CARNEGIE MELLON UNIVERSITY

A Trust Region Filter Algorithm for Surrogate-based Optimization

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree of

DOCTOR OF PHILOSOPHY

in

CHEMICAL ENGINEERING

by

JOHN P. EASON

B.S., CHEMICAL ENGINEERING, THE UNIVERSITY OF TULSA

Pittsburgh, Pennsylvania

Apr, 2018

Acknowledgments

First, I would like to extend my deepest thanks to my adviser Larry Biegler for all he has done for me over these years, for his extensive knowledge, patience, and constant encouragement. His devotion to his work and to his students is inspiring.

I would like to thank the members of my committee, Professors Ignacio Grossmann, Nick Sahinidis, and Bruno Sinopoli for their time, input, and advice, which has helped shape the direction of this thesis. I would also like to thank Professor Selen Cremaschi for leading me towards graduate school and training the research fundamentals that have allowed me to succeed.

Thanks to all of the members of the Biegler group for the welcoming and encouraging environment for research, and for their friendship. Throughout my studies, I have had the opportunity to collaborate closely with several fellow students and visiting researchers. Special thanks to Alex Dowling, both for the fruitful collaboration on the power plant optimization and for helping me navigate the beginning of grad school. Though not included in this thesis, collaborations with Haoshui Yu and Dehao Zhu were an enjoyable and important piece of my graduate studies. Thanks to Prof. Xi Chen for the memorable visit to Zhejiang University and Jiayuan Kang for collaboration on the surrogate equation of state project.

I would like to thank Dr. David Miller for forming and managing the Carbon Capture Simulation Initiative (CCSI) and the Institute for the Design of Advanced Energy Systems (IDAES), which have provided partial support for this work. Thanks also to my many colleagues at NETL, including Tony Burgard, John Eslick, and Andrew Lee for the stimulating discussions. Special thanks to Jinliang Ma, whose boiler simulation code helped form a central motivating application in this work. Thanks also to John Siirola and the team at Sandia for helping the trust region code live on past this thesis. Finally, I gratefully acknowledge support from the NSF Graduate Research Fellow Program under grant number DGE-1252522.

Finally, I would like to thank my family. Thanks to my parents for their love and encouragement, and especially thanks to my wife Wei. Through her help with brainstorming, as proofreader, and as code adviser, her influence is present throughout this thesis, but most of all thanks for her unwavering love and support. This would not have been possible without her.

John P. Eason
Pittsburgh, PA
Apr 2018

Abstract

Modern nonlinear programming solvers can efficiently handle very large scale optimization problems when accurate derivative information is available. However, black box or derivative free modeling components are often unavoidable in practice when the modeled phenomena may cross length and time scales. This work is motivated by examples in chemical process optimization where most unit operations have well-known equation oriented representations, but some portion of the model (e.g. a complex reactor model) may only be available with an external function call.

The concept of a surrogate model is frequently used to solve this type of problem. A surrogate model is an equation oriented approximation of the black box that allows traditional derivative based optimization to be applied directly. However, optimization tends to exploit approximation errors in the surrogate model leading to inaccurate solutions and repeated rebuilding of the surrogate model. Even if the surrogate model is perfectly accurate at the solution, this only guarantees that the original problem is feasible. Since optimality conditions require gradient information, a higher degree of accuracy is required.

In this work, we consider the general problem of hybrid glass box/black box optimization, or gray box optimization, with focus on guaranteeing that a surrogate-based optimization strategy converges to optimal points of the original detailed model. We first propose an algorithm that combines ideas from SQP filter methods and derivative free trust region methods to solve this class of problems. The black box portion of the model is replaced by a sequence of surrogate models (i.e. surrogate models) in trust region subproblems. By carefully managing surrogate model construction, the algorithm is guaranteed to converge to true optimal solutions. Then, we discuss how this algorithm can be modified for effective application to practical problems. Performance is demonstrated on a test set of benchmarks as well as a set of case studies relating to chemical process optimization. In particular, application to the oxycombustion carbon capture power generation process leads to significant efficiency improvements. Finally, extensions of surrogate-based optimization to other contexts is explored through a case study with physical properties.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Modeling, simulation, and optimization tools have become essential tools for engineers in many industries. By leveraging together physical insights, mathematics, and advanced computing, lower cost, more efficient systems can be developed with greater reliability. In the chemical process industries, simulation tools are already ubiquitous for process design and operations. Process simulators can predict the performance of a plant for a specified design configuration and at certain operating conditions. Mathematical optimization techniques have great potential to further accelerate the design process. However, several barriers remain to widespread adoption in the process industries.

The continuous advancement of simulation technology also acts as an impediment to adoption of optimization methods. Developments in computational fluid dynamics and molecular dynamics provide ever more simulation tools to engineers. However, these methods often use specialized computational strategies to find solutions. Optimization algorithms usually require explicit mathematical models, implemented in a way that is compatible with the solution algorithm. Therefore, optimization usually must be considered as a goal at the early stages of developing a model.

Powerful algorithms and software exist for solving many classes of optimization problems. One can classify a problem based on the presence of constraints (absent, inequality, equality), the presence of nonlinearity, convexity (of objective, constraints, or feasible region), presence of discrete decisions, and amount of information obtained from the model

(derivative free, first-order, second-order, or other methods). Many common process decisions are continuous variables, including temperature, pressure, and flowrates. Discrete decisions could include whether or not a certain piece of equipment is purchased, and require the use of mixed integer optimization techniques. In this thesis, we will not consider the case of discrete decisions and instead focus on continuous optimization problems. Because of detailed thermodynamic and reaction models, chemical process optimization is usually represented as nonconvex nonlinear optimization, or nonlinear programming (NLP) problems. The presence of safety or materials limitations often imposes bounds on variables, leading to inequality constraints as well. This class of optimization problems includes many important applications in chemical engineering including process optimization and optimal control. An NLP can be written mathematically as

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ s.t. \quad & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{1.1}$$

The objective function is $f : \mathbb{R}^n \to \mathbb{R}$, equality constraints are $h : \mathbb{R}^n \to \mathbb{R}^{m_e}$ with $m_e \leq n$, and inequalities constraints are $g : \mathbb{R}^n \to \mathbb{R}^{m_i}$. These functions are assumed to be sufficiently smooth, i.e. with Lipschitz continuous first derivatives on a compact domain.

Several choices exist for the solution of NLPs. Problems encountered in chemical process optimization may be high dimensional, (with thousands or tens of thousands variables/-constraints) but they are also highly sparse, allowing for efficient linear algebra routines. This makes second-order Newton-type methods highly effective for these problems. However, these methods requires accurate derivative information. If a mathematical model is implemented in an algebraic modeling language, derivatives are automatically obtained through automatic differentiation. However, it is not always possible to put a model within an optimization modeling framework. Many practitioners prefer to use previously estab-

lished and validated simulation models for optimization. In addition, complex simulations such as computational fluid dynamics require highly specialized methods to solve the mathematical models, which may consist of large scale partial differential algebraic equations. Therefore, there is significant demand for black box optimization techniques in process engineering.

Black box optimization is when the optimization solver only has access to input-output data from an objective function. This can be used directly with simulation software, but there are several drawbacks. Optimization is most useful when there are many degrees of freedom and intuition may be insufficient to guide decision making. However, black box optimization can only efficiently operate on problems with relatively few degrees of freedom (for example on the order 10-20). In addition, the elimination of equality constraints greatly slows progress since these equations must be fully simulated at each black box call. Finally, the lack of accurate derivatives also harms the convergence rate of the optimization, so precise solutions may be difficult to obtain.

In this thesis, we attempt to bridge the gap between conventional NLP and black box optimization. This is captured by the term "glass box/black box" optimization, or "gray box optimization," where glass box refers to equations that yield accurate derivatives, and black box refers to parts of the model that are derivative free, for example generated by an external simulation call. This work is motivated by the case of process optimization problems where one or more unit models are modeled with external simulations, yet we emphasize the generality for a wider class of optimization problems. In particular, applications in thermodynamic property calls are shown to be promising. The problem structure

of interest can be written as:

$$\begin{aligned} \min \quad & f(z, w, d(w)) \\ \text{s.t.} \quad & h(z, w, d(w)) = 0 \\ & g(z, w, d(w)) \leq 0 \end{aligned} \tag{1.2}$$

where minimization is over $z \in \mathbb{R}^n$ and $w \in \mathbb{R}^m$. w represents the inputs to the black box function, and z represents the remaining decision variables. $d(w) : \mathbb{R}^m \to \mathbb{R}^p$ represents the outputs of the black box as a function of inputs w. It is assumed that all functions f, h, g, d are twice continuously differentiable, although derivatives may be unavailable for $d(w)$.

The past approach to solving these "glass box/black box" optimization problems has largely been through the use of surrogate models. Surrogate models, also known as reduced models or metamodels, are equation-based approximations of a typically black box function. These can range from polynomial interpolation to proper orthogonal decomposition, and from Gaussian process regression to neural networks. Regardless, the input-output data from the black box function is converted into equations that may be used directly in optimization and other analysis. The construction of accurate surrogate models is itself an active area of research, with contributions from approximation theory, machine learning, statistics, and engineering. An appropriate choice of experimental design, functional form, and validation method will depend on the context in which a surrogate model is used. In this thesis, we focus on the use of surrogate models for optimization. Instead of specifying one particular methodology, we study the behavior of surrogates in optimization and place guarantees on the performance. This is accomplished through combining concepts from derivative free optimization with classical constrained nonlinear optimization theory.

1.1 Contributions

The main contributions of this thesis are summarized as follows:

1. A novel trust region filter (TRF) algorithm for glass box/black box optimization is proposed, which is both robust to lack of derivatives in some constraints while also taking advantage of equation oriented constraints.
2. The convergence of the TRF algorithm to first-order KKT points is proved. This is the first provably convergent algorithm specifically designed for this problem class.
3. The TRF algorithm is implemented in Pyomo with interfaces to multiple choices of surrogate model. The code is easily extensible to other choices of surrogates.
4. A test set for glass box/black box optimization was assembled from established NLP benchmarks. The resulting test set is used to validate the performance of the TRF algorithm and demonstrate the effect of algorithmic improvements.
5. The concept of a sampling region, first proposed by Powell for unconstrained derivative free optimization, is revisited in the context of glass box/black box optimization. The sampling region is able to greatly improve performance of the TRF algorithm. This is the first time that the effect of the sampling region has been directly examined.
6. The TRF algorithm is applied to several case studies in process engineering. Most notably, an oxycombustion carbon capture plant was optimized with an embedded detailed boiler simulation. The combination of modeling detail and a large system boundary identified new design concepts and increased efficiency.

1.2 Research Statement and Dissertation Overview

This dissertation is organized as follows: In Chapter 2, the basics of nonlinear optimization theory are reviewed, including the KKT conditions and constraint qualifications. These

concepts will be important for characterizing solutions of glass box/black box optimization and developing algorithmic solution approaches. Then, we will review some (derivative-based) solution methods for NLPs as well as basic categories of derivative-free optimization methods.

Chapter 3 presents a brief review of surrogate-based optimization approaches and applications. Surrogate model (reduced model) approaches are classified into data-driven and model-driven approaches, and general guidelines about their use are presented. Then, a certain class of derivative free optimization methods, model-based trust region methods, is reviewed. These basic trust region concepts are later expanded upon throughout the thesis.

Chapter 4 introduces a novel trust region filter algorithm for solving glass box/black box optimization problems. When equation based surrogate models are used in place of the black box, NLP solvers may be applied directly but an accurate solution is not guaranteed. Trust region concepts are used to manage the surrogate models and adaptively adjust their fit to manage errors. By combining concepts from trust region filter methods and derivative free optimization, the method guarantees convergence to first order critical points of the original glass box/black box problem. Algorithmic concepts are introduced and convergence is proved.

Chapter 5 discusses practical aspects of the trust region filter method. We discuss several modifications to the basic algorithm introduced in Chapter 4. This includes the sampling region, which maintains the algorithm's global convergence properties without requiring the trust region to shrink to zero in the limit. Additional modifications to the filter mechanism and trust region update formulas are also shown to improve performance. To benchmark the development of this optimization method, a test set of problems is generated based on modified problems from the CUTEr and COPS sets. The modified algorithm

demonstrates improved performance using the test problem set. In addition, we discuss the implementation of the method connected to the Pyomo modeling language.

Chapter 6 summarizes all case studies that have been solved with the trust region filter method. Initial studies on the Williams-Otto provide insight into the effect of functional form and experimental design for surrogate models used with the trust region method. The ammonia synthesis process is studied as a medium-scale process optimization problem. The reactor is treated as a black box, and the trust region filter performance is validated against a full simultaneous discretization approach of the reactor equations. Next, we study large-scale power plant optimization. The boiler of the power plant is represented by a detailed simulation code that requires several minutes to be solved externally using CFD strategies. The boiler is integrated into both conventional and oxycombustion (carbon capture) power plants and the models are optimized to maximize thermal efficiency. The results show that rigorous optimization can reduce the carbon capture efficiency penalty from 8-10% to under 6%. Next, a solid sorbent carbon capture process is studied to minimize the utility usage subject to a carbon capture requirement. The results demonstrate the advantage of the sampling region concept to improve performance on otherwise intractable problems. Finally, the application of the TRF algorithm to thermodynamic properties is investigated through the concept of a surrogate equation of state.

Chapter 7 concludes the dissertation, summarizes main contributions and outlines recommendations for future work.

Chapter 2

Nonlinear Optimization Background

This chapter will briefly review the basics of nonlinear programming theory as well as popular solution approaches. First, optimality conditions are reviewed, including first and second order KKT conditions and a discussion on constraint qualifications. Then, nonlinear programming (NLP) algorithmic concepts are introduced, with brief discussion of major classes of algorithms and summary of popular software.

2.1 Optimality Conditions

In this thesis, we focus on optimization methods that obtain locally optimal solutions. In this section, optimality conditions that characterize locally optimal solutions are reviewed. Recall the general formulation of an NLP, where bounds and inequalities are represented as $g_i(x) \leq 0$. This gives the following formulation:

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{2.1}$$

where we define decision variables $x \in \mathbb{R}^n$, objective function $f : \mathbb{R}^n \to \mathbb{R}$, equality constraints $h : \mathbb{R}^n \to \mathbb{R}^{m_e}$, and inequality constraints $g : \mathbb{R}^n \to \mathbb{R}^{m_i}$. The constraints define a feasible set, which we will refer as

$$\mathcal{X} := \{x \in \mathbb{R}^n : h(x) = 0,\ g(x) \leq 0\}$$

Then, a local minimum of NLP (2.1) is defined as follows:

Definition 1 (Local minimum). *A point x^* is a local minimum of (2.1) if $x^* \in \mathcal{X}$ and there exists $\epsilon > 0$ such that:*

$$f(x) \geq f(x^*) \quad \forall\, x \in \mathcal{X} \cap \{\|x - x^*\| \leq \epsilon\}. \tag{2.2}$$

2.1.1 First Order KKT Conditions

In the case of unconstrained optimization $\min f(x)$, the first order necessary optimality conditions are given by $\nabla f(x) = 0$. Therefore, many unconstrained optimization methods work in part by attempting to solve the system of nonlinear equations defined by this condition. Similarly, constrained optimization methods are often based on solving the system defined by the KKT conditions. The first order Karush-Kuhn-Tucker (KKT) conditions are the necessary optimality conditions for a local minimum x^* of constrained NLP (2.1). To introduce the KKT conditions, first define the Lagrangian function:

$$\mathcal{L}(x, \lambda, \eta) = f(x) + h(x)^T \lambda + g(x)^T \eta \tag{2.3}$$

An additional concept called a constraint qualification will be introduced in the following section. The KKT necessary conditions are the result of the following theorem:

Theorem 2.1.1. *If x^* is a local minimum of (2.1) and a constraint qualification holds at x^*, then there exist multipliers λ^* and η^* such that*

$$\begin{aligned} \nabla_x \mathcal{L}(x^*, \lambda^*, \eta^*) = \nabla f(x^*) + \nabla h(x^*)\lambda^* + \nabla g_j(x^*)\eta^* &= 0 \\ h(x^*) &= 0 \\ g(x^*) &\leq 0 \\ \eta^* &\geq 0 \\ g(x^*)^T \eta^* &= 0 \end{aligned} \tag{2.4}$$

2.1.2 Constraint Qualifications (CQs)

Constraint qualifications (or regularity conditions) provide some restriction on relation of the feasible set and the equations used to represent it. If no constraint qualification holds, an optimal solution may or may not satisfy the KKT conditions (2.4) There are a variety of constraint qualifications proposed in the literature. In general, there is trade-off between strict conditions that are easily computable and abstract conditions that may be difficult or impossible to verify in practice. The "weakest" possible constraint qualification for use with the KKT theorem, sometimes called the Guignard constraint qualification, is discussed in [1]. However it is clear that it is usually impossible to compute a certificate of satisfaction for this condition, except through checking strictly stronger conditions. In practice, the following two constraint qualifications are popular due to their ease of computation and intuitive interpretation.

Define $\mathcal{I} = \{1, \ldots, m_e\}$ and $\mathcal{J} = \{1, \ldots, m_i\}$ as the index sets for equality and inequality constraints respectively. The j^{th} inequality constraint is written $g_j(x)$.

Given a point x, we define the notion of an active set of inequalities $\mathcal{A}(x) \subseteq \mathcal{J}$ as follows:

$$\mathcal{A}(x) = \{j \mid g_j(x) = 0\} \tag{2.5}$$

Now we define the linear independence constraint qualification (LICQ) and Mangasarian-Fromovitz constraint qualification (MFCQ) as follows:

Definition (LICQ). Given a point x and corresponding active set $\mathcal{A}(x)$, LICQ is defined by linear independence of the constraint gradients $\{\nabla h_i(x), \nabla g_j(x)\}$ for all $i \in \mathcal{I}$ and $j \in \mathcal{A}(x^*)\}$.

Definition (MFCQ). Given a point x and corresponding active set $\mathcal{A}(x)$, MFCQ is defined by linear independence of the equality constraint gradients and the existence of a search direction p such that $\nabla h_i(x)^T p = 0$ for all $i \in \mathcal{I}$ and $\nabla g_j(x)^T p < 0$ for all $j \in \mathcal{A}(x)$.

Of these two, LICQ is strictly stronger since it is easy to show that satisfaction of LICQ implies satisfaction of MFCQ. If an optimal point x^* satisfies LICQ, it can be shown that there exist unique KKT multipliers λ^* and η^* [2]. If MFCQ holds, a weaker result states that the set of multipliers lies within a bounded polytope [3]. LICQ and MFCQ are both sufficiently strong to satisfy "sequential optimality conditions," where sequences of primal and dual variables that approaching satisfaction of (2.4) will in fact converge to a KKT point [4]

2.1.3 Second Order Conditions

To obtain sufficient optimality conditions, one must look to second order information. In unconstrained optimization, the second order sufficient conditions are straightforward: If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, then x^* is a strict minimizer of f. To extend this to constraints, one must consider the curvature of the objective function in feasible directions. Define the following cone:

$$d \in \mathcal{C}(x, \lambda, \eta) \iff \begin{cases} \nabla h(x)^T d = 0, \\ \nabla g_j(x)^T d = 0, \quad j \in \{i \mid g_i(x) = 0, \lambda_i > 0\} \\ \nabla g_j(x)^T d \leq 0, \quad j \in \{i \mid g_i(x) = 0, \lambda_i = 0\} \end{cases} \tag{2.6}$$

The second order sufficient conditions are given by the following theorem:

Theorem 2.1.2. *Suppose that x^* and multipliers λ^*, η^* satisfy the KKT conditions* (2.4) *and*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*, \eta^*) d > 0 \quad \textit{for all nonzero } d \in \mathcal{C} \tag{2.7}$$

then x^* is a strict local solution of (2.1)

Second order information can also be used to obtain additional *necessary* optimality conditions. For the unconstrained case: If x^* is a minimizer of f, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite. In the constrained case:

Theorem 2.1.3. *If x^* is a local minimum of (2.1), LICQ holds at x^*, and λ^*, η^* are the multipliers that satisfy the KKT conditions* (2.4)*, then:*

$$d^T \nabla_{xx} \mathcal{L}(x^*, \lambda^*, \eta^*) d \geq 0 \quad \forall\, d \in \mathcal{C} \tag{2.8}$$

For proofs of these theorems, see for example [2].

2.2 NLP Algorithms

This section briefly reviews gradient-based approaches to nonlinear programming in the form (2.1). Several algorithmic features are reviewed separately. NLP algorithms generate a sequence $\{x_k\}$ that converges to a KKT point of (2.1). Algorithms can be classified by several features, including the approach to generate a new point x_{k+1} and globalization strategy. Common approaches for each of these are reviewed below.

2.2.1 Step Generation

In this section, we review various methods for step generation for nonconvex constrained NLP. Derivatives are assumed to be available and matrix factorizations tractable.

2.2.1.1 Early NLP Methods

Early work on NLP worked in the framework of sequential unconstrained minimization. These algorithms have been almost completely superseded by SQP and interior point approaches but certain theoretical concepts are still used in many algorithms today. The main idea is to transform a constrained optimization problem into an approximating unconstrained problem. Each iterate in the overall NLP algorithm is therefore generated by the solution of an unconstrained problem. The unconstrained subproblems are updated

either according to a predetermined scheme or adaptively based on the result of the previous subproblem. Unconstrained subproblems can be solved with classical approaches such as Newton's method, perhaps after applying some smoothing to the underlying problem. There are two main approaches to transfer the constraints to the objective function. Penalty functions, as the name imply, penalize the violation of the constraint with a penalty parameter $\rho \in \mathbb{R}^+$

$$f_{pen}(x, \rho) = f(x) + \rho \|h(x)\| + \rho \| \max(0, g(x))\| \tag{2.9}$$

The penalty function is related to the Lagrangian, and as a result it can be shown that there is a sufficiently large penalty parameter ρ such that x^* being a local minimum of (2.1) implies that x^* is also a local minimum of (2.9). Therefore, the only task is to find a penalty parameter sufficiently large, and to converge to a local minimum that is also feasible for (2.1). Since algorithms would start with a low value of ρ to keep the problem well conditioned, iterates would be infeasible with respect to (2.1) until a sufficiently large value of ρ is found.

