaum, and M. Dellnitz. Set oriented methods for the numerical treatment of multiobjective optimization problems. In E. Tantar, A.-A. Tantar, P. Bouvry, P. Del Moral, P. Legrand, C. A. Coello Coello, and O. Schütze, editors, *EVOLVE-A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation*, volume 447 of *Studies in Computational Intelligence*, pages 187–219. Springer Berlin Heidelberg, 2013.

[100] V. K. Singh. *Equitable Efficiency in Multicriteria Optimization, Ph.D. Dissertation.* PhD thesis, Clemson University Department of Mathematical Sciences, 2007.

[101] R.B. Statnikov and J.B. Matusov. *Multicriteria Optimization and Engineering.* Chapman & Hall, New York, NY, 1995.

[102] R. E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application.* John Wiley & Sons, New York, 1985.

[103] R. E. Steuer and E.-U. Choo. An interactive weighted Tchebycheff procedure for multiple objective programming. *Mathematical Programming*, 26(3):326–344, 1983.

[104] S. Szykman and J. Cagan. A simulated annealing-based approach to three-dimensional component packing. *Journal of Mechanical Design*, 117(2):308–314, 1995.

[105] M. Tao and X.M. Yuan. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM J. Optim.*, 21:5781, 2011.

[106] R.A. Tapia. Diagonalized multiplier methods and quasi-Newton methods for constrained optimization. *Journal of Optimization Theory and Applications*, 22:135–194, 1977.

[107] R. V. Tappeta and J. E. Renaud. Multiobjective collaborative optimization. *Journal of Mechanical Design*, 119(3):403–411, 1997.

[108] K. Tarvainen and Y.Y. Haimes. Coordination of hierarchical multiobjective systems: Theory and methodology. *Systems, Man and Cybernetics, IEEE Transactions on*, 12(6):751–764, Nov.

[109] Natick MA USA The MathWorks, Inc. *Optimization Toolbox Users Guide Version R2011b.* 2011.

[110] Natick MA USA The MathWorks, Inc. *Simulink Version R2011b.* 2011.

[111] S. Thomas, G. Zaikov, S. Valsaraj, and A. Meera. *Recent Advances in Polymer Nanocomposites: Synthesis and Characterisation.* Brill, 2010.

[112] J. Tind and M. M. Wiecek. Augmented Lagrangian and Tchebycheff approaches in multiple objective programming. *Journal of Global Optimization*, 14:251–266, 1999.

[113] S. Tosserams, L. F. P. Etman, P. Y. Papalambros, and J. E. Rooda. An augmented Lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers. *Structural and Multidisciplinary Optimization*, 31:176–189, 2006.

[114] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:475–494, 2001.

[115] I. Viktorova, B. Dandurand, S. Alekseeva, and M. Fronya. The modeling of creep for polymer-based nanocomposites using an alternative nonlinear optimization approach. *Mechanics of Composite Materials*, 48(6):1–14, 2012.

[116] V. Volterra. *Theory of Functionals and of Integral and Integrodifferential Equations*. Dover Books on Mathematics. Dover Publications, 1959.

[117] A. Tamir W. Ogryczak. Minimizing the sum of the k largest functions in linear time. *Inf. Process. Lett.*, 85(3):117–122, February 2003.

[118] J. Warga. Minimizing certain convex functions. *SIAM Journal on Applied Mathematics*, 11:588–593, 1963.

[119] D. J. White. Weighting factor extensions for finite multiple objective vector minimization problems. *European Journal of Operational Research*, 36:256–265, 1988.

[120] Li Y, Lu Z., and Michalek J.J. Diagonal quadratic approximation for parallelization of analytical target cascading. *Journal of Mechanical Design*, 130(5), 2008.

[121] P. L. Yu. A class of solutions for group decision making. *Management Science*, 19:936–946, 1973.

[122] Yu.V.Suvorova. On the nonlinear hereditary type equation by Yu. N. Rabotnov and its applications. *Mechanics of Solids*, (1):174–181, 2004.

[123] A. Zakarian. *Nonlinear Jacobi and Epsilon-Relaxation Methods for Parallel Network Optimization*. PhD thesis, University of Wisconsin, 1995.

[124] M. Zeleny. Compromise programming. *In J.L. Cochrane and M. Zeleny, editors, Multiple Criteria Decision Making*, pages 262–301, 1973.

[125] K.-S. Zhang, Z.-H. Han, W.-J. Li, and W.-P. Song. Bilevel adaptive weighted sum method for multidisciplinary multi-objective optimization. *American Institute of Aeronautics and Astronautics*, 46(10):2611–2622, 2008.

[126] M. Zolot, A. A. Pesaran, and M. Mihalic. Thermal evaluation of Toyota Prius battery pack. In *Proc. 2002 Future Car Congress, Arlington, VA, 2002, SAE 2002-01-1962*, 2002.