Barrier methods provide an alternative way to move inequality constraints to the objective function. The issues of nonsmoothness in the norm and max operator in (2.9) are partially avoided, but this approach requires a starting point that satisfies $g(x) < 0$. The barrier term is formed as follows:

$$f_{bar}(x, \mu) = f(x) - \mu \sum_{j \in \mathcal{J}} \log(-g_j(x)) \tag{2.10}$$

The augmented Lagrangian is another approach for moving equality constraints to the objective function. For an equality-constrained NLP, the Augmented Lagrangian function is defined as:

$$L_A(x, \lambda, \rho) = f(x) + h(x)^T \lambda + (\rho/2) h(x)^T h(x) \tag{2.11}$$

where λ is an estimate of the multipliers. Compared to (2.9), the Augmented Lagrangian function is smooth, facilitating the solution of subproblems. The multiplier estimates λ are normally generated in an outer loop, for example using the least squares estimate:

$$\lambda(x) = -[\nabla h(x)^T \nabla h(x)]^{-1} \nabla h(x)^T \nabla f(x) \tag{2.12}$$

2.2.1.2 Sequential Quadratic Programming

Sequential Quadratic Programming (SQP) methods generate steps through the solution of constrained optimization subproblems, namely quadratic programs. These quadratic programs are in a sense local approximations of the original nonlinear program. This approach can be derived from the KKT necessary optimality conditions. For an equality constrained optimization problem, the first-order KKT conditions state

$$\begin{aligned} \nabla f(x^*) + A(x^*)\lambda^* &= 0 \\ h(x^*) &= 0 \end{aligned} \tag{2.13}$$

where $A(x) = \nabla h(x)$. If Newton's method is applied to solve this system for x and λ, we generate sequences $\{x_k\}$ and $\{\lambda_k\}$. The Newton step is then written as

$$\begin{bmatrix} W_k & A_k \\ A_k^T & 0 \end{bmatrix} \begin{pmatrix} d_k^x \\ d_k^\lambda \end{pmatrix} = - \begin{pmatrix} \nabla f(x_k) + A_k \lambda_k \\ h(x_k) \end{pmatrix} \tag{2.14}$$

where $W_k = \nabla_{xx} \mathcal{L}(x_k, \lambda_k)$ and $A_k = \nabla h(x_k)$ and new iterates are obtained by $x_{k+1} = x_k + \alpha d_k^x$ and $\lambda_{k+1} = \lambda_k + \alpha d_k^\lambda$, where $\alpha \in (0, 1]$ is a suitable stepsize.

Note that the solution of this linear system is in the same as finding a critical point of the quadratic program

$$\begin{aligned} \min \quad & \nabla f(x_k)^T d + \tfrac{1}{2} d^T W_k d \\ s.t. \quad & h(x_k) + A_k^T d = 0 \end{aligned} \tag{2.15}$$

Through repeated solutions of the QP subproblems, the sequence of points is generated to move towards the solution.

Extension to handle inequality constraints can be through the application of active set concepts: subsets of inequalities are assumed to be active ($g_j(x) = 0$) and the QP can be solved as normal. Alternatively the active set selection can be included within the QP solution process using linearized inequalities.

If W_k is positive definite on the null space of A_k^T, then d_k^x is the solution of QP (2.15). This observation forms the basis of SQP methods. The inequality constraints can be handled by an active set strategy in the QP, where active constraints are guessed and treated as equalities. It can be shown that under mild assumptions the linearized QP can identify the correct active set. See [2, 5, 6, 7] for more details on SQP.

2.2.1.3 Interior Point Methods

Interior Point methods generate points with an inner-outer loop approach. Throughout the algorithm, the barrier approach is used to eliminate inequality constraints. This is typically presented in the context of bound constrained optimization (general inequalities can be transformed to bound constraints with the addition of slack variables):

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t. } h(x) = 0, \;\; x \geq 0 \tag{2.16}$$

After moving bound constraints to the objective function, the barrier subproblem is formed as follows:

$$\min_{x \in \mathbb{R}^n} \varphi_\mu(x) = f(x) - \mu \sum_{i=1}^{n} \ln(x^{(i)}) \;\; \text{s.t. } h(x) = 0, \tag{2.17}$$

where $\mu \geq 0$ is the barrier parameter and $\varphi_\mu(x)$ is called the barrier function. The solutions of (2.17) converge to the solution of (2.16) as $\mu \to 0$ under certain conditions, see [3] or [8] for details.

After introducing dual variables

$$v^{(i)} = \frac{\mu}{x^{(i)}}$$

the KKT conditions of (2.17) are equivalent to the *primal-dual equations*:

$$\begin{aligned} \nabla f(x) + A(x)\lambda - v &= 0 \\ h(x) &= 0 \\ XVe - \mu e &= 0 \end{aligned} \tag{2.18}$$

Interior point methods solve the primal dual equations in inner iterations using Newton's method and control μ in an outer iteration.

2.2.2 Globalization

Since most NLP methods are based in Newton's method, it is important to consider the effect of starting point. To guarantee convergence from any starting point, a variety of globalization strategies have been proposed. Globalization consists of two pieces: a metric of progress towards the solution and a mechanism to generate a better step if sufficient progress is not made. In classical unconstrained optimization, the obvious metric of acceptance is decrease in the objective function. However, for convergence purposes it is usually better to require *sufficient* decrease in the objective function. Sufficient decrease normally can be measured as "actual reduction over predicted reduction." (This is the typical explanation in the trust region literature, whereas line search literature does not usually present this as a ratio). In the Armijo condition, predicted reduction is the amount the objective function would reduce based on a linearization at a point x_k. Therefore, the Armijo condition [9] can be written as:

$$\frac{f(x_k) - f(x_k + s_k)}{s_k^T \nabla f(x_k)} > \eta \tag{2.19}$$

for some constant η. If this holds, then the step s_k is "acceptable" because it gives sufficient reduction. If, however, the condition (2.19) does not hold, then the algorithm has to take some recourse to find a new step s_k. This is typically accomplished through a line search or trust region approach.

In a line search method, the step is adapted by reassigning $x_{k+1} = x_k + \alpha s_k$ for some $\alpha \in (0, 1]$. This means that the search direction is maintained, but the step length is modified. As long as $s_k^T \nabla f(x_k) < 0$, then it can be shown that there exists some sufficiently small step size in the direction of s_k that will satisfy (2.19). This is usually guaranteed by the design of the algorithm.

Trust region methods, in contrast, change both the step size and direction when seeking to find a step with sufficient improvement. In algorithms using a trust region, every step generation subproblem (e.g. a QP subproblem in SQP) contains an extra constraint, bounding the maximum possible step size. In unconstrained optimization with Newton's method, the addition of the trust region means that each iteration solves the following subproblem:

$$\begin{aligned} \min_s \quad & m(s) := s^T \nabla f(x_k) + \frac{1}{2} s^T \nabla^2 f(x_k) s \\ s.t. \quad & \|s\| \leq \Delta_k \end{aligned} \tag{2.20}$$

If the solution of this subproblem does not satisfy the sufficient decrease condition

$$\frac{f(x_k) - f(x_k + s_k)}{-m(s)} > \eta \tag{2.21}$$

then the trust region radius Δ_{k+1} is assigned to a smaller value and the subproblem (2.20) is solved again. This means that both step size and step direction may be changed. Trust region methods may use more effort in step generation but this can lead to better step directions and fewer iterations required. It also helps provide robust performance. Line search methods, on the other hand, tend to be faster and somewhat better for poorly scaled problems.

When moving to constrained problems, the notion of sufficient progress towards the solution becomes more ambiguous. Now, a new point must be judged both by decrease in the objective and by constraint satisfaction. Two of the most common ways of balancing

these two goals to judge progress are merit functions and filters. The use of merit functions derives from early penalty function algorithms. A step may be generated using an SQP or interior point approach, but progress towards the solution is judged by its ability to reduce the sum of the objective and penalized constraint violation. If a step s_k sufficiently reduces the merit function, then it is acceptable. However, similar to penalty function methods, there is a drawback that the appropriate value of the penalty parameter is difficult to predict a priori.

Filter methods were developed by Fletcher and Leyffer [10] as an alternative to merit functions that eliminate the need for estimating a penalty parameter. Instead of a single measure for both objective and constraints, filter methods borrow concepts from multi-objective optimization to somewhat separate these two goals. Compared to merit functions, filters tend to accept more steps, which can lead to fewer recomputed steps and larger steps, while still guaranteeing global convergence. In addition, it is somewhat more robust to scaling compared to merit functions. The filter approach will be discussed in detail in Chapter 4.

2.3 NLP Software

This section provides a brief, incomplete survey of popular software packages for NLP. This helps provide a context of existing tools and motivates the addition of a new method described in Chapter 4

1. **CONOPT** [11]: CONOPT includes several active set NLP solvers, including a gradient projection method, a sequential linear programming method, and an SQP-type method. These are automatically selected or even nested together.

2. **filterSQP** [10]: Filter SQP is a trust region SQP method, using the filter method for globalization. It uses the *bqpd* package to solve QPs. In the case of indefinite Hessians, a local QP solution is returned.
3. **IPOPT** [12]: IPOPT uses a filter line search globalization for an interior point method. It is available through the open-source COIN-OR project.
4. **KNITRO** [13]: The KNITRO package includes the interior point method described in [14] as well as an SLQP algorithm. Globalization is through merit function line search or a trust region approach depending on the features. The KKT matrices can be solved with either a direct factorization or indirect conjugate gradient method.
5. **LANCELOT** [15]: LANCELOT is an augmented Lagrangian method using a trust region approach. After moving equalities to the objective with the augmented Lagrangian function, the bound constrained trust region subproblem is solved with conjugate gradient steps.
6. **LOQO** [16]: LOQO uses a line search combined with elements of a filter, with recourse to a merit function in certain circumstances. Step generation is with the interior point approach.
7. **MINOS** [17]: MINOS is a reduced space augmented Lagrangian method. The augmented Lagrangian subproblem is solved with linearized constraints. A reduced gradient method with quasi-Newton updates is then used to solve the subproblem.
8. **NPSOL** [18] is an SQP algorithm using a line search and merit function for globalization. The merit function used is the augmented Lagrangian function. The Hessian is approximated with a dense BFGS update. The dense linear algebra methods make it more suitable to smaller problems.
9. **SNOPT** [19]: SNOPT is a large-scale SQP code. SNOPT implements sparse linear algebra routines so it is suitable for larger problems. A full-space, limited-memory

BFGS update is used for the Hessian matrix.

Chapter 3

Surrogate-based Optimization

In Chapter 2, we reviewed methods for nonlinear programming as well as some approaches to derivative free optimization. However, not all optimization problems of interest fit neatly into one of these problem classes. In particular, multi-disciplinary models are becoming more and more common. Computational models are increasingly complex, and frequently draw on knowledge from various domains, or utilize multi-scale modeling paradigms to model across length- and time-scales. As a result, a single optimization problem may need to model subsystems that are represented with various external software or simulations. In some circumstances, external models are computationally cheap with accurate derivative estimates available. In this case, these external functions may be provided to an optimization solver, for example through the ASL external function interface [20]. In the case that some part of the model cannot be addressed by this method, or accurate derivatives are not cheaply available, the problem is a hybrid "glass box/black box" optimization problem.

By glass box models, we refer to models that admit equation oriented representations. These models are easily implemented in an algebraic modeling language and derivatives may be efficiently obtained through automatic differentiation. This includes many common models in process engineering, where lumped parameter (algebraic) descriptions are widely accepted as sufficiently accurate. Black box models, by contrast, do not admit an equation oriented representation, or such a representation is unavailable. These models

are typically computationally expensive, so the performance of an optimization method for glass box/black box problems should be judged on its ability to find a solution with the fewest calls to the black box. We will assume that derivatives of the black box exist but are not available for use in the optimization algorithm. In practice, black box functions may have either stochastic noise or numerical noise (usually due to nested convergence loops solved to loose tolerances). Rigorous consideration of this noise is beyond the scope of this work, and we will assume in this study that noise is negligible or its effects can be avoided in the optimization.

3.1 Derivative Free Optimization (DFO) Algorithms

Before considering the solution of glass box/black box problems, we first review the basics of black box optimization, or derivative free optimization (DFO). DFO arose out of the demand for practical optimization algorithms that can be applied by directly querying an objective function. This mimics the trial-and-error that may be used to naively search for better solutions, but automates this process and (hopefully) comes with convergence theory to guide the search to an optimal solution. These can generally be divided into deterministic and stochastic approaches. The latter will not be considered in detail here, as these approaches often rely on extensive calls to the objective function, which may be costly to run, and convergence results are often lacking. Deterministic derivative free optimization is usually motivated by the optimization of a (possibly) expensive objective function.

Algorithms for deterministic derivative free optimization can be broken down into several categories. First, direct search methods seek better local solution through directly sampling the objective function. These can be further subdivided into Nelder-Mead type

algorithms and pattern search methods. The Nelder-Mead algorithm [21] was an early local search method and remains popular in several software packages. It is sometimes referred to as the Nelder-Mead simplex algorithm, not to be confused with the simplex method for nonlinear programming. It is so called because the algorithm maintains a set of $n+1$ affinely independent candidate solutions in n dimensional space. At each iteration, the point x^i with the worst objective value is replaced with a new point. This new point is generated according to some algorithmic rules, by expanding, contracting, reflecting etc. the simplex formed by the $n + 1$ candidate solutions. Therefore the set of candidate solutions slowly moves downhill to approach a local optimum. A modified version of the algorithm with convergence theory is presented in [22].

Pattern search methods and mesh adaptive direct search methods work with one incumbent point and seek a point with new point satisfying a sufficient decrease condition. Convergence is driven by polling positive spanning sets at a given step size. If the sampling fails to find a point that improves the objective function, then the step size is reduced. The various methods differ largely in their methods to choose a positive spanning set, handling of constraints, and heuristics to seek global solutions.

Several global search methods also operate deterministically. Lipschitzian partitioning techniques use an adaptively updated estimate of Lipschitz constants to propose "underestimators" of the objective function. Then, they proceed by sequentially partitioning the space minimizing the "underestimators," and updating the Lipschitz parameters. Termination is usually determined by a fixed function evaluation budget. Other deterministic methods still use probabilistic arguments: Assuming that samples of the objective function are realizations of a Gaussian process can lead to sampling strategies that maximize the "expected improvement" (for example, see [23]). Although derivations of these expressions can be involved, the initial assumption that the objective function may be modeled

as a Gaussian process may not always be appropriate.

3.1.1 Trust Region DFO

The class of methods that we will focus on is model-based trust region methods for DFO. In these methods, the objective function evaluations are used to construct interpolation/regression models, which then act as approximations for minimizing the original problem. This is usually combined with a trust region for globalization, where the trust region intuitively describes the region of the search space where the approximation is "trusted." The simplified outline algorithm is as follows, where the goal is to minimize $f(x)$.

1. Initialize algorithm, choose initial point x_0, trust region radius Δ_0, and algorithmic parameters $\eta, \gamma_c \in (0, 1)$ and $\gamma_e > 1$. Set $k = 0$.
2. Construct approximation model $r_k(x) \approx f(x)$.
3. Solve the trust region subproblem $\min_s r_k(x_k + s) \quad s.t.\ \|s\| \leq \Delta_k$ to obtain solution s_k.
4. Ratio test: Evaluate ρ_k = actual reduction / predicted reduction $= \frac{f(x_k)-f(x_k+s_k)}{r(x_k)-r(x_k+s_k)}$.
5. If $\rho_k < \eta$, then $x_{k+1} = x_k$ and set $\Delta_{k+1} = \gamma_c \Delta_k$. Set $k := k + 1$ and go to 2.
6. Else, set $x_{k+1} = x_k + s_k$ and $\Delta_{k+1} \in [\Delta_k, \gamma_e \Delta_k]$. Set $k := k + 1$ and go to 2.

The trust region is updated based on whether sufficient decrease was achieved. If the step was rejected, i.e. $x_{k+1} = x_k$, then the trust region radius is reduced by a contraction factor γ_c. Otherwise, it may be increased to try to take larger steps and speed progress.

The key to convergence is that the approximation models $r_k(x)$ are made to be sufficiently accurate within the trust region. The notion of sufficient accuracy may be quantified with the κ-fully linear property. First we denote the trust region as the ball of radius

Δ_k centered at x_k:

$$B(x_k, \Delta_k) := \{x : \|x - x_k\| \leq \Delta_k\}$$

This is defined as follows:

Definition 2 (κ-fully linear models)**.** *A model $r_k(w)$ is κ-fully linear approximation of f on $B(x_k, \Delta_k)$ if*

$$\|\nabla r_k(x) - \nabla f(x)\| \leq \kappa_g \Delta_k \qquad \textit{and} \qquad \|r_k(x) - f(x)\| \leq \kappa_f \Delta_k^2 \tag{3.1}$$

for all $x \in B(x_k, \Delta_k)$ and for some finite $\kappa_g > 0$ and $\kappa_f > 0$ independent of k

Roughly stated, this means that the approximation $r_k(x)$ must behave at least as well as a first order Taylor expansion within the trust region. Derivation or verification of this bound will depend on the particular form of approximation function $r(x)$ and the data used to fit it. In most implementations, the approximation is polynomial interpolation or regression, and the form of κ_f and κ_g can be derived as follows:

Consider the approximation of a black box function $\bar{d}(w) : \mathbb{R}^m \to \mathbb{R}$ using a polynomial in $\mathcal{P}_m^l$, the space of polynomials in $\mathbb{R}^m$ of degree l or less with real valued coefficients. We define $\phi(w) = \{\phi_1(w), \phi_2(w), \ldots, \phi_q(w)\}^T$ as a vector of basis functions for $\mathcal{P}_m^l$ evaluated at a particular point $w \in \mathbb{R}^m$. Denote the sample set $W = \{w^1, w^2, \ldots, w^p\} \subset \mathbb{R}^m$. Let $\bar{d}(W)$ denote the vector whose elements are the black box outputs $\bar{d}(w^i),\ i = 1, \ldots p$. The regression matrix M has entries $m_{i,j} = \phi_j(w^i)$. Thus, given a polynomial basis set ϕ and sample set W the least squares regression problem is stated as follows:

$$\min_{\alpha} \quad \|M\alpha - \bar{d}(W)\|^2 \tag{3.2}$$

If M has full column rank, we say that the sample set W is poised for least squares regression and the unique solution α^* is given by $\alpha^* = [M^T M]^{-1} M^T \bar{d}(W)$.

The error in the regression model can be bounded using the regression Lagrange polynomials $\ell_i,\ i = 1, \ldots p$. For a sample set W, each regression Lagrange polynomial $\ell_i \in \mathcal{P}_m^l$ is defined as $\bar{\alpha}_i^T \phi$, where $\bar{\alpha}_i$ is the solution of:

$$\min_{\alpha} \quad \|M\alpha - e_i\|^2 \tag{3.3}$$

where e_i is the i^{th} unit vector. Note that there is one Lagrange polynomial ℓ_i corresponding to each sample point w^i. Lemma 4.6 in [22] shows that the unique polynomial $r(w)$ that approximates $\bar{d}(w)$ in the least squares sense on the sample set W satisfies

$$r(w) = \sum_{i=1}^{p} \bar{d}(w^i)\ell_i(w) \tag{3.4}$$

The error in the approximation $r(w)$ may be bounded using the Lagrange polynomials. From Section 4.3 in [22] when $r(w)$ is a regression polynomial in $\mathcal{P}_m^l$,

$$|\bar{d}(w) - r(w)| \leq \frac{1}{(l+1)!}\nu_l \Lambda p \Delta^{l+1} \tag{3.5}$$

for any w in the smallest ball $B(W)$ containing sample set W. Δ denotes the diameter of this ball. Λ is defined as

$$\Lambda := \max_{1 \leq i \leq p} \max_{w \in B(W)} \ell_i(w). \tag{3.6}$$

ν_l is an upper bound on the $(l+1)^{th}$ derivative of $\bar{d}(w)$, and p is the sample size.

The functional form of (3.5) clearly gives the desired error bound for the κ-fully linear property. The only remaining effort is to show that the constants κ_f and κ_g are bounded, which is equivalent to showing that Λ is bounded for each iteration. Λ is strictly related to the geometry of the sample set, and can be estimated using the condition number of $M^T M$ [22]. Thus it is possible to verify, just through the choice of polynomial regression and the sample geometry, that the resulting approximation model will be κ-fully linear on $B(x_k, \Delta_k)$.

3.2 Constrained DFO

One approach for these hybrid glass box/black box systems is to simply apply derivative free optimization for the whole system. However, this strategy is restricted by inherent limitations of derivative free optimization methods. First, DFO methods have limited options in handling constraints. This is especially true when the constraints may be coupled with black box evaluations, as when a black box unit operation is embedded within a recycle loop of a chemical process. To apply DFO in the degrees of freedom, the recycle loop must be converged repeatedly. This requires many calls to the black box and therefore can be computationally expensive. Also, DFO methods do not scale well on problems with very many degrees of freedom. Model based trust region methods discussed in 3.1.1 usually use quadratic interpolation models to propose steps. Quadratic interpolation of a function $f : \mathbb{R}^n \to \mathbb{R}$ requires $\frac{(n+1)(n+2)}{2}$ poised sample points. Therefore, the expense of objective function evaluation can quickly become prohibitive.

A few derivative free optimization solvers have added features to consider some types of constrained problems. Powell's COBYLA [24] was the earliest available solver to explicitly handle general constraints in a DFO setting. Recently, more solvers can implement inequality constraints through a penalty or barrier approach (e.g. NOMAD [25, 26]). The recent trust funnel method [27] is able to handle equality constraints as well. For a recent review covering constrained derivative free optimization and applications, see [28]. Most methods are restricted in some way, for example only allowing linear constraints or only inequalities. Methods that handle general constraints may assume that all constraints are black box, when in fact glass box information may be known.

3.3 Surrogate-based Optimization

In the engineering literature, glass box/black box problems are often solved with the use of surrogate model (also known as reduced models, model functions, or meta-models). The black box function is replaced by an approximation that is compatible with the equation oriented optimization environment, resulting in a fully equation-oriented problem that can be solved with off-the-shelf solvers.

3.3.1 Model Reduction and Surrogate Modeling

Model reduction, in various forms, has played an integral role in chemical engineering process simulation and optimization. For the purpose of steady state simulation and design, most processes can be modeled with systems of algebraic equations. These equations can be solved with commercial process simulators or utilized in conjunction with state-of-the-art optimization methodologies. However, these models rely on several assumptions such as perfect mixing, plug flow, or thermodynamic equilibrium, which may restrict their domain of applicability. The choice of these assumptions represents an early form of model reduction. Recently, advances in detailed, distributed parameter, and multi-scale models have led to interest in new model reduction methods that aim to directly approximate the behavior of the original detailed model (ODM) for use in simulation and optimization. The detailed models often follow a "bottom up" modeling approach, where a set of (partial) differential and algebraic equations are derived from fundamental physical laws. Models at this higher fidelity can include transport behavior and detailed reaction kinetics, which require computationally costly simulations.

As stated in a recent perspective article [29], many challenges in energy and the environment require solutions using multi-scale analysis, design, simulation, and optimization.

Models developed at small length and time scales provide crucial insights for overall system performance. For example, the performance of entrained gasifiers is inherently tied to the transport and reaction behavior. A computational fluid dynamics (CFD) simulation of the reactor can take over 20 CPU hours to solve [30]. When considering that this gasifier is only one component in a pre-combustion carbon capture power plant, most design and optimization methodologies quickly become impractically time-consuming. Moreover, there are also software compatibility issues arising from the simultaneous use of distributed parameter CFD models for the gasifier but lumped parameter (algebraic) models for other equipment, including heat exchangers, turbines etc. These two reasons, the computational expense of detailed models and model compatibility across length- and time-scales, have motivated significant interest in surrogate modeling in process systems engineering.

There has always been a balance between model fidelity and computational tractability since the earliest use of computers in chemical engineering. Early computer-based flash calculations were greatly simplified by the introduction of surrogate models for the physical properties [31, 32, 33]. The use of surrogate models proved very effective for speeding the calculation without sacrificing much accuracy. Computing hardware has improved substantially since that time, but these early works show how model reduction can be used to solve problems that otherwise may be intractable.

Model reduction is a broad topic with contributions from many different research communities. Within chemical engineering, it is useful to classify strategies into two categories: model-based order reduction and data-driven model reduction. Model-based order reduction attempts to maintain the original structure while reducing the model size, while data-driven reduction views the model as an input-output relation that can be approximated by a surrogate model.

In model-based order reduction, the original structure of the model is somehow main-

tained. When the original model is a large-scale system of ODE's specialized methods can take advantage of that structure [34]. For nonlinear systems, model order reduction can include multi-grid methods, where discretization schemes may be adapted, and spectral methods, such as proper orthogonal decomposition. A survey of applications for chemical processes can be found in [35].

Proper Orthogonal Decomposition (POD), also known as Karhunen-Loéve decomposition, can reduce large spatially distributed models to much smaller models. POD models are formulated by projecting the PDAE system onto a set of basis functions, which are themselves generated from the numerical solution of the original equations. Applications are numerous, with examples including [36, 37, 38, 39, 40, 41, 42, 43, 44, 45]. POD models are built by using a Galerkin projection of the model onto a set of basis functions. These basis functions are often generated empirically from numerical solutions of the ODM, as through the *method of snapshots*. Therefore, despite classifying this as a model-based reduction method, there is still some dependency upon data and experimental design.

In addition to model-based reduction methods, data-based reduction methods have been successfully applied on many problems. The most common aspect of this approach is surrogate modeling, where the ODM is treated as a fully black box function. This black box may be sampled, and regression/ interpolation approaches can be used to fit the sampled data. The resulting surrogate model, or metamodel, is then used in place of the ODM for simulation, optimization, or other analysis. There is considerable flexibility in the functional form and fitting methods used for surrogate construction, and this flexibility can be used to customize an approach suitable for a particular problem. Simpson et al. [46] provide a seminal review of the field, which outlines several important steps and existing surrogate modeling frameworks. The main steps of surrogate modeling construction include experimental design, model selection, and model fitting. Several established methodolo-

gies suggest combinations of choices for each of these three steps. For example, response surface methodology, typically used in optimization settings uses central composite designs in combination with quadratic models constructed with least-squares regression. The central composite design helps determine curvature information for the quadratic models. A more complete description can be found in Myers and Montgomery [47]. In some ways, response surface methodology is a predecessor to the trust region methods for derivative free optimization. Other surrogate modeling approaches include Gaussian process regression (including Kriging) and artificial neural networks. These methods may perform better with space filling or sequential experimental designs [48, 49].

Surrogate modeling strategies remain an active area of research. Two noteworthy reviews including more recent developments can be found in [50, 51]. Contemporary works often borrow concepts from machine learning, such as the desire to fit "sparse" models, i.e. models with simpler functional forms. Best subset techniques with integer programming can be used to select appropriate basis functions from a larger set to arrive at sparse models [52, 53].

3.3.2 Algorithms with Convergence Theory

Even if considerable effort is used to construct a surrogate model, optimization using a surrogate model can lead to inaccurate answers. Any error in the surrogate model can be used to artificially improve the objective, and hence optimization may terminate in regions of poor surrogate accuracy. The surrogate model can then be adjusted to increase accuracy in that region. However, such a method offers no guarantee of converging to the optimum of the actual glass box/black box optimization problem, as shown by [54, 55]. In these works, the authors give an example where a surrogate model update algorithm even converges to a local maximum rather than minimum. The reason for this failure is

that the update scheme only adjusts for feasibility of the overall system, not optimality. If the surrogate model and black box function agree at a point, then the glass box/black box optimization problem is feasible. However, optimality conditions involve gradient information so a higher degree of accuracy is needed. In addition, globalization is required to drive convergence to a local minimum.

Several researchers have explored the use of trust region methods to use surrogate models effectively in optimization. Alexandrov et al. [56] applied the trust region concept to general surrogate models in engineering. They considered the task of unconstrained minimization using arbitrary approximation functions in place of the actual function. Surrogate models are constructed so that the function value and gradient of the surrogate model match those of the black box model at the center of the trust region. The trust region subproblem was the minimization of the surrogate model subject to the trust region constraint. Standard methods for unconstrained trust region methods were used to evaluate the step, including the ratio test shown in Section 3.1.1. Convergence was proved to the optimum of the original unconstrained black box problem. These concepts were extended to constrained optimization problems and implemented for the DAKOTA package [57], but convergence behavior was not proved. Another early work by Arian, Fahl, and Sachs [58] demonstrated the use of POD reduced models with a trust region method. The algorithm was shown to be convergent under the assumption that the gradients of the POD model are sufficiently accurate, although that accuracy condition may be difficult to verify in practice. Wild, Regis, and Shoemaker use the framework of κ-fully linear models to develop an algorithm for the use of radial basis function surrogate models [59]. March and Wilcox [60] also use the κ-fully linear framework from DFO for the use of multi-fidelity models. In multi-fidelity optimization, the reduced model is typically a coarser discretization of the PDE system. Caballero and Grossmann [61] use Kriging models to represent unit op-

erations for process optimization. Trust region concepts are used to shrink the domain of the Kriging models but convergence to local optima is not proved. Henao and Maravelias [62, 63] use a similar methodology with neural network representations of common process units, March and Wilcox [60] use constrained trust region subproblems with surrogate models. Globalization is managed with the use of a merit function, but the authors stop short of a convergence proof. Agarwal and Biegler [64] discuss convergence results for constrained subproblems when the derivatives of the black box are known. To ensure consistency and ultimately drive convergence to correct optimum, corrective terms are added to a POD approximation model. Biegler et al. [55] solve inequality constrained problems where the derivatives are unavailable using penalty function method combined with DFO concepts. Stopping criteria based on the reduced model errors are suggested.

Surrogate models have been used in global optimization / global search using "grey box" models [65, 66, 67]. In this broader class of problems, simplified models stand in for challenging or computationally expensive modeling elements. Asymptotic behavior is ignored because of tight budgets on function calls and dimensionality is small.

3.4 DFO Software

For a thorough comparison of DFO solvers, see [68]. This section mentions a few approaches that were influential in this dissertation.

1. **DFO** [69] A model-based trust region algorithm using quadratic interpolation models of the objective function. Sample sets are monitored for poisedness and adapted as necessary.
2. **ORBIT** [59] Optimization by radial basis function interpolation in trust regions uses a similar framework as the DFO algorithm but uses radial basis function approxima-

tions instead of polynomials.

3. **UOBYQA** [70] Unconstrained Optimization BY Quadratic Approximation is Powell's first software for derivative free optimization. Similar to the approach in the DFO software, Powell's approach uses a slightly different trust region update strategy as well as the concept of a sampling region to allow larger steps near the solution.
4. **NEWUOA** [71] The New Unconstrained Optimization Algorithm, developed to largely replace UOBYQA uses minimum Frobenius norm updates to a quadratic model at each iteration. By not requiring a full set of interpolation points, the algorithm can make progress with fewer function calls and numerical performance is promising, but rigorous convergence theory has not been developed.
5. **NOMAD** [26] Nonlinear Optimization by Mesh ADaptive search contains many configurable features to solve methods with direct search type algorithms. Inequality constraints can be handled through the use of barrier methods.

Chapter 4

Algorithm and Theory

In this chapter, we propose trust region filter (TRF) method for general glass box/black box optimization. The TRF method combines features of an SQP filter method [72] with model-based derivative free optimization concepts introduced in [22]. Whereas filter SQP solves a nonlinear program through a sequence of quadratic programming subproblems, the TRF method solves the glass box/black box nonlinear program (4.2) through a sequence of nonlinear programming subproblems. The motivation for this approach is that the glass box portion of the model provides cheap derivative information that should be exploited as much as possible, while function calls to the black box should be minimized. If a black box function is given directly to a gradient based NLP solver using finite difference gradients, the noisy derivative estimates can cause the solver to fail. Even if the method works, traditional NLP solvers are built with the assumption that function and gradient calls are reasonably cheap. However in a glass box/black box problem we may be able to use many calls to the cheap glass box derivatives to reduce the calls to the black box. The TRF method is motivated by this observation.

A trust region filter framework is used to control errors in the surrogate model while balancing improvement in feasibility and optimality. The algorithm is based on the SQP filter method in Fletcher et al. [72], its extension to inexact Jacobians by Walther and Biegler [73], and first order consistent surrogate models by [64]. By moving from quadratic programming subproblems to surrogate model based NLPs, more glass box information may be

utilized at each iteration. In the following section, the glass box/black box problem is formulated and the concepts of the surrogate model trust region filter method are introduced. Then, the trust region filter method for surrogate models is presented in Section 4.2. Then, the global convergence proof is discussed in Section 4.3. Finally, Section 4.4 concludes the paper and outlines areas for future work.

4.1 Algorithmic Components

Consider the following formulation of the glass box/black box optimization problem:

$$\begin{aligned} \min \quad & f(z, w, d(w)) \\ \text{s.t.} \quad & h(z, w, d(w)) = 0 \\ & g(z, w, d(w)) \leq 0 \end{aligned} \tag{4.1}$$

where minimization is over $z \in \mathbb{R}^n$ and $w \in \mathbb{R}^m$. w represents the inputs to the black box function, and z represents the remaining decision variables. $d(w) : \mathbb{R}^m \to \mathbb{R}^p$ represents the outputs of the black box as a function of inputs w. We assume that all functions f, h, g, d are twice continuously differentiable, although derivatives may be unavailable for $d(w)$. Because of the inherent difficulty of black box optimization problems, we assume that the black box model forms a small portion of the overall system. The number of black box inputs m is assumed to be small (less than a hundred). However, this assumes that the black box function is evaluated as a single function call. If $d(w)$ is in fact made up of several smaller simulation calls (for example multiple black box unit operations, or black box property calls for each stream), the dimension of w may be able to increase accordingly as long as the dimensionality of each individual simulation call is small. By contrast, the number of glass box variables n could be very large (on the order of tens of thousands), corresponding with the capabilities of conventional derivative-based nonlinear program-

ming methods.

To isolate the black box and glass box information, problem (4.1) is reformulated as follows:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) \leq 0 \\ & y = d(w) \end{aligned} \tag{4.2}$$

By introducing new variables $y \in \mathbb{R}^p$ and defining $x^T = [w^T, y^T, z^T]$, all black box information is moved to a single set of constraints. The remaining constraints $h(x) = 0$, $g(x) \leq 0$, and the objective function $f(x)$ can be calculated without calling the black box. To solve this problem, we will develop an algorithm that generates a sequence of points x_k converging to the solution of (4.2). This sequence will maintain feasibility for glass box constraints (i.e. $h(x_k) = 0$, $g(x_k) \leq 0$ for all k), while simultaneously converging towards optimality of (4.2) and feasibility of black box constraints $y = d(w)$. The aim is to accomplish this with as few calls to the black box $d(w)$ as possible.

At each iteration, an optimization problem using a surrogate model to propose a new point x_{k+1}, which may or may not be accepted. Below, we discuss the details of the trust region subproblem (TRSP) and the conditions we require on the surrogate models. Next, we discuss the *compatibility check*, which is solved before the TRSP to ensure that it is sufficiently feasible in a certain sense. Then, we discuss the *criticality check* that is used to help determine when the algorithm can terminate. After this, we discuss the filter mechanism that decides how to apply the solution of the TRSP in order to proceed to the next iteration $k + 1$. Finally, we present the TRF algorithm and also discuss the feasibility restoration procedure that is called when the compatibility check fails.

We will use equation based surrogate models to approximate the black box model. In

particular, the TRF method utilizes a sequence of surrogate models $r_k(w)$ approximating black box $d(w)$. Each surrogate model is built to be accurate in a trust region of radius Δ_k around a point x_k. By substituting the surrogate model in place of the black box and restricting the optimization to stay within the trust region where the surrogate model is accurate, we form the trust region subproblem for iteration k ($TRSP_k$) as follows:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) \leq 0 \\ & y = r_k(w) \\ & \|x - x_k\| \leq \Delta_k. \end{aligned} \tag{4.3}$$

In order to ultimately guarantee convergence of the subproblem solutions to the solution of (4.2), the surrogate models $r_k(w)$ are constructed to satisfy the κ-fully linear property within the trust region. This condition, introduced in Section 3.1.1, essentially requires that the gradients and function values of $r_k(w)$ and $d(w)$ converge as the trust region radius $\Delta_k \to 0$. It is an inherent property of the method used to construct the surrogate model, and under mild assumptions on sample set geometry, many interpolation and regression methods can satisfy the κ-fully linear condition [22, 74]. This agnostic approach to the functional form of the surrogate model provides great flexibility in customizing a surrogate model to a specific black box $d(w)$. If some knowledge is available from the application as to what functional form to expect, this can be used directly to improve optimization performance.

Note that the trust region constraint is written for all variables x rather than just black box inputs w. If the glass box variables were not bounded with the trust region, i.e. if the trust region constraint only bounds $\|w - w_k\| \leq \Delta_k$, then subproblem $TRSP_k$ may become unbounded, even if the original problem (4.2) has a unique minimizer. Moreover,

the convergence proof assumes that the solution to the trust region subproblem converges to x_k as the trust region converges to zero. This form of the trust region constraint helps enforce that assumption.

4.1.1 Compatibility and Criticality Check

After constructing a κ-fully linear surrogate model, we could attempt to solve the trust region subproblem (4.3). However, after replacing $d(w)$ with $r_k(w)$ and introducing a trust region constraint $\|x - x_k\| \leq \Delta_k$, the subproblem (4.3) might not be feasible. If the feasible region of $TRSP_k$ is close to the trust region center, then the subproblem is likely to be a good approximation of (4.2) because it has room to explore the feasible set and improve the objective function. In this case we say that $TRSP_k$ is compatible.

Definition 3 (Compatibility)**.** *If there exists a point $\bar{x}$ feasible for $TRSP_k$ (4.3) such that, for fixed parameters $\kappa_\Delta \in (0, 1)$, $\kappa_\mu > 0$, and $\mu \in (0, 1)$,*

$$\|\bar{x} - x_k\| \leq \kappa_\Delta \Delta_k \min[1, \kappa_\mu \Delta_k^\mu] \tag{4.4}$$

then $TRSP_k$ is compatible.

The following optimization problem, called the compatibility check, is used to measure the feasibility of the subproblem:

$$\begin{aligned} \min_x \quad & \|y - r_k(w)\| \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) \leq 0 \\ & \|x - x_k\| \leq \kappa_\Delta \Delta_k \min[1, \kappa_\mu \Delta_k^\mu] \end{aligned} \tag{4.5}$$

Problem (4.5) is always feasible (at $x = x_k$; recall that $h(x_k) = 0$ and $g(x_k) \leq 0$ for all k). The objective function $\|y - r_k(w)\|$ searches for a feasible point to (4.3), while the trust

region constraint is a requirement for compatibility. At each iteration define

$$n_k = x^*_{n,k} - x_k$$

where $x^*_{n,k}$ is a minimizer of (4.5). n_k may be partitioned into $n_{w,k}$, $n_{y,k}$, and $n_{z,k}$, just as x is partitioned into w, y, and z variables. If

$$y_k + n_{y,k} - r_k(w_k + n_{w,k}) = 0, \tag{4.6}$$

then $TRSP_k$ passes the compatibility test. When $TRSP_k$ is not compatible, a restoration procedure is called to generate a compatible trust region subproblem $TRSP_{k+1}$ (see Section 4.1.3).

If $TRSP_k$ passes the compatibility check, then we apply a criticality test. The criticality test uses a criticality measure $\chi(x)$, which approaches zero when x is near a KKT point of (4.3) when the trust region constraint is omitted. We interpret a small value of $\chi(x)$ as an indication that x may be near a solution of (4.2), so the trust region radius should be reduced to increase the surrogate model accuracy via the κ-fully linear property (3.1). By eventually shrinking Δ_k to zero, the algorithm is able to certify optimality in the limit.

The criticality measure $\chi(x)$ is derived by linearizing the subproblem (4.3) at one of its feasible points. However, the original trust region constraint is dropped and replaced with a unit trust region, as shown in (4.7):

$$\begin{aligned} \chi(x) \quad = \quad & \left| \min_{v} \quad \nabla f(x)^T v \right| \\ \text{s.t.} \quad & \nabla h(x)^T v = 0 \\ & g(x) + \nabla g(x)^T v \leq 0 \\ & v_y - \nabla r_k(w)^T v_w = 0 \\ & \|v\| \leq 1 \end{aligned} \tag{4.7}$$

where the partitioning of $v^T = [v_w^T, v_y^T, v_z^T]$ corresponds to that of x. If a polyhedral norm

(e.g. $\|\cdot\|_1$ or $\|\cdot\|_\infty$) is used in the final constraint of (4.7), this problem is a linear program and is relatively easy to solve.

In Algorithm 1, criticality is always checked at $x_k + n_k$. For convenience, define $\chi_k := \chi(x_k + n_k)$. If the following condition holds:

$$\chi_k < \xi \Delta_k \tag{4.8}$$

where $\xi > 0$ is a fixed parameter, then $TRSP_k$ requires a criticality update. Condition (4.8) implies that we are close to optimality relative to the trust region radius and the trust region radius is decreased in order to refine the accuracy of the surrogate model. This criticality test directly corresponds to the criticality phase in Chapter 10 of [22].

If $TRSP_k$ is compatible and (4.8) fails, then the trust region subproblem (4.3) is solved. Note that this problem will be feasible since we can initialize at $x_k + n_k$ and $TRSP_k$ is compatible.

4.1.2 Filter

Define

$$s_k := x^*_{s,k} - x_k \tag{4.9}$$

where $x^*_{s,k}$ is a minimizer of (4.3). After solving $TRSP_k$ to obtain s_k, we evaluate the quality of this step. If s_k makes sufficient progress towards feasibility and/or optimality of (4.2), then the step is successful and we assign $x_{k+1} = x_k + s_k$. Otherwise, the step is unsuccessful and $x_{k+1} = x_k$.

To define the notion of sufficient progress, our method will use the concept of a filter originally developed in [10]. As an alternative to the merit function [75], a filter borrows concepts from multi-objective optimization to balance the trade-off between feasibility and

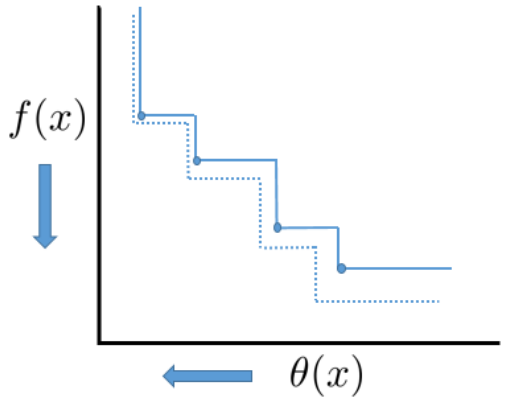


Figure 4.1: Convergence of the filter

the objective function. We define the following infeasibility measure of NLP (4.2):

$$\theta(x) = \|y - d(w)\|.$$

Because all trust region subproblems maintain feasibility for glass box constraints $h(x) = 0$ and $g(x) \leq 0$, the infeasibility of black box constraints comprises the entire infeasibility of the NLP.

The algorithm will define a subset of iterates $\mathcal{Z} \subset \mathbb{N}$ for which (θ, f) pairs are stored in the filter set, defined as follows:

$$\mathcal{F}_k = \{(\theta_j, f_j) : j < k, j \in \mathcal{Z}\}$$

where $\theta_j := \theta(x_j)$ and $f_j := f(x_j)$. When we say that (θ_k, f_k) is added to the filter, then we mean that k is assigned to $\mathcal{Z}$. These filter points can be interpreted as building a Pareto front for the minimization of θ and f.

This is illustrated in Figure 4.1. If a point lies sufficiently below or to the left of the filter front, it is acceptable to the filter; i.e. if for all $(\theta_j, f_j) \in \mathcal{F}_k \cup (\theta_k, f_k)$,

$$\theta(x_k + s_k) \leq (1 - \gamma_\theta)\theta_j \quad \text{or} \quad f(x_k + s_k) \leq f_j - \gamma_f \theta_j \tag{4.10}$$

then the step s_k is acceptable to the filter, where $\gamma_\theta, \gamma_f \in (0, 1)$ are fixed tuning parameters.

For small θ, the reduction in f must be prioritized. To see this, note that if every iterate is added to the filter, the algorithm may converge to a feasible but suboptimal solution. For example, suppose that x_0 is in fact optimal for (4.2). Then, due to inaccuracy in the surrogate model used in the subproblem (4.3), we may take a step that improves the objective function, but is actually infeasible for (4.2). However, because the (θ, f) pair for x_0 was added to the filter, there is no way to return to this optimal point. To prevent this behavior, we evaluate the following switching condition:

$$f(x_k) - f(x_k + s_k) \geq \kappa_\theta \theta(x_k)^{\gamma_s} \tag{4.11}$$

where κ_θ and γ_s are tuning parameters. If (4.11) is true, we say that the step s_k is an "f-type step" and (θ_k, f_k) should not be added to the filter. Otherwise, we call k a "θ-type step" and assign k to set $\mathcal{Z}$. In addition to θ-type steps, the filter is augmented whenever restoration is called, i.e. (θ_k, f_k) is added to the filter when $TRSP_k$ is not compatible.

Note that for f-type steps we do not need to check for sufficient decrease in the objective function in (4.11) as required in the filter method in [72]. Instead of a quadratic model, the subproblem (4.3) uses the actual objective function, so sufficient decrease will always be achieved. Hence the usual sufficient decrease check for f-type steps is unnecessary and every step s_k that is acceptable to the filter is successful, i.e. $x_{k+1} = x_k + s_k$.

The filter also determines how the trust region radius is updated. If a step s_k is unacceptable to the filter, then the trust region radius is decreased in order to refine the accuracy of surrogate model $r_k(w)$. If s_k is acceptable and k is an f-type step, then the error in the surrogate model is under control and the trust region radius may be increased to take bigger steps. For θ-type steps, the convergence theory in [72] allows total freedom as long as Δ_k remains bounded. We use the following update rule, based on the ratio test in derivative

free trust region methods [22]. First, calculate the ratio of reduction in θ.

$$\rho_k = \frac{\theta(x_k) - \theta(x_k + s_k)}{\|y_k - r_k(w_k)\|} = \frac{\theta(x_k) - \theta(x_k + s_k)}{\theta^r(x_k) - \theta^r(x_k + s_k)} \tag{4.12}$$

where $\theta^r(x) = \|y - r_k(w)\|$. This is the standard ratio test from trust region methods, i.e. the actual reduction divided by the predicted reduction. Note that $\theta^r(x_k + s_k) = 0$ since the subproblem is compatible. If the surrogate model interpolates at the trust region center, i.e. $r_k(w_k) = d(w_k)$, then (4.12) simplifies as follows:

$$\rho_k = 1 - \frac{\theta(x_k + s_k)}{\theta(x_k)}$$

The trust region is then updated as follows, using parameters $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \gamma_c < 1 < \gamma_e$.

$$\Delta_{k+1} = \begin{cases} \gamma_c \Delta_k & \text{if} \quad \rho_k < \eta_1, \\ \Delta_k & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \\ \gamma_e \Delta_k & \text{if} \quad \rho_k \geq \eta_2. \end{cases} \tag{4.13}$$

Note that because $x_k + s_k$ is acceptable to the filter, the step s_k is accepted for any value of ρ_k.

4.1.3 Restoration

As mentioned in Section 4.1.1, restoration is called whenever $TRSP_k$ is not compatible. A restoration procedure is any algorithm that, given a point x_k, returns x_{k+1}, Δ_{k+1}, and $r_{k+1}(w)$ such that x_{k+1} is acceptable to the filter and $TRSP_{k+1}$ is compatible.

The restoration phase seeks a point that is acceptable to the filter (4.10). A sufficient condition for a successful restoration algorithm is to return a point x_{k+1} that is (nearly) feasible for (4.2) along with a surrogate model $r_{k+1}(w)$ that interpolates the trust region center (i.e. $r_{k+1}(w_{k+1}) = d(w_{k+1})$). In this case, x_{k+1} is acceptable to the filter because fea-

sible points are always acceptable. In addition, one can check that $TRSP_{k+1}$ is compatible for any choice of $\Delta_{k+1} > 0$ by taking x_{k+1} as the solution of (4.5).

Finding a feasible point can be formulated as minimization of $\theta(x)$ subject to the glass box constraints, but this is also a glass box/black box optimization problem. Instead, application of problem-specific simulation techniques may be able to find a feasible solution using calls to $d(w)$ directly. For chemical process optimization, sequential modular concepts can be used to converge the flowsheet using $d(w)$. The inputs w to the black box may be used as a tear stream. A fixed point iteration may be run by first evaluating the black box $d(w)$ and then solving the rest of the flowsheet to obtain a new value of w. Though performance is not guaranteed, we found that using this method for restoration resulted in fewer calls to the black box than Newton-type methods for process optimization problems.

4.2 Algorithm 1: Trust Region Filter (TRF) Method

In this section, the trust region filter algorithm is given for solving hybrid glass box/black box problems. The detailed algorithm is presented below, and the overall logic is sketched in Figure 4.2. Note that termination tolerances are not included; for convergence analysis, the sequence of iterates must extend to infinity. A modified algorithm including termination conditions is presented in the following chapter.

Algorithm 1:

1. Initialization: Choose $0 < \gamma_c < 1 < \gamma_e$, $\gamma_f \in (0,1)$, $\gamma_\theta \in (0,1)$, $\kappa_\theta \in (0,1)$, $\kappa_\Delta \in (0,1)$, $\kappa_\mu > 0$, $\mu \in (0,1)$, $\gamma_s > \frac{1}{1+\mu}$, $\xi > 0$, $\omega \in (0,1)$, $0 < \eta_1 \leq \eta_2 < 1$, $\kappa_{tmd} \in (0,1]$, an initial trust-region radius Δ_0, and an initial iterate $x_0 \geq 0$. Initialize the filter $\mathcal{F}_0 = \emptyset$. Evaluate $d(x_0)$ and then calculate $\theta(x_0)$. Set $k = 0$.
2. Generate a surrogate model $r_k(x)$ that is κ-fully linear on Δ_k.

3. Solve the compatibility check (4.5). If $TRSP_k$ is compatible as given by Definition 3, go to Step 4. Otherwise, add (θ_k, f_k) to the filter and go to Step 10.
4. Compute criticality measure χ_k. If criticality check (4.8) holds, reassign $\Delta_k := \omega\Delta_k$ and go to Step 2. Else, continue to Step 5.
5. Solve the subproblem (4.3) to compute a step s_k.
6. *Filter:*

   Evaluate $\theta(x_k + s_k)$. If the step is acceptable to the filter as given in (4.10), continue to Step 7. Else, set $x_{k+1} = x_k$, $\theta_{k+1} = \theta_k$, and $\Delta_{k+1} = \gamma_c\Delta_k$, then set $k := k + 1$ and go to Step 2.
7. *Switching condition:*

   If the switching condition (4.11) is true, go to Step 8. Else, go to Step 9.
8. *f-type step*

   Accept trial step and increase trust region: Set $x_{k+1} = x_k + s_k$, $\Delta_{k+1} = \gamma_e\Delta_k$, and $\theta_{k+1} = \theta(x_k + s_k)$. Set $k := k + 1$ and go to Step 2.
9. *θ-type step*

   Add (θ_k, f_k) to the filter and accept trial step $x_{k+1} = x_k + s_k$. Update the trust region using (4.12) and (4.13). Set $k := k + 1$ and go to Step 2.
10. *Restoration*

    Call restoration algorithm to compute a point x_{k+1}, trust region radius Δ_{k+1}, and surrogate model r_{k+1} such that x_{k+1} is acceptable to the filter $\mathcal{F}_k \cup (\theta_k, f_k)$ and $TRSP_{k+1}$ is compatible. Set $k := k + 1$ and go to Step 2.

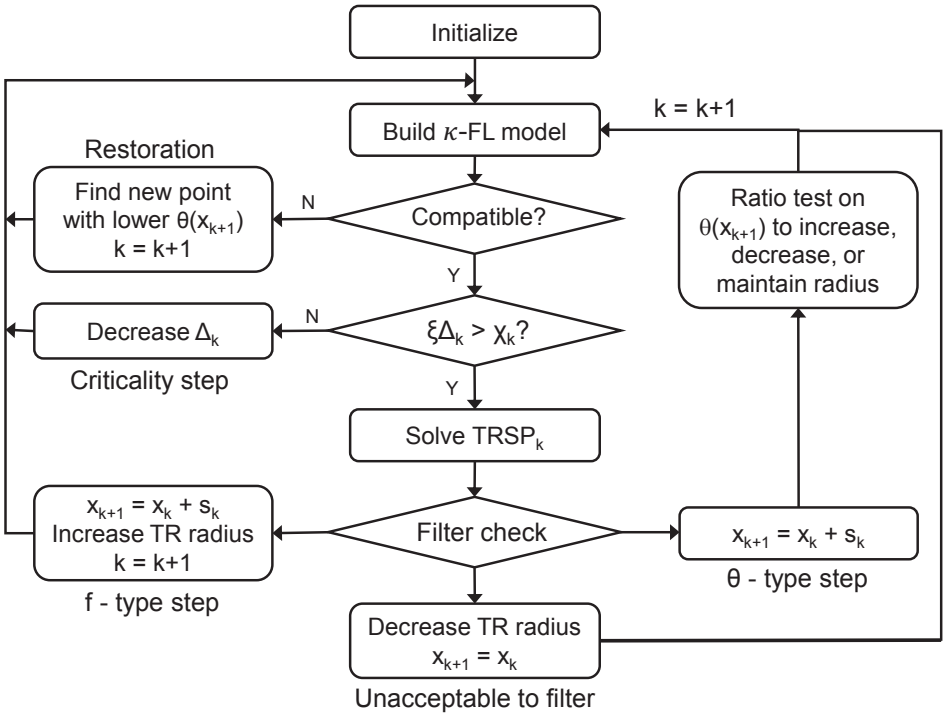


Figure 4.2: Algorithm 1 Flowchart

4.3 Global Convergence

Under mild assumptions, the sequence x_k generated by Algorithm 1 will converge to a first order KKT point of (4.2). Standard NLP assumptions regarding smoothness of the underlying functions and regularity of limit points are required. Also, assumptions are made regarding the solutions of trust region subproblems (e.g. the solver should return a local solution that sufficiently improves the feasible initial point). Under these assumptions, the proof of Fletcher et al. [72] may be used with slight modification to show that there exists a subsequence k_j for which $x_{k_j} \to x^*$, where x^* is a first order KKT point of (4.2). The proof first shows that θ_{k_j}, χ_{k_j}, and Δ_{k_j} all limit to zero. This in turn implies that x^* is a KKT point. Details of the convergence proof are given as follows.

To begin, we present a modified set of assumptions to accommodate NLP subproblems. Then, we present two modified lemmas. The remainder of the proof then follows directly from Fletcher et al. with little modification.

4.3.1 Assumptions

(A1) All functions are twice-continuously differentiable

(A2) All iterates $\{x_k\}$ remain in a closed bounded domain $\Omega \subset \mathbb{R}^{m+n+p}$.

These two assumptions are fairly standard in the nonlinear optimization literature; they guarantee that the objective function is bounded below and therefore a minimum must exist. The second derivatives are also bounded, which provide smoothness guarantees for the proof. Assumption (A2) may be enforced by including bounds on all constraints in the glass box inequalities $g(x) \leq 0$, because the algorithm will never violate these constraints. Assumption (A1) could be relaxed to hold for the union of all trust regions, but we keep the assumption for all x for simplicity of presentation and to match with the assumption

from Fletcher et al.

(A3) Approximations $r_k(w)$ are κ-fully linear for each iteration k, and are twice continuously differentiable with uniformly bounded second derivatives.

The requirement of κ-fully linear models is given in Definition 2 in Section 3.1.1. We further assume that whatever method is used to build surrogate models will give sufficiently smooth models. Since the black box $d(w)$, which is approximated by surrogate model $r_k(w)$, is also assumed to be smooth, this assumption is relatively mild.

(A4) The Mangasarian Fromovitz Constraint Qualification (MFCQ) holds for the constraint set of (4.2) at all limit points of $\{x_k\}$.

MFCQ is a regularity condition on the constraint set so that the KKT necessary optimality conditions apply (see Chapter 2). This helps draw the connection between χ in (4.7) and optimality of (4.2). In addition, MFCQ implies that $\chi(x)$ is a continuous function of x (See Theorem 2.2.1 in [76] combined with Theorem 2.3 in [77]), which will help us show that a sequence x_k for which $\chi_k \to 0$ in fact converges to an optimal point of (4.2). To illustrate the relationship between MFCQ and continuity of χ, consider the following example: $\min x \quad \text{s.t.} \quad 0 \leq y \leq x^3$. At the origin, MFCQ is violated because there does not exist a direction into the strict interior of the linearized feasible region. One can easily check that $\lim_{x \to 0^+} \chi(x) = 0$, but $\chi(0) = 1$. Hence violation of MFCQ leads to discontinuity in the optimal value function.

The remaining two assumptions state the requirements on the subproblem solutions.

(A5) The solution s_k to the trust region subproblem (4.3) satisfies the fraction of Cauchy decrease condition, as follows:

$$f(x_k + n_k) - f(x_k + s_k) \geq \kappa_{tmd}\, \chi_k \min\left[\frac{\chi_k}{\beta_k}, \Delta_k\right] \tag{4.14}$$

for some constant κ_{tmd} and bounded sequence $\beta_k > 1$.

Whatever method we use to solve the subproblem is assumed to reduce the objective function by some quantity related to the criticality measure. Because NLP (4.3) is initialized at $x_k + n_k$, and the criticality measure is evaluated at this point, the sufficient decrease is considered relative to that point as well. For detailed motivation and discussion of the right hand side of (4.14), see Chapter 15 of Conn et al. [78].

(A6) For some δ_n and κ_{usc} independent of k, if $\theta_k \leq \delta_n$, there exists a solution to the compatibility check (4.5) such that (4.6) holds and

$$\|n_k\| \leq \kappa_{usc}\theta_k. \tag{4.15}$$

Moreover, it is assumed that whatever method is used to solve (4.5) will find such a solution.

The existence of an n_k satisfying (4.15) functions as regularity assumption on the subproblem $TRSP_k$. To illustrate the relationship between n_k and θ_k, consider a surrogate model $r_k(w)$ that interpolates at the trust region center x_k (i.e., $r_k(w_k) = d(w_k)$, a condition that we typically enforce in the case studies). Therefore, the objective function value of (4.5) evaluated at x_k gives $\|y_k - r_k(w_k)\| = \theta_k$. (A6) only has to hold for small θ_k, so (A6) states that it does not require an arbitrarily large step in the set $\{x : h(x) = 0,\, g(x) \leq 0\}$. The assumption that a solver will find such a solution is likewise mild because we only require (4.15) when $\theta(x_k)$ is small.

4.3.2 Convergence Proof

Fletcher et al. [72] give a global convergence proof for a filter SQP method, which we will adapt to prove convergence of Algorithm 1. The full set of lemmas are included below, but proofs are omitted when they follow directly from a proof in [72].

First, the notion of finite termination is modified. In Fletcher et al., finite termination refers to the case when a point x_k is found such that $\chi_k = \theta_k = 0$. In our work, the trust region also has to go to zero. We define finite termination as the case where the sequence $\{x_k\}$ generated by Algorithm 1 is finite. This can only result when the algorithm loops infinitely in between Steps 2 and 4. If this happens for some iterate k, then x_k is a first order critical point if MFCQ holds at x_k, which can be shown using a similar argument to that in the proof of Lemma 4.3.2.

Lemma 4.3.1. (Bound 2 on normal step) *Assume Algorithm 1 is applied and finite termination does not occur. If (A2) and (A6) hold, $k \notin \mathcal{R}$, and $\theta_k \leq \delta_n$, then there exists a constant $\kappa_{lsc} > 0$ independent of k such that*

$$\kappa_{lsc}\theta_k \leq \|n_k\| \tag{4.16}$$

Proof. Follows from Lemma 3.1 in Fletcher et al. □

Lemma 4.3.2. (Replaces Lemma 3.2 in Fletcher et al.) *Assume that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose that (A1), (A2), (A3), (A4), and (A6) hold, and that there exists a subsequence $\{k_i\}$ such that $TRSP_{k_i}$ is compatible for all k_i, and*

$$\lim_{i\to\infty} \chi_{k_i} = 0, \quad \lim_{i\to\infty} \theta_{k_i} = 0, \quad \textit{and} \lim_{i\to\infty} \Delta_{k_i} = 0. \tag{4.17}$$

Then every limit point of the subsequence $\{x_{k_i}\}$ is a first order critical point for problem (4.2).

Proof. The existence of a limit point x^* is given by (A2). Consider the following criticality measure, which will allow us to draw the relationship between a stationary point of problem (4.2) and observed criticality χ_k (4.7). The argument δ will be used to represent the

error in the gradient of the surrogate model.

$$\begin{aligned} \hat{\chi}(x,\delta) \quad = \quad & \left|\min_{v} \quad \nabla f(x)^T v\right| \\ \text{s.t.} \quad & \nabla h(x)^T v = 0 \\ & g(x) + \nabla g(x)^T v \leq 0 \\ & v_y - (\nabla d(w) + \delta)^T v_w = 0 \\ & \|v\|_1 \leq 1 \end{aligned} \tag{4.18}$$

Note that if $\hat{\chi}(x^*, 0) = \theta(x^*) = 0$, then x^* is a first order critical point of (4.2).

Theorem 2.2.1 in [76] combined with Theorem 2.3 in [77] provides the following result. If MFCQ holds at a point (x, δ), then the optimal value function $\hat{\chi}(x, \delta)$ of the convex program (4.18) is continuous at that point. Therefore, with assumption (A4), we conclude that $\hat{\chi}(x, \delta)$ is continuous at $(x^*, 0)$.

Define the sequence

$$\delta_k := \nabla r_k(x_k + n_k) - \nabla d(x_k + n_k)$$

such that

$$\chi_k = \chi(x_k + n_k) = \hat{\chi}(x_k + n_k, \delta_k).$$

By assumption (A6), $\theta_{k_i} \to 0$ implies that $\|n_{k_i}\| \to 0$. Since $TRSP_k$ is compatible, $x_{k_i} + n_{k_i}$ lies within the trust region and we know from the fully linear property (A3):

$$\|\delta_{k_i}\| \leq \kappa_g \Delta_{k_i}$$

Since $\Delta_{k_i} \to 0$, we know that

$$\lim_{i \to \infty} \delta_{k_i} = 0$$

Finally, this allows us to state that

$$\hat{\chi}(x^*, 0) = \lim_{i \to \infty} \hat{\chi}(x_{k_i} + n_{k_i}, \delta_{k_i}) = \lim_{i \to \infty} \chi_{k_i} = 0.$$

Note that all x_{k_i} satisfy the glass box constraints $h(x) = 0$ and $g(x) \leq 0$, so x^* is also glass box feasible since the feasible set is closed. Together with $\theta(x^*) = 0$ (by continuity), we conclude x^* is a first order critical point of (4.2). □

Lemma 4.3.3. (Limit of θ_k for filter iterates) *Assume that Algorithm 1 is applied to NLP (4.2) and that finite termination does not occur. Suppose that (A1) and (A2) hold and that $|\mathcal{Z}| = \infty$. Then:*

$$\lim_{\substack{k \to \infty \\ k \in \mathcal{Z}}} \theta_k = 0.$$

Proof. Same as Lemma 3.3 in Fletcher et al. The proof only relies on the definition of the filter mechanism, which has not changed in this algorithm. □

Lemma 4.3.4. (Lemma 3.4 Fletcher et al.) *Assume that Algorithm 1 is applied to problem (4.2) and finite termination does not occur. Suppose that (A1), (A2), and (A3) hold, and that $TRSP_k$ is compatible. Then there exists a constant $\kappa_{ubt} > 0$ such that*

$$\theta_k \leq \kappa_{ubt} \Delta_k^{1+\mu} \tag{4.19}$$

and

$$\theta(x_k + s_k) \leq \kappa_{ubt} \Delta_k^2. \tag{4.20}$$

Proof. For the first conclusion, Lemma 3.4 in Fletcher et al. is applied directly to show that there exists some $\hat{\kappa}_{ubt}$ such that

$$\theta_k \leq \hat{\kappa}_{ubt} \Delta_k^{1+\mu}. \tag{4.21}$$

The second conclusion follows directly from the fully linear property (3.1) as follows: Denote $c_k(x) = y - r_k(w)$. Since $TRSP_k$ is compatible, we have $c_k(x_k + s_k) = 0$. Therefore $\theta(x_k + s_k) \leq \kappa_f \Delta_k^2$. Then, we may choose κ_{ubt} as the maximum of κ_f from the fully linear property and the constant $\hat{\kappa}_{ubt}$ from relation (4.21). □

Lemma 4.3.5. (Lower bound on objective improvement) *Assume that Algorithm 1 is applied to NLP (4.2) and finite termination does not occur. Suppose also that (A1), (A2), (A5), and (A6) hold and that $k \notin \mathcal{R}$. Suppose, for some $\epsilon > 0$,*

$$\chi_k \geq \epsilon$$

and

$$\Delta_k \leq \min\left[\left(\frac{\epsilon}{\kappa_{umh}}\right), \left(\frac{2\kappa_{ubg}}{\kappa_{umh}\kappa_\Delta\kappa_\mu}\right)^{\frac{1}{1+\mu}}, \left(\frac{\kappa_{tmd}\epsilon}{4\kappa_{ubg}\kappa_\Delta\kappa_\mu}\right)^{\frac{1}{\mu}}\right] =: \delta_m \tag{4.22}$$

where $\kappa_{ubg} = \max_{x\in\Omega} \|\nabla f(x)\|$ and $\kappa_{umh} = \max\left[\max_k\{\beta_k\}, \max_{x\in\Omega} \|\nabla^2 f(x)\|\right]$. Then,

$$f(x_k) - f(x_k + s_k) \geq \frac{1}{2}\kappa_{tmd}\epsilon\Delta_k. \tag{4.23}$$

Proof. Follows from Lemma 3.5 in Fletcher et al. The assumption of bounded Hessians (assumption AS2 in Fletcher et al.) is not required since the actual objective function is used, and this result follows from (A1) and (A2). □

Lemma 4.3.6. (f-type steps for small Δ) *Suppose that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose also that (A1), (A2), (A3), (A5), and (A6) hold, that $k \notin \mathcal{R}$, that (4.16) holds, $\chi_k \geq \epsilon$, and that*

$$\Delta_k \leq \min\left[\delta_m, \left(\frac{\kappa_{tmd}\epsilon}{2\kappa_\theta\kappa_{ubt}^{\gamma_s}}\right)^{\frac{1}{\gamma_s(1+\mu)-1}}\right] =: \delta_f. \tag{4.24}$$

Then

$$f(x_k) - f(x_k + s_k) \geq \kappa_\theta\theta_k^{\gamma_s}.$$

Proof. Same as Lemma 3.7 Fletcher et al. Follows from Lemmas 4.3.4 and 4.3.5. □

Lemma 4.3.7. (Small trust region will guarantee filter acceptance) *Suppose that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose also that (A1), (A2),*

(A3), (A5), and FCD condition holds, $\chi_k \geq \epsilon$, $k \notin \mathcal{R}$, *and*

$$\theta_k \leq \kappa_{ubt}^{-\frac{1}{\mu}} \left(\frac{\kappa_{tmd}\epsilon}{2\gamma_\theta} \right)^{\frac{1+\mu}{\mu}} =: \delta_\theta. \tag{4.25}$$

Then

$$f(x_k + s_k) \leq f(x_k) - \gamma_\theta \theta_k.$$

Proof. Follows easily from Lemma 3.8 in Fletcher et al. where η_2 is omitted because $m(x) = f(x)$. Derived from Lemmas 4.3.4 and 4.3.5. □

Lemma 4.3.8. (Small TR and small θ will not call restoration) *Suppose that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose also that (A1), (A2), (A3), (A5), (A6),* $\chi_k \geq \epsilon$, *and that*

$$\Delta_k \leq \min \left[\left(\frac{1}{\kappa_\mu} \right)^{\frac{1}{\mu}}, \left(\frac{\gamma_c^2 (1-\gamma_\theta)\kappa_\Delta \kappa_\mu}{\kappa_{usc}\kappa_{ubt}} \right)^{\frac{1}{1-\mu}} \right] =: \delta_{\mathcal{R}} \tag{4.26}$$

Suppose furthermore than $k > 0$ *and that*

$$\theta_k \leq \min[\delta_\theta, \delta_n] \tag{4.27}$$

Then, $k \notin \mathcal{R}$.

Proof. By equation (4.27) and the stated assumptions, we know that the conclusions of Lemmas 4.3.1 and 4.3.7 apply. Assume, for the purposes of contradiction, that $k \in \mathcal{R}$. Because we assume (A6) holds, this means that

$$\|n_k\| > \kappa_\Delta \kappa_\mu \Delta_k^{1+\mu} \tag{4.28}$$

where we have used the fact that $\kappa_\mu \Delta_k^\mu$ is less than 1 because of (4.26). The construction of restoration phase must guarantee that if $k \in \mathcal{R}$ then $k - 1 \notin \mathcal{R}$. This can easily be accomplished if restoration phase returns a feasible point.

Now first suppose that iteration $k-1$ is unsuccessful, i.e., $x_{k-1} = x_k$. Then, because $\theta_k = \theta_{k-1}$ and Lemma 4.3.7 holds, we know that

$$f(x_{k-1} + s_{k-1}) \leq f(x_{k-1}) - \gamma_\theta \theta_{k-1}. \tag{4.29}$$

It is a property of the filter that if (4.29) holds and also

$$\theta(x_{k-1} + s_{k-1}) \leq (1 - \gamma_\theta)\theta_{k-1}$$

then s_{k-1} is acceptable. So, since iteration $k-1$ is unsuccessful, we have

$$\theta(x_{k-1} + s_{k-1}) > (1 - \gamma_\theta)\theta_k \tag{4.30}$$

Now, applying Lemma 4.3.4 (which in turn uses Lemma 4.3.1),

$$(1 - \gamma_\theta)\theta_k < \theta(x_{k-1} + s_{k-1}) \leq \kappa_{ubt}\Delta_{k-1}^2 \leq \frac{\kappa_{ubt}}{\gamma_c^2}\Delta_k^2$$

Now, using our supposition (4.28) and normal step requirement (4.15),

$$\kappa_\Delta \kappa_\mu \Delta_k^{1+\mu} < \|n_k\| \leq \kappa_{usc}\theta_k \leq \frac{\kappa_{usc}\kappa_{ubt}}{\gamma_c^2(1 - \gamma_\theta)}\Delta_k^2$$

and hence,

$$\Delta_k^{1-\mu} > \frac{\gamma_c^2(1 - \gamma_\theta)\kappa_\Delta \kappa_\mu}{\kappa_{usc}\kappa_{ubt}}$$

which contradicts (4.26), so the assumption that $k-1$ is unsuccessful must be false.

Thus, iteration $k-1$ is successful and $\theta_k = \theta(x_{k-1} + s_{k-1})$. By (4.28), (4.15), and Lemma 4.3.4,

$$\kappa_\Delta \kappa_\mu \Delta_k^{1+\mu} < \|n_k\| \leq \kappa_{usc}\theta_k \leq \kappa_{usc}\kappa_{ubt}\Delta_{k-1}^2 \leq \frac{\kappa_{usc}\kappa_{ubt}}{\gamma_c^2}\Delta_k^2$$

implying that

$$\Delta_k^{1-\mu} > \frac{\gamma_c^2 \kappa_\Delta \kappa_\mu}{\kappa_{usc}\kappa_{ubt}}$$

again contradicting (4.26) since $(1 - \gamma_\theta) \in (0, 1)$. Therefore, the initial supposition (4.28) must be false thus completing the proof. □

Lemma 4.3.9. (Convergence for infinite filter) *Suppose that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose also that (A1), (A2), (A3), (A5), and (A6) hold. Suppose furthermore that $|\mathcal{Z}| = \infty$. Then there exists a subsequence $\{k_j\} \subseteq \mathcal{Z}$ such that*

$$\lim_{j\to\infty} \theta_{k_j} = 0 \tag{4.31}$$

$$\lim_{j\to\infty} \chi_{k_j} = 0 \tag{4.32}$$

$$\lim_{j\to\infty} \Delta_{k_j} = 0 \tag{4.33}$$

Proof. The first conclusion, (4.31), is simply a restatement of Lemma 4.3.3. Next, Fletcher et al. (Lemma 3.10) show that for any infinite subsequence $\{k_i\} \subseteq \mathcal{Z}$, a contradiction with (4.31) implies that there exists $\{k_l\} \subseteq \{k_i\}$ such that

$$\lim_{l\to\infty} \Delta_{k_l} = 0.$$

Next, Lemmas 4.3.6 and 4.3.8 imply that we will have successful f-type steps as long as χ_{k_l} is bounded away from zero, which prevent the trust region from vanishing. Therefore, $\chi_{k_l} \to 0$. □

Lemma 4.3.10. ($\theta \to 0$ for finite filter) *Suppose that Algorithm 1 is applied to problem (4.2), that finite termination does not occur, and that $|\mathcal{Z}| < \infty$. Suppose also that (A1), (A2), (A3), (A5), and (A6) hold. Then,*

$$\lim_{k\to\infty} \theta_k = 0. \tag{4.34}$$

Proof. Because $|\mathcal{Z}| < \infty$, we note that $|\mathcal{R}| < \infty$. Denote $k_0 > 0$ as the last iterate for which x_{k_0-1} is added to the filter. We know that all successful iterates with $k \geq k_0$ are f-type steps. If the set of successful iterates is finite, then $\Delta_k \to 0$ and Lemma 4.3.7 implies that

finite termination has occurred. We know that all successful iterates with $k \geq k_0$ are f-type steps and for successful iterate k

$$f(x_k) - f(x_{k+1}) \geq \kappa_\theta \theta(x_k)^{\gamma_s} \geq 0$$

Because (A1)-(A2) imply that f is bounded below and the objective function cannot increase for $k \geq k_0$, we have that

$$\lim_{k \to \infty} f(x_k) - f(x_{k+1}) = 0 \tag{4.35}$$

so considering the previous relation, the desired conclusion (4.34) follows. □

Lemma 4.3.11. (for finite filter, Δ_k bounded away from zero if χ bounded away from zero) *Suppose that Algorithm 1 is applied to problem (4.2) that finite termination does not occur and that $|\mathcal{Z}| < \infty$. Suppose also that (A1), (A2), (A3), (A5), and (A6) apply. If $\chi_k \geq \epsilon$ for all $k \geq k_0$ (where k_0 is defined as in the previous lemma), then there exists a $\Delta_{min} > 0$ such that*

$$\Delta_k \geq \Delta_{min}$$

for all k.

Proof. Follows from Fletcher et al. Lemma 3.12.

□

Lemma 4.3.12. (Convergence of χ) *Suppose that Algorithm 1 is applied to problem (4.2) that finite termination does not occur and that $|\mathcal{Z}| < \infty$. Suppose also that (A1), (A2), (A3), (A5), and (A6) apply. Then,*

$$\liminf_{k \to \infty} \chi_k = 0. \tag{4.36}$$

Proof. Lemma 4.3.10 gives $\theta_k \to 0$, which implies that $\|n_k\| \to 0$ by (A6). The convergence of f in successful iterates (4.35) implies in turn that

$$\lim_{\substack{k\to\infty \\ k\in\mathcal{S}}} [f(x_k + n_k) - f(x_k + s_k)] = 0 \tag{4.37}$$

If $\chi_k \geq \epsilon$ for all $k \geq k_0$, Lemma 4.3.11 and the definition of the FCD condition imply for all $k \geq k_0$,

$$f(x_k + n_k) - f(x_k + s_k) \geq \kappa_{tmd}\epsilon \min\left[\frac{\epsilon}{\kappa_{umh}}, \Delta_{min}\right]$$

immediately giving a contradiction with (4.37). Hence, χ_k is not bounded away from zero and the desired conclusion follows. □

The convergence result is summarized in the following theorem.

Theorem 4.3.13. *Suppose that Algorithm 1 is applied to problem (4.2) and that finite termination does not occur. Suppose also that (A1), (A2), (A3), (A4), (A5) and (A6) hold. Then either the restoration procedure terminates unsuccessfully or there is a subsequence $\{k_j\}$ for which*

$$\lim_{j\to\infty} x_{k_j} = x_*$$

where x_* is a first order KKT point of problem (4.2).

Proof. Suppose that the restoration procedure always terminates successfully. Then, from Lemma 4.3.9, Lemma 4.3.10, and Lemma 4.3.12 we have.

$$\lim_{j\to\infty} \theta_{k_j} = \lim_{j\to\infty} \chi_{k_j} = 0.$$

Step 4 in the algorithm gives

$$\lim_{j\to\infty} \Delta_{k_j} = 0$$

and finally, Lemma 4.3.2 from above gives the final conclusion. □

4.4 Conclusions

A surrogate model trust region algorithm is presented for glass box/black box optimization. Surrogate models are frequently used to embed black box functions into glass box models, but care must be taken to correctly optimize these systems. Trust region methods provide a natural framework for managing error in surrogate models. In addition, the concept of a filter helps measure the success of a surrogate model in achieving feasibility and optimality. The convergence of our method if KKT points is proved by combining properties from the trust region filter developed in [72, 73] with derivative free optimization concepts [22]. The applicability to chemical process optimization is shown on the Williams-Otto process, the ammonia process, and a large-scale power plant optimization problem in Chapter 6.

In the future, we plan to improve performance as we gain experience on more problems. When multiple black box functions are included in one problem, it may be beneficial to use a separate trust region to manage each surrogate model. This may involve some algorithmic modifications to the filter concepts. In addition, the handling of noise is an open problem. The ability to rigorously account for stochastic or numerical noise would greatly expand potential areas of application.

Chapter 5

Implementation and Improvements

A trust region filter (TRF) method specifically for solving glass box/black box problems was presented in Chapter 4 along with discussion of convergence properties. The version of the algorithm presented in Section 4.2 can serve as a "vanilla" version, which is easiest to follow for the purpose of convergence. However, several algorithmic adjustments can improve performance while still working within the convergence theory presented. In this chapter, we discuss these improvements to the TRF method as well as its implementation in an integrated optimization framework. This chapter discusses and analyzes several of the more important developments including separate trust and sampling regions as well as their updating rules. To benchmark the improvements and implementation, a test set of problems is gathered and the effects of major algorithmic modifications are demonstrated.

The structure of this chapter is as follows, we first discuss our algorithmic improvements in Section 5.1 and the modified algorithm is given in Section 5.2. Then, we present a Pyomo-based implementation of our algorithm in Section 5.3. and discuss the performance of the algorithm on a set of test problems in Section 5.4. Finally, we conclude in Section 5.5.

5.1 Proposed Modifications

In gaining experience with initial implementations of Algorithm 4.2, several concerns emerged. First, the algorithm tends to take very small steps s_k when moderately close

to an optimum. This occurred because the filter required θ to remain small, which in turn requires accurate surrogate models. A small trust region keeps the surrogate models very accurate, but also slows the progress because all variables are restricted by the trust region.

This means that there is conflict between the trust region's role in globalization, i.e. in guaranteeing overall convergence, and its role in surrogate model management. To address this problem, we will introduce the concept of a sampling region. This method can partially decouple the two tasks of globalization and surrogate model management.

In addition, several issues arise when introducing numerical tolerances. Several algorithmic adjustments to consider tolerances are discussed, and then a modified version of the TRF algorithm is presented including stopping conditions.

5.1.1 Sampling Region

In trust region-based derivative free methods, the trust region simultaneously serves two purposes: surrogate model error management and step size control (globalization). For algorithms addressing fully black box problems, e.g. unconstrained DFO, this may not be particularly troublesome. On the other hand, Powell separated these two tasks through the concept of a sampling region in the unconstrained DFO solver UOBYQA [70]. The sampling region describes the domain where the surrogate model is certifiably accurate (e.g. where most data is contained for interpolation models), while a trust region is used for globalization. The method works well, but when compared to other trust region-based DFO algorithms that do not make a distinction between sampling region and trust region, the advantage is not clear. However, for glass box / black box problems, the trust region includes all variables, not just the inputs to the black box. A small trust region gives a more accurate surrogate model, but greatly restricts the progress in the variables z that do not directly interact with the black box. The sampling region offers a solution to this problem.

We define the sampling region for iteration k as the ball $B(x_k, \sigma_k)$, where σ_k represents the sampling region radius. We assume that the sampling region is contained within the trust region, so $\sigma_k \leq \Delta_k$ for all iterations k. This assumption is based on our context of polynomial surrogate models, and a larger sampling region may make sense for other types of surrogate models or when the black box is extremely expensive. By building models that are κ-fully linear property only for $B(x_k, \sigma_k)$, the models can more accurately capture local behavior. Note that while we require the surrogate model to be κ-fully linear on the sampling region, this does not necessarily require that all samples need to be contained within this region.

The benefit of the sampling region is two-fold. The original algorithm requires the trust region to vanish as the algorithm converges; now we only require that the sampling region goes to zero. This will allow for larger trust region steps near the solution, which is particularly important since the glass box variables z are also controlled by the trust region for globalization purposes. Second, with the sampling region radius $\sigma_k < \Delta_k$, it may not be necessary to rebuild the surrogate model if the step is rejected. Therefore, the trust region can adjust for globalization purposes, and unless the new trust region radius is set below the current sampling region radius, the original surrogate model may be re-used.

A potential drawback of the sampling region is that is decreases the chance that sample points can be reused if the trust region center moves. However, the potential to reuse points tends to become smaller as the number of black box inputs increases. Consider the volume of the intersection of two trust regions. For example, suppose that the trust region at iteration k is the set $\{x \in \mathbb{R}^n : \|x\|_\infty \leq 1\}$, where the current iterate $x_k = 0$. Then, the subproblem proposes a full step to the corner of the trust region, i.e. $x_{k+1} = [1, 1, \ldots, 1]^T$. Further suppose that the trust region radius stays the same, $\Delta_{k+1} = \Delta_k = 1$. Then, the volume of each trust region is 2^n but the volume of the intersection between

the two iterations is only 1. This demonstrates the dimension-dependency of the potential to reuse sample points between two consecutive iterations. As the dimension increases, the "probability" that previous sample points will lie near the new trust region center decreases exponentially.

The sampling region concept has not been shown before to be compatible with the convergence framework of κ-fully linear models. Conn et al. [22] suggest that the κ-fully linear property can hold on a region of radius $\alpha\Delta_k$, where α is a fixed positive constant. However, this will always require the trust region to decrease in criticality phase. Instead, we will develop an algorithm that only requires the sampling region radius to tend to zero, while the trust region can remain larger near optimality.

5.1.1.1 κ-fully linear on $B(w_k, \sigma_k)$ implies κ-fully linear on $B(w_k, \Delta_k)$

Recall the definition of a κ-fully linear model: A model $r(w)$ approximating $d(w)$ is κ-fully linear on $B(w_k, \sigma)$ for constants κ_f and κ_g if for all w in $B(w_k, \sigma)$:

$$\|d(w) - r(w)\| \leq \kappa_f \sigma^2 \qquad \|\nabla d(w) - \nabla r(w)\| \leq \kappa_g \sigma \tag{5.1}$$

We can show that a surrogate model $r_k(w)$ that is κ-fully linear on $B(w_k, \sigma)$ will be κ-fully linear on $B(w_k, \Delta)$ with $\Delta > \sigma$. Standard assumptions are used, $d(w)$ and $r_k(w)$ are both twice differentiable on $B(w_k, \Delta)$, and the second derivatives are uniformly bounded for all k.

Given any $w \in B(w_k, \Delta)$, Taylor's theorem states that:

$$d(w) = d(w_k) + \nabla d(w_k)^T (w - w_k) + \mathcal{O}(\|w - w_k\|^2) \tag{5.2}$$

$$r(w) = r(w_k) + \nabla r(w_k)^T (w - w_k) + \mathcal{O}(\|w - w_k\|^2) \tag{5.3}$$

Subtracting (5.3) from (5.2) and applying the Cauchy-Schwarz inequality:

$$\|d(w) - r(w)\| \leq \|d(w_k) - r(w_k)\| + \|\nabla d(w_k) - \nabla r(w_k)\| \|w - w_k\| + L_r \|w - w_k\|^2 \tag{5.4}$$

where L_r is a bounding constant. Then using (5.1) for point w_k and recalling that $w \in B(w_k, \Delta)$,

$$\|d(w) - r(w)\| \leq \kappa_f \sigma^2 + \kappa_g \sigma \Delta + L_r \Delta^2 \tag{5.5}$$

Since $\sigma < \Delta$,

$$\|d(w) - r(w)\| \leq (\kappa_f + \kappa_g + L_r)\Delta^2 \tag{5.6}$$

so κ-fully linearity on $B(w_k, \Delta)$ follows from boundedness of L_r. A similar argument holds for the bound on the gradients, outlined as follows. Again given any $w \in B(w_k, \Delta)$, we have:

$$\nabla d(w) = \nabla d(w_k) + \mathcal{O}(w - w_k) \tag{5.7}$$

$$\nabla r(w) = \nabla r(w_k) + \mathcal{O}(w - w_k) \tag{5.8}$$

Subtracting (5.8) from (5.7) and introducing bounding constant L_{rg}:

$$\|\nabla d(w) - \nabla r(w)\| \leq \|\nabla d(w_k) - \nabla r(w_k)\| + L_{rg}\|w - w_k\| \tag{5.9}$$

Now applying the Cauchy-Schwarz inequality, then applying (5.1):

$$\|\nabla d(w) - \nabla r(w)\| \leq \kappa_g \sigma + L_{rg}\Delta \tag{5.10}$$

$$\|\nabla d(w) - \nabla r(w)\| \leq (\kappa_g + L_{rg})\Delta \tag{5.11}$$

thus proving the following Lemma.

Lemma 5.1.1. *If a model $r_k(w)$ is κ-fully linear on $B(w_k, \sigma_k)$ for constants κ_f, κ_g, and the second derivatives of $r_k(w)$ are uniformly bounded on $B(w_k, \Delta_k)$, then there exist $\hat{\kappa}_f, \hat{\kappa}_g$ such that $r_k(w)$ is κ-fully linear on $B(w_k, \Delta_k)$.*

Above, we showed that $\hat{\kappa}_f = \kappa_f + \kappa_g + L_r$ and $\hat{\kappa}_g = \kappa_g + L_{rg}$.

5.1.1.2 Update of sampling region radius

As with the use of a sampling region in UOBYQA, we avoid shrinking the sampling region until necessary. If the trust region radius would be set to some value less than the sampling radius, then the sampling region is shrunk so that it is again contained within the trust region. In addition, the criticality phase of the original algorithm no longer has to take the trust region to zero. Instead, because of the κ-fully linear property, first order optimality is guaranteed as the sampling region radius decreases to zero.

The original criticality test, given in (4.8), shrinks the trust region by a constant factor each time it is satisfied. When using a sampling region, the criticality phase shrinks the sampling region instead of the trust region. In addition, computational experience showed that reducing the sampling region by a constant factor may be too conservative. Often once the criticality phase is invoked, the algorithm has nearly found the solution, allowing a more aggressive reduction in σ to be used. Therefore with the sampling region, the criticality test (4.8) changes to

$$\chi_k < \xi\sigma_k \tag{5.12}$$

If this condition is true, then σ_k is reduced. The following formula then uses the observed criticality to attempt to set σ to a value where the criticality test (5.12) no longer holds.

$$\sigma_k = \max(\min(\sigma_{k-1}, \chi_k/\xi), \Delta_{min})$$

5.1.2 Step Size Update

The basic trust region update formula (4.13) increases or decreases the trust region radius by constant factors. However, as noted in [78], this may be too restrictive of a requirement. In general, the size of the trust region at iteration $k+1$ is based on the norm of the step computed at iteration k. Since the trust region has been largely decoupled from surrogate

model accuracy, this allows the trust region to rapidly adapt to its role in globalization.

As shown in the convergence proof (see Section 4.3), the following two features are required for the trust region radius. First, the trust region must not decrease when taking an f-type step. Next, the trust region must decrease when a step is rejected. Outside of this, we have freedom to define an update procedure as desired, using computational experiments as a guide. The following scheme has been effective in practice.

One problem that emerged in Algorithm 1 was a sequence of f-type steps with small norm $\|s_k\|$. This allowed the trust region to grow very large during convergence, in turn causing a prolonged criticality phase to certify optimality. For f-type steps, the trust region is now updated as follows:

$$\Delta_{k+1} := \max\left[\gamma_e \|s_k\|, \Delta_k\right]$$

where $\gamma_e \geq 1$ and s_k is the step as defined in (4.9). This means that if the current trust region constraint is (nearly) active, the trust region should be expanded to allow faster progress. However, if the step at this iteration is small, we maintain the current value of Δ_k since a decrease in the radius is not allowed.

For θ-type steps, the convergence proof has no restrictions on the trust region update (provided $\Delta_k \in \mathbb{R}^+$). Therefore, we propose the following modification.

$$\Delta_{k+1} = \begin{cases} \gamma_c \|s_k\| & \text{if} \quad \rho_k < \eta_1, \\ \Delta_k & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \\ \max\{\gamma_e \|s_k\|, \Delta_k\} & \text{if} \quad \rho_k \geq \eta_2 \end{cases} \tag{5.13}$$

where ρ_k is given by

$$\rho_k = \frac{\theta(x_k) - \theta(x_k + s_k) + \epsilon_\theta}{\max(\|y_k - r_k(w_k)\|, \epsilon_\theta)} \tag{5.14}$$

and ϵ_θ is a small tolerance. This new definition of ρ is derived from the standard trust region definition of actual reduction divided by predicted reduction (4.12). Since the trust

region subproblem is compatible, we know that the predicted reduction is equal to $\theta(w_k)$. The small tolerance ϵ_θ is added to control behavior near $\theta = 0$. The form of this definition was chosen to prevent the trust region from decreasing when both θ_k and $\theta(x_k + x_k)$ are very small. Since the trust region update on θ-type steps does not impact global convergence, this form is chosen for its better numerical performance.

When a step is rejected, the following update is used:

$$\Delta_{k+1} = \gamma_c \|s_k\|.$$

This aggressively shrinks the trust region to prevent the same failed step being tried again. In other words, this eliminates the possibility that $s_{k+1} = s_k$, which may occur when the surrogate model r_{k+1} is nearly or exactly the same as the one used at iteration k.

5.1.3 Changes to filter mechanism

In this section, we discuss additional modifications to the filter mechanism introduced in Section 4.1.2. We will say that a step s_k is acceptable to the filter if:

$$\theta(x_k + s_k) \leq (1 - \gamma_\theta)\theta_j \quad \text{or} \quad f(x_k + s_k) \leq f_j - \gamma_f \theta_j \tag{5.15}$$

for all $(\theta_j, f_j) \in \mathcal{F}_k$. In previous trust region filter methods [72], the following additional requirement is enforced for all acceptable steps:

$$\theta(x_k + s_k) \leq (1 - \gamma_\theta)\theta_k \quad \text{or} \quad f(x_k + s_k) \leq f(x_k) - \gamma_f \theta_k \tag{5.16}$$

By contrast, the line search filter method proposed in [12] only requires (5.16) for θ-type steps.

Because imposition of (5.16) is a tighter condition on the acceptance of s_k, it may lead to more frequent shrinkage of the trust region and calls to the restoration phase at the current

iterate. On the other hand, if (5.16) is not imposed, a bad s_k would be allowed that may impede performance at subsequent iterations.

We observed this trade-off in our computational results, and found that allowing s_k "inside" the filter (disregarding (5.16)) actually improved performance and reduced the number of calls to restoration phase. This means that in general, taking a relatively "bad" step that doesn't satisfy (5.16) seems better at preventing restoration than decreasing the trust region directly. Moreover, only condition (5.15) is required to guarantee convergence.

We also borrow the following requirement from IPOPT [12] that restricts f-type steps to small values of θ. This may be viewed as a modification of the switching condition (4.11). A step s_k is an f-type step if

$$\theta(x_k) \leq \theta_{min} \quad \text{and} \quad f(x_k) - f(x_k + s_k) \geq \kappa_\theta \theta(x_k)^{\gamma_s} \tag{5.17}$$

The requirement that θ be small to take an f-type step is justified as we do not want to increase the trust region radius for very large values of θ. This places more emphasis on reducing infeasibility in the earlier stages of the algorithm.

5.2 Algorithm 2: Modified TRF Method

The TRF method with the above modifications is presented below. Unlike the presentation in Chapter 4, termination tolerance parameters are also included here. For convenience, the algorithm from Section 4.2 is referred to as Algorithm 1, and the modified version presented here is referred to as Algorithm 2.

Algorithm 2:

1. Initialization: Choose an initial trust-region radius Δ_0, an initial iterate x_0, and an initial sampling radius σ_0. Choose termination tolerances ϵ_θ (for feasibility), ϵ_χ (for criticality), and $\epsilon_\Delta \geq \Delta_{min}$ for ensuring accurate surrogate models and preventing

numerical issues with small trust regions, respectively. Choose compatibility check parameters $\kappa_\Delta \in (0,1)$, $\kappa_\mu > 0$, $\mu \in (0,1)$, and $\epsilon_{compat} > 0$. Choose criticality scaling parameter $\xi > 0$. Select trust region update parameters $0 < \gamma_c < 1 \leq \gamma_e$. Choose switching condition parameters $\gamma_s > \frac{1}{1+\mu}$, $\kappa_\theta \in (0,1)$, and $\theta_{min} > 0$. Initialize the filter $\mathcal{F}_0 = \emptyset$ and select positive filter parameters γ_f, γ_θ, and θ_{max}. Select ratio test thresholds $0 < \eta_1 < \eta_2 < 1$ and sampling region reset parameter $\psi \in (0,1]$. Evaluate $d(x_0)$ and then calculate $\theta(x_0)$. Set $k = 0$.

2. Generate a surrogate model $r_k(w)$ that is κ-fully linear on $B(x_k, \sigma_k)$. Reuse previous surrogate model if possible.
3. *Criticality and termination check:*

   Calculate criticality measure χ_k (4.7)

   (a) If $\theta_k \leq \epsilon_\theta$, $\chi_k \leq \epsilon_\chi$, and $\sigma_k \leq \epsilon_\Delta$, STOP. A first order critical point has been found to tolerance.

   (b) If $\Delta_k \leq \Delta_{min}$, $\Delta_{k-1} \leq \Delta_{min}$, $\theta_k \leq \epsilon_\theta$, and $\theta_{k-1} \leq \epsilon_\theta$, STOP. A feasible solution has been found but progress to optimality is too slow.

   (c) *Criticality phase:* Set $\sigma_k = \max(\min(\sigma_{k-1}, \chi_k/\xi), \Delta_{min})$
4. *Compatibility Check:*

   Solve the compatibility check (4.5). Denote the optimal objective function value as β. If $\beta < \epsilon_{compat}$ (i.e. $TRSP_k$ is compatible), go to Step 5. Otherwise, add (θ_k, f_k) to the filter and go to Step 10.
5. Solve subproblem (4.3) and obtain the solution $\bar{x}$. Define the step $s_k := \bar{x} - x_k$.
6. *Filter:*

   Evaluate $\theta(x_k + s_k)$. If for all $(\theta_j, f_j) \in \mathcal{F}_k$,

$$\theta(x_k + s_k) \leq (1 - \gamma_\theta)\theta_j \quad \text{or} \quad f(x_k + s_k) \leq f_j - \gamma_f \theta_j$$

then the step s_k is acceptable to the filter. If the step is acceptable to the filter, continue to Step 7. Else, set $x_{k+1} = x_k$, $\theta_{k+1} = \theta_k$, $\Delta_{k+1} = \gamma_c \|s_k\|$, and $\sigma_{k+1} = \min\{\sigma_k, \psi\Delta_{k+1}\}$, then set $k := k + 1$ and go to Step 2.

7. *Switching condition:*

   If the switching condition (5.17) holds, then go to Step 8. Else, go to Step 9.

8. *f-type step*

   Accept trial step and possibly increase trust region: Set $x_{k+1} = x_k + s_k$, $\Delta_{k+1} = \max\{\gamma_e \|s_k\|, \Delta_k\}$, $\sigma_{k+1} = \sigma_k$, and $\theta_{k+1} = \theta(x_k + s_k)$. Set $k := k + 1$ and go to Step 2.

9. *θ-type step*

   Add (θ_k, f_k) to the filter and accept trial step $x_{k+1} = x_k + s_k$. Update the trust region using (5.13) and (5.14). Set $\sigma_{k+1} = \min[\sigma_k, \psi\Delta_{k+1}]$. Set $k := k + 1$ and go to Step 2.

10. *Restoration Phase* Return a new point x_{k+1}, trust region radius Δ_{k+1}, sample region radius σ_{k+1} and new surrogate model $r_{k+1}(w)$ such that $TRSP_{k+1}$ is compatible. Set $k := k + 1$ and go to step 2.

5.3 Implementation

The TRF method, as shown in Section 5.2 has been implemented in Python/Pyomo. Pyomo (Python Optimization Modeling Objects) is an open source framework for modeling and analyzing mathematical programming problems [79, 80]. The implementation within Python, a full-featured high-level programming language, provides great flexibility to researchers and practitioners compared to other modeling platforms. In addition, Pyomo includes interfaces to many of the most popular mathematical programming solvers. This is especially well suited to the TRF method, where models are repeatedly solved with small modifications, and a user has freedom to select a particular solver for the subprob-

lem. By default, all subproblems are solved with IPOPT 3.12.4 [12], although any other Pyomo-supported solver may be substituted by the user. The criticality check (4.7) requires the derivatives of the problem evaluated at a point; these are obtained using the AMPL pseudo-solver *gjh*.

5.3.1 Polynomial Interpolation Models

One of the advantages of the TRF method is the flexibility for using many types of surrogate models. If any particular structure or information is known, a user may define her own function to construct a surrogate model using that information. The TRF method assumes that the result will satisfy the κ-fully linear property.

Computational experience suggests polynomial surrogate models are most suitable for general problems. The current implementation of the TRF method includes methods for building linear and quadratic interpolation models. With linear interpolation models, we use a forward difference perturbation with magnitude equal to the sampling radius. If using quadratic interpolation, the method uses a fixed, a priori calculated sample geometry at every iteration. In typical derivative free codes, the geometry is allowed to adapt, reusing points from previous iterations when possible. The fixed geometry is a simplifying assumption, showing the resulting behavior with well-poised sample geometry at every iteration, and this could be treated as an upper bound to the computational expense of a more careful implementation. However, the use of a sampling region makes it highly unlikely that points will be available for re-use.

The fixed sample geometries are designed to interpolate the trust region center and points on a sphere of radius σ_k. For a black box with input dimension m, linear models require at least $m+1$ function evaluations per trust region iteration, whereas quadratic interpolation models require $(m+1)(m+2)/2$ function evaluations. Note however that

the TRF method is in no way restricted to these polynomial surrogate models. The sample geometry consists of $(m+1)(m+2)/2-1$ points on the unit sphere $\{w : \|w\| = 1, w \in \mathcal{R}^m\}$. Together with the origin, this forms the full interpolation set. At each iteration, these sample points are scaled multiplicatively by the sampling radius, then shifted to the sample region center w_k. The black box is evaluated at these points, then the interpolation problem can be uniquely solved.

This paragraph briefly reviews polynomial interpolation: Consider the approximation of a black box function $\bar{d}(w) : \mathbb{R}^m \to \mathbb{R}$ using a polynomial in $\mathcal{P}_m^l$, the space of polynomials in $\mathbb{R}^m$ of degree l or less with real valued coefficients. We define

$$\phi(w) = \{\phi_1(w), \phi_2(w), \ldots, \phi_q(w)\}^T$$

as a vector of basis functions for $\mathcal{P}_m^l$ evaluated at a particular point $w \in \mathbb{R}^m$. Denote the sample set $W = \{w^1, w^2, \ldots, w^p\} \subset \mathbb{R}^m$. Let $\bar{d}(W)$ denote the vector whose elements are the black box outputs $\bar{d}(w^i), \ i = 1, \ldots p$. The interpolation matrix M has entries $m_{i,j} = \phi_j(w^i)$. Thus, given a polynomial basis set ϕ and sample set W the interpolating polynomial may be found from solving the linear system

$$M\alpha = \bar{d}(W) \tag{5.18}$$

If M is square and nonsingular, we say that W is poised for polynomial interpolation.

The sample set W is always a fixed geometry in our implementation, so the matrix M for each dimension of quadratic polynomial is stored on disk and retrieved at the beginning of the TRF method. The interpolation problem is solved for the normalized sample set around the origin, then the polynomial coefficients are modified to account for the scaling and shifting to the sample region center.

5.3.2 Restoration Phase

The restoration phase is normally designed to minimize infeasibility measure θ. In the implementation for general problems, the restoration algorithm works as follows: The compatibility check (4.5) is solved repeatedly, using the classical trust region update formula (4.12) and (4.13) to update the trust region. This is repeated until either the compatibility check shows that the trust region subproblem is compatible or the trust region shrinks to a small tolerance, at which point the algorithm terminates at a local minimum of infeasibility.

5.3.3 Convergence Properties

All the modifications can be applied without modifying the core elements of the convergence proof. The only significant change is that now the sampling region radius σ_k is driven to zero instead of the trust region radius Δ_k. Lemma 4.3.2 is therefore can be rewritten

Lemma 5.3.1. *Assume that Algorithm 2 is applied to problem (4.2) and that finite termination does not occur. Suppose that (A1), (A2), (A3), (A4), and (A6) hold, and that there exists a subsequence $\{k_i\}$ such that $TRSP_{k_i}$ is compatible for all k_i, and*

$$\lim_{i\to\infty} \chi_{k_i} = 0, \quad \lim_{i\to\infty} \theta_{k_i} = 0, \quad \textit{and} \ \lim_{i\to\infty} \sigma_{k_i} = 0. \tag{5.19}$$

Then every limit point of the subsequence $\{x_{k_i}\}$ is a first order critical point for problem (4.2).

5.4 Numerical Results

In order to test the effectiveness of proposed modifications to the TRF algorithm, a set of test problems was assembled. There is no known standard library of test problems for

glass box/black box optimization, so a test set has been developed from modified versions of established NLP benchmarks from the CUTEr and COPS sets [81, 82].

22 problems from the CUTEr set were modified through the replacement of one or several nonlinear constraints with a black box. The black box constraint or expression was chosen to be sufficiently nonlinear so as to be challenging, but not to include all variables as arguments, thus preserving a "glass box" portion of the model. As an example, consider the problem *hs100lnp*. The original CUTEr problem is given as follows,

$$\begin{aligned} \min_{x\in\mathbb{R}^7} \quad & (x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6+7x_6^2+x_7^4-4x_6x_7-10x_6-8x_7 \\ \text{s.t.} \quad & 2x_1^2+3x_2^4+x_3+4x_4^2+5x_5-127=0 \\ & -4x_1^2-x_2^2+3x_1x_2-2x_3^2-5x_6+11x_7=0 \end{aligned} \tag{5.20}$$

This can be transformed into a glass box / black box problem by assuming the first constraint as black box. The output of this constraint is x_3, which will be relabeled as y. Then, labeling $w=[x_1,x_2,x_4,x_5]$ and $z=[x_6,x_7]$, we know that $d(w)=127-2w_1^2-3w_2^4-4w_3^2-5w_4$ and the whole problem (5.20) can be rewritten as:

$$\begin{aligned} \min_{w,y,z} \quad & (w_1-10)^2+5(w_2-12)^2+y^4+3(w_3-11)^2+10w_4^6+7z_1^2+z_2^4-4z_1z_2-10z_1-8z_2 \\ \text{s.t.} \quad & -4w_1^2-w_2^2+3w_1w_2-2y^2-5z_1+11z_2=0 \\ & y=d(w) \end{aligned} \tag{5.21}$$

Six parameter estimation problems were also adapted into glass box/black box models. They consist of four problems from the COPS [82] test set (*marine, methanol, gasoil, pinene*), one epidemiology example (*disease*) [83], and a final example, (*paramest*), which is problem 5 from [84]. These problems are discretized into several finite elements. One of the finite elements is then treated as a black box, while the dynamics in the remaining elements are

Figure 5.1: Converting dynamic models to glass box black box problems

handled explicitly with collocation using Radau roots. As shown in Figure 5.1, the inputs to the black box are the values of the parameters and the values of the states at time t_i, where t_i is the beginning of finite element i. The outputs after integration are the states at time t_{i+1}. Within the black box time horizon, the differential equations are integrated using SciPy's *odeint* function. Any data within finite element i are neglected for the parameter estimation.

For each of the parameter estimation problems, different variants were created by making each of the finite elements black box, except for the first finite element $i = 0$ and the last finite element. In Table 5.1, the number after the problem name refers to which finite element is treated as the black box. The total number of finite elements can then be inferred from the number of problem variants by adding 2 (the first and last finite element). For example, the problem *methanol* used 6 finite elements because there are 4 variants of the problem. When adding the CUTEr problems to our test set, we get a total set size of 62. The problems are listed in Table 5.1.

Table 5.1: Test Set

Problem	n_w	n_y	n_z	Alg 1, Linear		Alg 1, Quad		Alg 2, Linear		Alg 2, Quad	
				Iter	Evals	Iter	Evals	Iter	Evals	Iter	Evals
marine-1	23	8	328	–	–	–	–	31	727	–	–
marine-2	23	8	328	–	–	–	–	11	299	–	–

Problem	n_w	n_y	n_z	Alg 1, Linear		Alg 1, Quad		Alg 2, Linear		Alg 2, Quad	
				Iter	Evals	Iter	Evals	Iter	Evals	Iter	Evals
marine-3	23	8	328	–	–	–	–	12	324	–	–
methanol-1	8	3	129	–	–	–	–	94	922	–	–
methanol-2	8	3	132	–	–	–	–	4	49	5	203
methanol-3	8	3	135	–	–	–	–	4	49	–	–
methanol-4	8	3	135	–	–	–	–	3	39	11	434
paramest-1	4	2	116	7	47	8	113	48	293	41	311
paramest-2	4	2	116	7	47	–	–	3	23	2	37
paramest-3	4	2	118	7	47	7	97	12	77	10	105
paramest-4	4	2	118	7	47	7	97	10	65	10	105
paramest-5	4	2	118	7	47	7	97	10	65	12	117
paramest-6	4	2	118	7	47	7	97	10	65	12	117
paramest-7	4	2	118	7	47	–	–	11	71	13	123
paramest-8	4	2	116	7	47	–	–	11	71	13	123
disease-1	5	3	249	–	–	–	–	5	41	4	79
disease-2	5	3	249	–	–	–	–	6	48	4	79
disease-3	5	3	249	–	–	–	–	6	48	4	79
disease-4	5	3	249	–	–	–	–	5	41	4	79
disease-5	5	3	249	–	–	–	–	6	48	4	94
disease-6	5	3	252	–	–	–	–	6	48	4	94
disease-7	5	3	252	–	–	–	–	7	55	4	94
disease-8	5	3	246	–	–	–	–	7	55	4	94
disease-9	5	3	249	–	–	–	–	8	62	4	94

Problem	n_w	n_y	n_z	Alg 1, Linear		Alg 1, Quad		Alg 2, Linear		Alg 2, Quad	
				Iter	Evals	Iter	Evals	Iter	Evals	Iter	Evals
disease-10	5	3	249	–	–	–	–	11	77	4	94
disease-11	5	3	249	–	–	–	–	–	–	5	116
gasoil-1	5	2	146	–	–	–	–	3	27	4	94
gasoil-2	5	2	146	7	55	–	–	4	34	4	94
gasoil-3	5	2	146	–	–	–	–	4	34	4	94
gasoil-4	5	2	144	7	55	–	–	4	34	7	145
gasoil-5	5	2	144	7	55	–	–	4	34	6	138
gasoil-6	5	2	146	7	55	–	–	4	34	6	123
gasoil-7	5	2	148	7	55	–	–	4	34	6	123
gasoil-8	5	2	146	7	55	–	–	3	27	4	94
pinene-1	10	5	245	8	107	8	437	5	71	3	212
pinene-2	10	5	245	8	107	8	437	5	71	3	212
pinene-3	10	5	245	8	107	8	437	5	71	3	212
pinene-4	10	5	245	8	107	9	493	5	71	3	212
pinene-5	10	5	245	9	119	13	772	5	71	3	212
pinene-6	10	5	245	11	143	20	1241	6	83	3	212
hs0801	2	1	3	–	–	11	77	7	31	6	42
fletcher1	4	1	0	–	–	–	–	–	–	11	161
allinitc1	3	1	1	24	124	28	300	42	214	5	65
bt11	2	2	1	–	–	12	84	17	71	6	48
hs046	2	1	3	–	–	–	–	8	35	4	34
hs047	2	2	3	19	73	20	131	21	84	47	251

Problem	n_w	n_y	n_z	Alg 1, Linear		Alg 1, Quad		Alg 2, Linear		Alg 2, Quad	
				Iter	Evals	Iter	Evals	Iter	Evals	Iter	Evals
hs107	4	2	5	–	–	–	–	7	47	–	–
hs100lnp	4	1	2	–	–	13	193	–	–	6	111
rk23	3	1	14	–	–	–	–	8	44	4	54
hs99exp	7	7	24	–	–	–	–	–	–	10	406
hs067	3	1	6	–	–	–	–	–	–	4	54
dnieper	5	1	55	11	83	–	–	3	27	2	65
bt6	2	1	3	9	39	8	53	7	31	4	34
csfi2	2	1	2	–	–	–	–	9	39	6	48
csfi1	2	1	2	–	–	11	74	99	399	8	62
bt9	2	1	1	–	–	4	31	–	–	2	20
hs111lnp	4	1	6	–	–	–	–	–	–	24	399
hs0781	2	1	3	10	43	8	53	10	43	4	34
hs0751	2	1	2	–	–	–	–	3	15	3	27
hs0741	2	1	2	–	–	–	–	3	15	3	27
hs0771	2	1	3	–	–	–	–	16	67	7	55
hs0811	2	1	3	–	–	11	77	7	31	6	42

This test set is used to compare Algorithm 1 to the modified Algorithm 2. The original algorithm and the modified version are run using both linear and quadratic surrogate models for a total of four algorithmic variants. The default tolerances are listed as follows: feasibility $\epsilon_\theta = 10^{-6}$, criticality $\epsilon_\chi = 10^{-5}$, maximum sample radius for termination $\epsilon_\Delta = 10^{-5}$, and minimum allowable sampling region/trust region radius $\Delta_{min} = 10^{-6}$.

For the algorithms that do not use a sampling region, very few problems are solvable at

this tolerance level. The algorithms get close to optimality then proceed to take a sequence of very small f-type steps. The steps all remain successful but the rate of convergence is too slow to reach the desired tolerance in the limit of 10000 black box function evaluations. For the results obtained here for Algorithms 1 and 2, the tolerance ϵ_χ is relaxed to 10^{-3} in order to recognize that the solver had nearly identified an optimal solution before being trapped in the slow convergence pattern.

The full list of test problems and performance of each of the four methods (Algorithm 1 with Linear/Quadratic surrogate models, and Algorithm 2 with linear/quadratic surrogate models) is listed in Table 5.1. For all test problems, at least one of the four methods was able to solve the problem in under the 10000 function evaluation limit. The table presents the number of iterations and number of black box evaluations required to reach the solution. These results are also plotted using Dolan-Moré performance profiles in Figures 5.2 and 5.3. The performance measure τ represents the number of black box evaluations (Figure 5.2) or the number of iterations (Figure 5.3) required to solve the problem divided by the number of evaluations / iterations required by the best method for the given problem. The horizontal axis is $\log(\tau)$. The vertical axis represents the percentage of problems solved within that function evaluation / iteration budget. So higher lines on the left represent faster methods, while higher lines on the right side represent more robust methods.

The results indicate that the recent improvements to the algorithm have successfully improved the performance on this test set. Algorithm 2 is significantly more efficient (fewer function evaluations) and more robust (more problems solved) than Algorithm 1 with both linear and quadratic surrogate models. From Figure 5.3, it is clear that quadratic surrogate models reduce the iteration requirement for Algorithm 2. The number of iterations acts as a proxy for the computational expense of solving the subproblems. Normally we assume that sampling the black box is significantly more expensive, but if black box evaluations

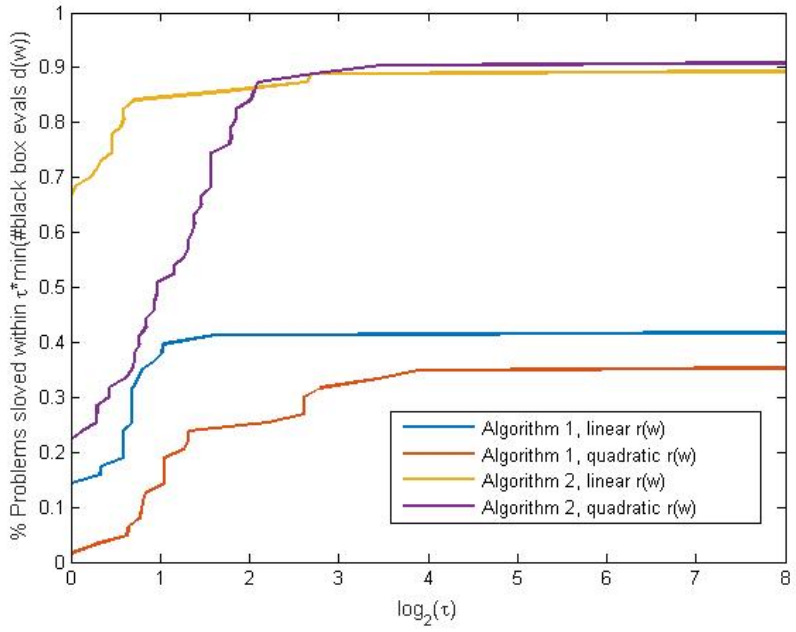


Figure 5.2: Performance profile for black box calls

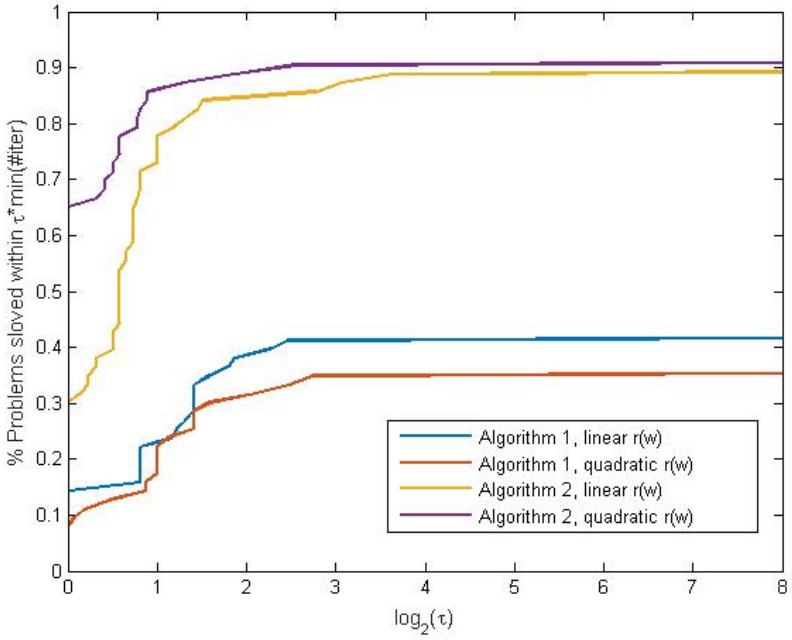


Figure 5.3: Performance profile for iterations

are relatively cheap then considering the number of iterations becomes important. Algorithm 2 with quadratic surrogate models is the clear choice in this case. However, Algorithm 2 with linear models generally requires fewer black box evaluations as shown on Figure 5.2 because fewer samples are needed to construct the surrogate model. In terms of robustness, Algorithm 2 with linear and quadratic models are roughly the same, with quadratic models slightly more robust. This makes sense as the quadratic models are able to solve some highly nonlinear problems that may fail with linear surrogate models. We note that all four methods were initialized with the same starting points and initial trust region, and all test problems are solvable by at least one of the four methods. Moreover, we observed that Algorithm 2 was more robust to different initializations than Algorithm 1. Nevertheless, there is still some room for improvement, since neither approach can solve all problems.

Finally, note that the termination tolerance for Algorithm 1 (without the sampling region) is looser than Algorithm 2. Despite this, the modified algorithm with sampling region greatly outperforms Algorithm 1. Without loosening the tolerance, the convergence of Algorithm 1 is too slow to be competitive with Algorithm 2. The sampling region was originally motivated by this very behavior, so the results indicate that it is successful in mitigating the problem. Without a sampling region, the test set on average does suffer from slow local convergence as a result of the small trust region. When adding the sample region, the bigger steps allow for much better local convergence behavior.

5.5 Conclusions

The trust region filter (TRF) method for glass box/black box optimization problems can be improved by adding various algorithmic modifications, including a sampling region,

step size adjustment strategy, and filter specifications. The sampling region allows decoupling the two purposes of the trust region, leading to larger steps and more robust convergence. The various other modifications helped keep the trust region size appropriate and terminate quickly. In addition, a Pyomo-based implementation of the method has been developed along with a set of 62 test problems. Using this test set as a guide, different versions of the algorithm can be benchmarked. The algorithmic modifications discussed in this chapter more than double the number of problems that are successfully solved compared to the vanilla Algorithm 1. Future directions include further tuning of the algorithm, and consideration of automatic or adaptive choices of functional forms for the surrogate models.

Chapter 6

Case Studies

In this chapter, several case studies of black box / glass box optimization are presented. The trust region filter (TRF) algorithm is applied to solve problems in chemical process optimization, carbon capture power generation, and efficient physical property models.

In most of these case studies, equation oriented process models are used to handle standard unit operations, while reactor models are treated as black box. For chemical processes, the concept of sequential modular simulation provides a robust method for finding feasible flowsheets, and is used here to find a feasible point as part of a restoration procedure.

6.1 Williams-Otto

The Williams-Otto process is a classical example problem for process optimization. This model is used as a simple example to demonstrate the TRF approach and compare to alternative optimization approaches. Even though the full equation oriented formulation is available, the reactor is treated as a black box. Therefore, the performance of Algorithm 1 can be compared to solution of the original problem. In this chemical process, a feed stream is mixed with a recycle stream and fed into a reactor. Three sequential second or-

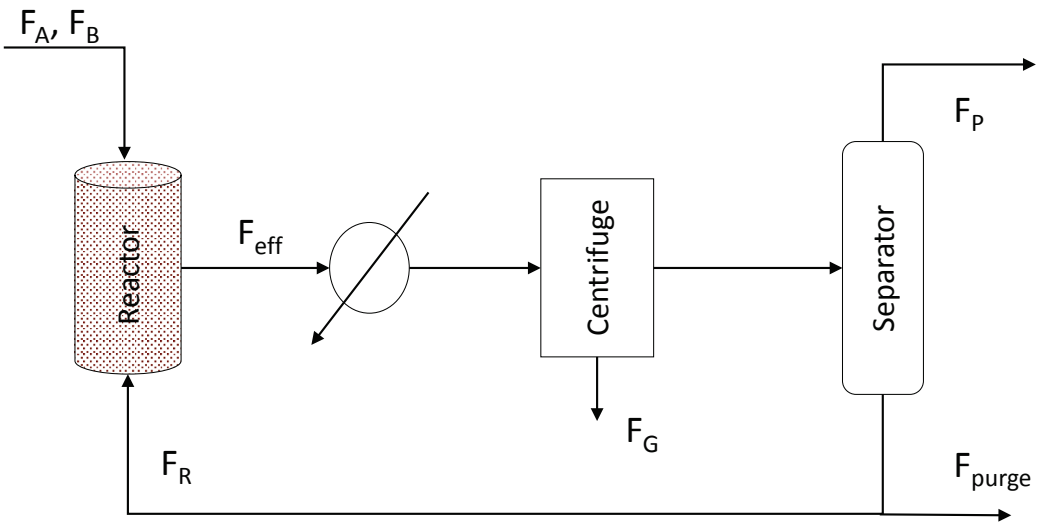


Figure 6.1: Williams-Otto flowsheet

der reactions occur, of which the third reaction produces an unwanted byproduct.

$$A + B \rightarrow C,$$

$$B + C \rightarrow P + E,$$

$$P + C \rightarrow G.$$

The reactor effluent passes through a heat exchanger before the byproduct is precipitated out. Finally a separator uses a fixed split fraction to separate the product and recycle streams.

The full optimization formulation from [2] is given as follows:

$$\begin{aligned} \max ROI = \; & 100(2207F_P + 50F_{purge} - 168F_A - 252F_B \\ & - 2.22F_{eff}^{sum} - 84F_G - 60V\rho)/(600V\rho) \end{aligned}$$

The kinetics, given by the following equations, are treated as a black box. The y variables in formulation (4.2) correspond to the extents of reaction $y = [r_1, r_2, r_3]$ and the w variables correspond to reactor conditions $w = [T, x_A, x_B, x_C, x_P, V]$. Therefore, the right hand side of these constraints is the black box function $d(w) : \mathbb{R}^6 \rightarrow \mathbb{R}^3$:

$$r_1 = a_1\, exp(-120/T)\, x_A x_B V \rho,$$

$$r_2 = a_2\, exp(-150/T)\, x_B x_C V \rho,$$

$$r_3 = a_3\, exp(-200/T)\, x_P x_C V \rho.$$

where $\rho = 50$, $a_1 = 5.9755$, $a_2 = 2.5962 \times 10^{12}$, and $a_3 = 9.6283 \times 10^{15}$ are constants. The remaining process models are all glass box. All remaining variables not classified above correspond to the z variables in formulation (4.2). The constraints are given as follows:

Reactor balance equations:

$$F_{eff}^A = F_A + F_R^A - r_1,$$

$$F_{eff}^B = F_B + F_R^B - (r_1 + r_2),$$

$$F_{eff}^C = F_R^C + 2r_1 - 2r_2 - r_3,$$

$$F_{eff}^E = F_R^E + 2r_2,$$

$$F_{eff}^P = 0.1F_R^E + r_2 - 0.5r_3,$$

$$F_{eff}^G = 1.5r_3,$$

$$F_{eff}^{sum} = \sum_j F_{eff}^j, \quad j \in \{A, B, C, E, P, G\}$$

$$F_{eff}^j = F_{eff}^{sum} x_j$$

Waste Stream:

$$F_G = F_{eff}^G$$

Table 6.1: Williams-Otto flowsheet optimization

Type of surrogate	Iterations	d(w) function calls
Kriging (average)	123	3141
Linear, perturbation size: 1e-5	13	91
Linear, perturbation size: $\Delta_k/2$	92	644
Other solvers:		
fmincon 'sqp' (finite difference gradients)	15	210
Pattern Search	158	15190
fminsearch	failed	failed

Product Stream:

$$F_{purge} = \eta(F_{eff}^{A} + F_{eff}^{B} + F_{eff}^{C} + 1.1F_{eff}^{E}).$$

Recycle Stream:

$$F_{R}^{j} = (1-\eta)F_{eff}^{j}, \quad j \in \{A, B, C, E, P, G\}.$$

Bound Constraints:

$$V \in [0.03, 0.1],\ T \in [5.8, 6.8],\ F_P \in [0, 4.763],$$

$$F_{purge}, F_G, F_{eff}^{j} \geq 0,\ F_A, F_B \geq 1.$$

First, this problem was used to compare the performance of Algorithm 1 for three strategies for building surrogate models. For black box/glass box optimization, the metric for evaluating algorithmic performance will be primarily the number of black box function evaluations required to reach the optimum solution. As in derivative free optimization,

the cost of evaluating the black box usually outweighs any computational effort expended in the optimization algorithm proper.

The first method used Kriging models (Gaussian process regression), constructed with the DACE MATLAB toolbox [85]. The Kriging hyper-parameters were optimized with the default DACE settings within the range $[0.01, 20]$. At each iteration, Latin hypercube samples (n = 18) were generated around the trust region center, which almost surely guarantees a well-posed positive spanning set and therefore fully linear models [74]. To further increase accuracy, points were re-used from previous iterations whenever the points lay within the trust region. Points cannot be re-used when they lie within some threshold distance of another sample point (to prevent ill-conditioning in construction of the Kriging model). Because of the random nature of Latin hypercube samples, the overall optimization was run 10 times and the average results are given in Table 6.1. The same optimum solution, in agreement with the original model, was obtained for each of the 10 runs with an average expense of 3141 function evaluations of the kinetics model.

The second surrogate modeling approach uses a linear model built with finite difference perturbations. Note that one may assume that the derivative is sufficiently accurate and the trust region does not have to be forced to zero, i.e. Step 4 of Algorithm 1 may be omitted. (For this problem, omitting Step 4 did not impact the results). As shown in Table 6.1, this achieves a drastic reduction in the number of function calls to reach the correct solution. This suggests several explanations: a) the linear functional form of the surrogate model decreases the nonconvexity of $TRSP_k$ and allows for longer steps, or b) the accurate derivative information allows for better search directions. To test this, the following strategy was proposed.

The third surrogate strategy uses forward differences with a perturbation size of $\Delta_k/2$. This maintains the linear functional form, but relies on the κ-fully linear property and

the trust region must go to zero to guarantee convergence. The performance degrades in comparison to the finite difference derivatives, but the linear models still outperform the Kriging models. Any additional information in the nonlinear Kriging models fails to offset the performance penalty from more complicated subproblems. There may also be an issue with multiple local minima in the trust region subproblem due to the high degree of nonlinearity of the Kriging models. In any case, the results suggest that the value of the derivative at the trust region center cannot be over-emphasized and the finite difference case offers superior performance.

If derivative estimates are obtained, one may ask whether Algorithm 1 offers any benefit over competing NLP approaches. The derivative estimates may be given to a nonlinear programming solver directly and treated as a typical glass box NLP. MATLAB's fmincon SQP algorithm was applied to the problem directly using finite difference derivatives for the kinetics equations. Since the solver makes no distinction between the glass box and black box constraints, the black box is called more often than when using the trust region filter (210 calls vs. 91). This demonstrates the benefit of Algorithm 1 to exploit the structure of the problem by isolating the black box constraints and only calling the black box when absolutely necessary.

The final two rows in Table 6.1 show the performance when MATLAB's derivative free solvers 'fminsearch' and 'patternsearch' are applied to the problem. To apply the derivative free methods, inequality constraints are moved to the objective function using a penalty function as introduced in Chapter 2. Then, a set of independent variables are fixed and the flowsheet is converged. The independent variables in this case were F_A, F_B, T, V, and η. Unconstrained derivative free techniques may then be applied to the five dimensional problem. The pattern search algorithm successfully found the same solution as the trust region filter method in 158 iterations. However, the flowsheet must be converged

multiple times at each iteration, and each converging each flowsheet requires multiple calls to the black box function. The 'fminsearch' algorithm (based on the Nelder-Mead simplex approach) converges a point where the penalized inequalities are significantly violated. Changing the penalty parameter could not correct this behavior. From the high cost of the pattern search method and the failure of the Nelder-Mead method, it is concluded that treating this problem as a black box is not an efficient solution technique. Recognizing the glass box/black box structure and using it in the trust region filter method can greatly reduce the computational effort required to obtain a solution.

6.2 Ammonia Synthesis

The ammonia synthesis process was used as a second chemical process optimization case study for the trust region filter method. Ammonia is synthesized at high pressure from gaseous nitrogen and hydrogen at high pressure in a fixed bed reactor. The ammonia product is purified with a sequence of two flash vessels and remaining reactants are recycled (see Figure 6.2).

Standard process units, consisting of three compressors, four heat exchangers, and two flash units, were modeled in the GAMS-based equation oriented process optimization framework presented in [86]. This framework generates process models for these units using the SRK equation of state as the thermodynamic model. The objective function maximizes revenue from ammonia sales minus the utility costs.

For the ammonia synthesis reactor, a differential algebraic model was adapted from Murase et al. [87]. The reactor model captures complex trade-offs between temperature, pressure, composition, and conversion, which are crucial for overall process economics. The differential equations of the reactor (describing heat transfer and reaction rate) were

discretized sing Radau IIa collocation [88]. The resulting system of algebraic equations can then be solved independently as a system of nonlinear equations or alternatively included as constraints in the simultaneous collocation approach. This simultaneous approach (first shown for the ammonia process in [89]) generates an NLP with 2,281 variables and 2,297 constraints, This model is straightforward to solve with an equation oriented solver. On a desktop with Intel i7-4770 CPU with 16 GB of RAM, the problem takes only 0.548 CPUs in GAMS 24.2 using CONOPT 3.15M. However, if the reactor model is too large or if the details of the model are unavailable, the simultaneous approach may not be possible. Consequently, we treat the reactor as a black box in order to demonstrate the glass box/black box approach. The solutions may then be verified against the simultaneous case.

The discretized differential algebraic equation system to model the ammonia reactor is given as follows. First, introduce the set of components

$$c = \{H_2, N_2, NH_3, Ar, CH_4\},$$

the set of finite elements,

$$t = \{1, ..., 15\},$$

and the set of collocation points

$$j = \{1, 2, 3\}.$$

Define partial pressures:

$$P_{c,t,j} = P_{tot} \frac{N_{c,t,j}}{\sum_c N_{c,t,j}} \quad \forall t, j \tag{6.1}$$

Component balances:

$$N_{H_2,t,j} = N^0_{H2} - 3(N^0_{N2} - N_{N_2,t,j}) \tag{6.2}$$

$$N_{NH3,t,j} = N^0_{NH3} + 2(N^0_{N2} - N_{N_2,t,j}) \tag{6.3}$$

$$N_{Ar,t,j} = N^0_{Ar} \tag{6.4}$$

$$N_{CH4,t,j} = N^0_{CH4} \tag{6.5}$$

Rate expressions:

$$rate^1_{t,j} = exp(k^1_{t,j}) \tag{6.6}$$

$$rate^2_{t,j} = exp(k^2_{t,j}) \tag{6.7}$$

$$k^1_{t,j} = 9.7923 - 20800/(RT^g_{t,j}) \tag{6.8}$$

$$k^2_{t,j} = 37.7858 - 47400/(RT^g_{t,j}) \tag{6.9}$$

$$\mathcal{R}_{t,j} = rate^1_{t,j}\frac{P_{N2,t,j}(P_{H2,t,j})^{1.5}}{P_{NH3,t,j}} - rate^2_{t,j}\frac{P_{NH3,t,j}}{(P_{H2,t,j})^{1.5}} \tag{6.10}$$

Differential Equations:

$$\dot{T}^f_{t,j} = \frac{-S_1}{WC_{pf}}U(T^g_{t,j} - T^f_{t,j}) \tag{6.11}$$

$$\dot{T}^g_{t,j} = \frac{-S_1}{WC_{pg}}U(T^g_{t,j} - T^f_{t,j}) + \frac{\Delta HS_2}{WC_{pg}}\mathcal{R}_{t,j} \tag{6.12}$$

$$\dot{N}_{t,j} = -\mathcal{R}_{t,j} \tag{6.13}$$

Collocation:

$$T^f_{t,j} = T^{f0}_t + h\sum_{k=1}^{3}\omega_{j,k}\dot{T}^f_{t,k} \tag{6.14}$$

$$T^g_{t,j} = T^{g0}_t + h\sum_{k=1}^{3}\omega_{j,k}\dot{T}^g_{t,k} \tag{6.15}$$

$$N_{N2,t,j} = N^0_{N2,t} + h\sum_{k=1}^{3}\omega_{j,k}\dot{N}_{t,k} \tag{6.16}$$

Continuity:

$$T^{f0}_t = T^f_{t-1,3} \quad t \geq 2 \tag{6.17}$$

$$T^{g0}_{t,j} = T^g_{t-1,3} \quad t \geq 2 \tag{6.18}$$

$$N^0_{N2,t} = N_{N2,t-1,3} \quad t \geq 2 \tag{6.19}$$

$$T^f_{NE,3} = T^f_{init} \tag{6.20}$$

$$T_1^{g0} = T_1^{f0} \tag{6.21}$$

$$N_{N2,0}^0 = N_{init}^0 \tag{6.22}$$

sBased on the Williams-Otto results, the trust region filter method uses linear surrogate models in place of the reactor. The black box function $d(w)$ outputs the extent of reaction and conditions as a function of inlet stream conditions. Each time the black box is called, the differential algebraic system is solved to return the reactor effluent stream conditions. To construct the linear models, the following strategies were compared: forward difference perturbations of $0.8\Delta_k$, forward difference perturbations of $\min[0.01, 0.8\Delta_k]$, and linear regression models using Latin hypercube samples. In the first two cases, the trust region filter method converges in 30 iterations without rejecting a single step or calling restoration as shown in Table 6.2. The similar performance suggests that the reactor model is relatively smooth and the perturbation size does not greatly impact the linearizations. The linear regression models perform slightly worse due to several rejected steps and a more prolonged sequence of criticality steps. Regression models do not appear to offer a benefit for the increased sampling effort (15 samples were used per iteration compared to 8 samples for interpolation). All instances of the trust region filter method converged to the same solution as the simultaneous approach, thus validating the results. However, the interpolation scheme that favors smaller perturbations is able to reduce the calls to the reactor model (168 vs. 240) because it is not always necessary to rebuild the surrogate model when the trust region shrinks in a criticality step. This suggests the later development of the sampling region. The trust region subproblems have 1,257 variables and 1,289 constraints and are solved by CONOPT in 0.133 CPUs. This problem demonstrates that the trust region filter method can work well as an alternative to the simultaneous approach when the reactor model is very large, or when the model details are not available to construct an equation-based discretization. The next case study provides such an example.

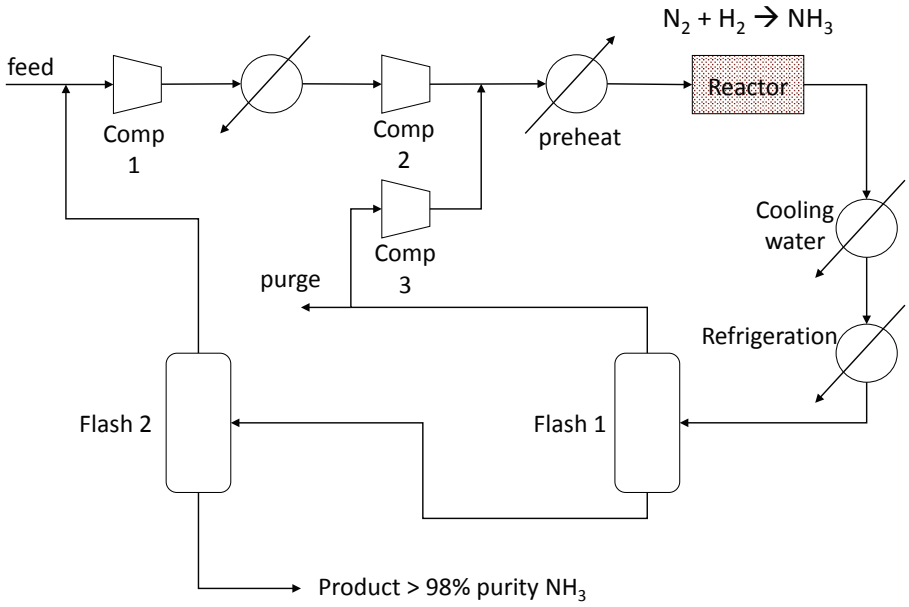


Figure 6.2: Ammonia synthesis flowsheet

Table 6.2: Ammonia synthesis optimization using TRF method

Type of surrogate	Iterations	d(w) function calls
Linear, perturbation size: $0.8\Delta_k$	30	240
Linear, perturbation size: $\min[0.01, 0.8\Delta_k]$	30	168
Linear, regression to Latin hypercube sample $(n = 14)$	46	690

6.3 Power Plant Optimization

The algorithm was used for optimization of an oxycombustion power plant. In oxycombustion, a mixture of purified oxygen and recycle carbon dioxide is used to combust fuel in a boiler. This results in a flue gas that is primarily water and CO_2. The water is easily separated and the task of purifying the CO_2 is greatly simplified because most nitrogen was removed before combustion. Compared to conventional air fired power plants, the oxycombustion process has several key differences. The introduction of new separation tasks, namely an air separation unit and a compression and purification unit, lower the overall power output of the plant due to the energy required to run them. In addition, the recycle loop further couples the temperature and composition of pre- and post- combustion gas streams. Finally, the oxy-fired boiler behaves differently than air-fired boilers, and it is necessary to consider the effect of these changes on the rest of the power plant design. To rigorously manage the interactions between these subsystems and reduce the cost of carbon capture, a comprehensive optimization approach is proposed. Optimization of a conventional air-fired power plant is also conducted to analyze the cost of carbon capture on a consistent basis (i.e. using the same models and assumptions).

Most unit operations in a power plant can be modeled using algebraic equations. As in the ammonia case study, units such as heat exchangers and turbines are modeled using the equation-oriented modeling framework of [86]. However, modeling the boiler presents special challenges, and detailed first principles models are required. A full scale CFD simulation including reactions and transport behavior in three dimensions can take several weeks to solve. Instead, we use a hybrid 1D/3D zonal boiler model as described in [90]. The hybrid boiler model uses a series of nine vertical zones to model reactions and particle/gas flow and integrates the radiation PDE in three dimensions with the discrete ordinate method on 21888 finite elements. The resulting model can converge in about 1 CPU

minute on a desktop computer. The model is custom built in C++ with specialized methods to help guarantee robustness when converging the simulation. Both design and operational decisions can be modified for the boiler. For this study, boiler design was fixed, using the geometry and burner configuration of an existing utility boiler (Pacificorp's Hunter 3 unit). The hybrid 1D/3D boiler model is approximated with surrogate models and used to solve a plant-wide optimization problem with the trust region filter method.

For the purposes of the optimization problem, the boiler model can be viewed as a function from $\mathbb{R}^m \rightarrow \mathbb{R}^p$, where the input and output variables are chosen to capture the interactions of the boiler with the rest of the power plant. These inputs and output variables are given below (corresponding to the w and y variables in the glass box/black box formulation (4.2)):

Boiler Inputs:

1. Primary air temperature
2. Secondary/over-fired air temperature
3. Average temperature of boiling water inside water wall tubes
4. Average secondary superheater steam temperature
5. Primary air component flowrates (O_2, N_2, H_2, CO, CO_2, H_2O, SO_2, H_2S, CH_4, Ar)
6. Secondary air component flowrates (same components as primary air, but different compositions)
7. Over-fired air total flowrate (same composition as secondary air)

Boiler Outputs:

1. Boiler enclosure water wall heat duty
2. Secondary superheater heat duty
3. Flue gas component flowrates (same components as primary air)

4. Flue gas temperature

The primary air is the gas stream into the boiler that carries the pulverized coal particles, whereas secondary and over-fired air streams are added into the boiler directly to aid in combustion. The temperature and composition of these streams have strong impact on the combustion behavior in the boiler. In general, higher heat transfer through the boiler enclosure wall is better, but the flue gas temperature must remain below a bound for materials considerations. The compositions flowing into the boiler are indirectly coupled with the composition leaving the boiler through recycle, and the heat transfer behavior is also indirectly coupled with the average temperature of water in the water wall tubes through the steam cycle. The use of detailed process models helps capture these interactions accurately.

Figure 6.3 shows the general configuration of the steam and gas sides of the steam cycle. On the steam side, high pressure water enters at the bottom of the boiler and steam exits the boiler after the secondary superheater. The turbines are divided into high, intermediate, and low pressure sections (HP, IP, and LP respectively). Steam extraction is considered with a superstructure of potential steam extraction sites between stage groups. The extracted steam may be sent to the boiler feedwater heaters. Heat exchangers are modeled with heat exchanger halves, specified either as heater or cooler. The heating and cooling duties are matched for certain streams that are known to use a single heat exchanger in most power plants, including the primary superheater, reheaters, and the economizer. The remaining heat exchanger halves are integrated with a pinch-based model. This heat integration model is developed using the Duran-Grossmann formulation [91]. The deaerator is simply modeled as a flash vessel, and the outlet stream is bounded between 4.8-9.3 bar and 420-450 K, to be consistent with the recommendation of [92]. Then, the water is pumped to high pressure before entering the boiler.

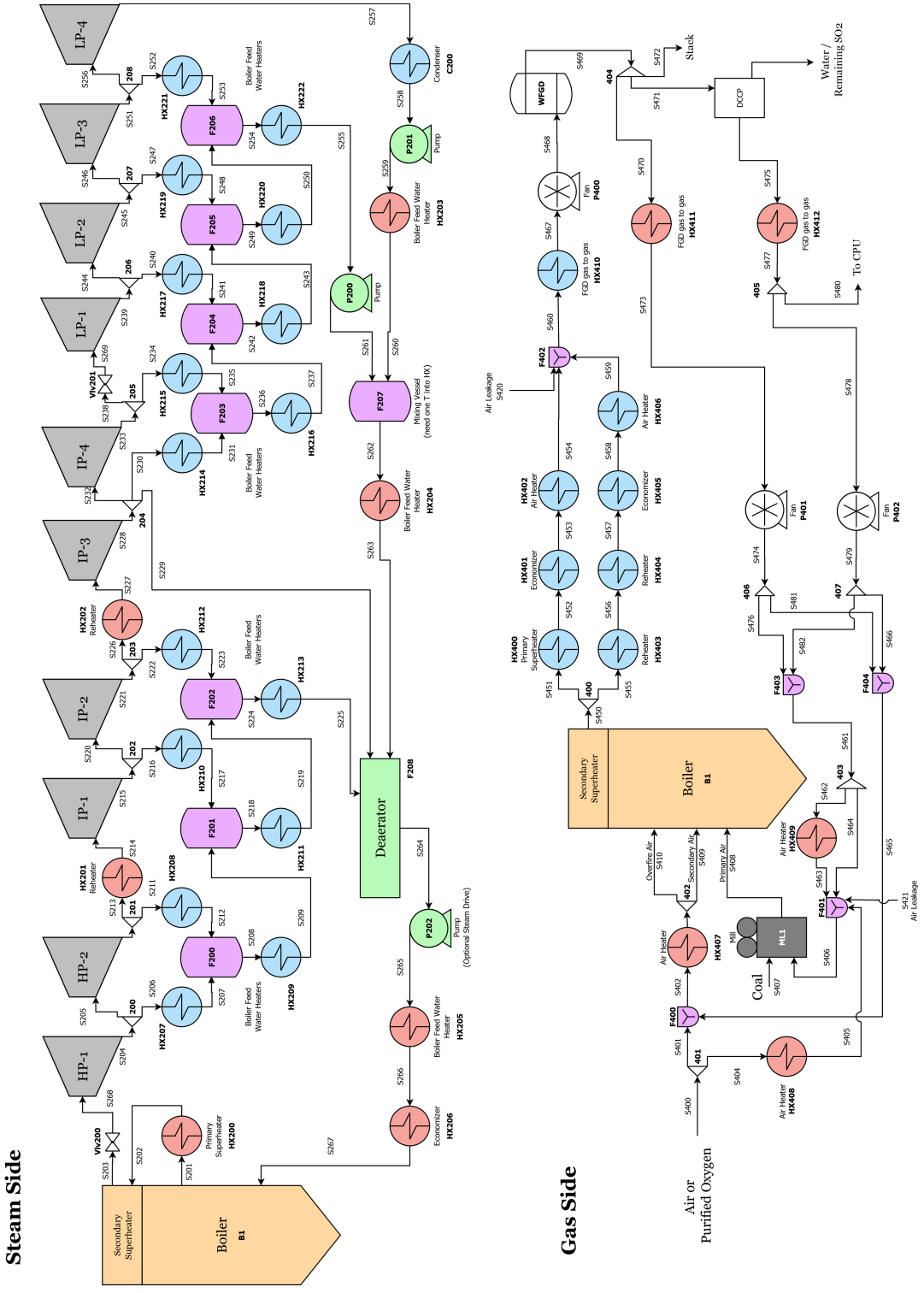


Figure 6.3: The water/steam and gas sides of the steam cycle

On the gas side, Figure 6.3 considers both of air- and oxy-fired configurations. The input stream on the left, either air or purified oxygen, is split to be sent to primary and secondary air. Primary air flow rate is bounded below to ensure at least a 2:1 gas-to-coal ratio to ensure that the gas can carry the coal. For safety reasons, the primary air stream also has upper bounds on temperature and (in oxy-fired mode) O_2 mole fraction of 400 K and 35%, respectively. In the oxyfired case, the oxygen is then mixed with recycled flue gas. This also allows primary and secondary air streams to have different compositions. In either firing mode, the flue gas is split to two series of heat exchangers, then sent to pollution controls. After sulfur dioxide is removed in the Wet Flue Gas Desulfurization (WFGD) unit, the air fired configuration will send all flue gas to the stack, perhaps recovering some heat for pollution control processes on the way. In the oxy-fired configuration, the flue gas is again split to two streams. One stream is sent to the direct contact cooler/polishing scrubber (DCCP), which removes much of the water from the flue gas as well as any residual SO_2. The ratio of flue gas sent for cooling is also a key decision variable due to the role of water vapor in the radiation behavior in the boiler. Then, some flue gas is sent for compression and purification while the rest is recycled to the boiler. Additional assumptions about the model are listed in Table 6.3.

The trust region filter algorithm was used to maximize the thermal efficiency of a double reheat oxy-fired steam cycle. The optimization problem is summarized in (6.23). Both design and operational decisions could be modified for the boiler, but we consider the case where boiler geometry and burner configuration match an existing utility boiler (Pacificorp's Hunter 3 unit).

The objective was to maximize thermal efficiency with a fixed coal feed flow rate, with

Table 6.3: Assumptions for steam cycle optimization case study

Steam Side Pressure Drops			
Primary Superheater (HX200)	2.5%	Reheaters (HX201 & 202)	8%
Feed Water Heaters (HX203 - 205)	4%	Economizer (HX206)	4%
Throttle Valve (Vlv200)	4%	IP/LP Crossover (Vlv201)	3%
Gas Side Pressure Drops			
Primary Superheater (HX400)	2 %	Reheaters (HX403)	2%
Economizer (HX401 & 405)	2%	Air Heaters (HX402 & 406-408)	2%

a small penalty for utility consumption.

$$
\begin{aligned}
\max \quad & \text{Thermal Efficiency} + \rho_w Q_w \\
\text{s.t.} \quad & \text{Thermal Efficiency} = \frac{\sum W_{turbine} - \sum W_{pump} - \sum W_{fan} - W_{CPU} - W_{ASU}}{\text{Thermal Input Rate}} \\
& \text{Fixed Thermal Input Rate} \\
& \text{Steam Turbine Model} \\
& \text{Pump and Fan Models} \\
& \text{Pinch-Location Heat Integration Equations} \\
& \text{Flue Gas Thermodynamics Model} \\
& \text{Steam Thermodynamics} \\
& \text{Hybrid Boiler Model} \\
& \text{Correlation Model ASU (oxy-fired configuration only)} \\
& \text{Correlation Model CPU (oxy-fired configuration only)}
\end{aligned}
\tag{6.23}
$$

where ρ_w is a small penalty term for cooling water usage. The air separation unit (ASU)

and carbon dioxide processing unit (CPU) are modeled with correlations derived from [86] and [93]. Both steam thermodynamics (using steam tables) and the hybrid boiler model are incorporated into (6.23) using linear surrogate models. For the steam thermodynamics, enthalpy and entropy are approximated as linear functions of temperature and pressure. The finite difference perturbation size for the boiler was set as $\min\{0.1, 0.8\Delta_k\}$, while steam table perturbation size was fixed at 10^{-5}. Note that the default perturbation size for the boiler is larger in order to compensate for greater numerical noise in the boiler model outputs. The boiler simulations are run in parallel on 4 cores when collecting samples to construct the surrogate model.

The details of the optimization solution are provided in Table 6.4. As would be expected when maximizing thermal efficiency, the flue exit gas temperature and steam exit temperatures are pushed to their upper bounds of 1600 K and 835 K, respectively for both case studies. The value of these bounds has a great impact on the overall steam cycle performance, but comes at the expense of material costs to withstand the high temperatures. For the purposes of this study, these parameters are assumed fixed. Interestingly, we only have a 5.7% penalty for oxycombustion. In carbon capture studies, penalties closer to 10% are more common. These simulation studies often treat the boiler as an equilibrium reactor and its radiative characteristics are ignored (see [94] for a review). Here, the oxy-fired configuration is able to recover more energy from the fuel, before accounting for ASU and CPU energy usage. This is largely due to the beneficial effect of higher emissivity and heat capacities of the recycled CO_2 and water as compared to air. Without using a detailed boiler model, these interactions may be neglected.

Each trust region subproblem is an NLP with 4204 variables and 4680 constraints for the air fired case, and slightly higher for oxy-fired. The NLP subproblems are built in GAMS version 24.2 and solved with CONOPT 3 [95]. The boiler simulations are run in parallel on

Table 6.4: Steam cycle optimization results

	*A*ir	*O*xy
Flue exit gas temperature (K)	1600	1600
Steam exit temperature (K)	835	835
Steam exit pressure (bar)	223	223
Work from Turbines (MW)	525.4	570.3
Pumping Work (MW)	11.0	12.4
Heat from Boiler (MW)	650.8	531.3
Boiler Walls	575.6	457.3
Secondary Superheater	75.2	74.0
Fuel Heat Rate (MW)	1325.5	1325.5
Net Power (MW)	515.5	440.4
Thermal Efficiency (HHV)	**38.9%**	**33.2%**
Trust region filter optimization		
Solution time (CPUh)	9.8	8.6
Boiler simulations	759	598

4 cores when collecting samples to construct the surrogate model.

It would be difficult to solve a glass box/black box problem of this size using existing tools. The number of variables is far beyond the scope of derivative free optimization. To apply derivative free optimization methods, one could instead identify independent variables and optimize in that reduced space. However, this involves converging the remaining constraints (i.e. converging the flowsheet) for every function call in the optimization, which is prohibitively expensive. Alternatively, finite difference derivative estimates of

the black box functions could be given directly to an NLP solver. However, NLP solvers always assume that the derivative is accurate and judge progress based on these derivatives. If the black box model derivatives from finite differences are inaccurate, the NLP solver may fail. By contrast, the trust region inherently plans for possible inaccuracies in the surrogate model and the filter judges progress without using derivatives. Therefore, the proposed glass box/black box method leads to a solution of this problem.

One may ask what the optimizer found to enable the better efficiency than literature reported designs. The optimization model can be easily modified to quantify the effect of several design choices. Consider this example, where the effect of oxygen purity from the ASU is examined. Optimization problem (6.23) was solved in oxy-fired configuration for two different scenarios. In Case A, the oxygen purity supplied by the ASU was fixed at 97 mol%, which means that the power requirement of the ASU is only dependent on the flowrate of oxygen supplied. In Case B, the oxygen purity was allowed to vary between 90 mol% and 98 mol%. This allows the optimizer to trade-off the pre- and post-combustion separation tasks, while simultaneously considering the interactions with the detailed kinetics and radiation behavior in the boiler. In the lower oxygen environment, gasification reactions are more favored and the emissivity of the gas mixture changes. In Case A, the optimum design found by the trust region filter algorithm is an oxy-fired power plant with a net power output of 437.4 MW and net efficiency of 33.0%. In Case B, the optimum solution has a net power output of 440.4 MW and efficiency of 33.2%[1]. Interestingly, in Case B the oxygen composition is pushed to its lower bound of 90 mol%. In both scenarios the optimizer pushes the steam temperatures leaving the secondary superheater and reheaters to their upper bounds of 835 K and 867 K, respectively, as expected when maximizing thermal efficiency. Similarly, the lower bound of 0.068 bar for the condenser

[1]Higher heating value basis

operating pressures is active. The optimizer also pushes the temperature and oxygen content of the primary flue gas recycle streams (S408) to their upper bounds of 400 K and 35 mol%. Another interesting conclusion is in the recycle distribution of the flue gas. The temperature and composition of the flue gas (influenced by drying) has complex interactions with the detailed boiler model. By optimizing using the hybrid boiler model, these interactions can be considered and additional efficiencies are identified. However, it is also important to validate these predictions with detailed CFD simulations or pilot scale data.

Table 6.5: Power plant optimization results.

Case A: fixed oxygen purity. Case B: variable oxygen purity

	Case A	**Case B**
Work from Turbines (MW)	568.2	570.3
HP	94.5	94.5
IP	267.3	267.7
LP	206.4	208.1
Pumping Work (MW)	12.4	12.4
Fan Work (MW)	3.6	3.7
Heat from Boiler (MW)	520.6	531.3
Boiler Walls	446.4	457.3
Secondary Superheater	74.2	74
Heat from Flue Gas (MW)	659.2	653.0
Primary Superheater	201.0	220.5
Reheater (HX201)	168.6	168.7
Reheater (HX202)	146.6	148.4
Economizer	143.0	115.4
Heat Rejected (MW)	620.9	623.3
Fuel Heat Rate (MW)	1325.5	1325.5
ASU Power (MW)	71.2	65.6
CPU Power (MW)	43.6	48.2
Net Power (MW)	437.4	440.4
Thermal Efficiency (HHV)	**33.0%**	**33.2%**

Flue Gas Recycle Distribution		
Bypasses DCCP, to Secondary Rec.	29.7%	31.3%
To CPU after DCCP	28.0%	29.9%
To Primary Recycle after DCCP	23.6%	25.3%
To Secondary Recycle after DCCP	18.7%	13.5%

6.4 Solid Sorbent Carbon Capture System

In this case study, we consider an alternative carbon capture technology. CO_2 is captured from a flue gas stream using solid sorbent. The system consists of two bubbling fluidized beds as shown in Figure 6.4. The first bubbling fluidized bed is the adsorber that removes CO_2 by contacting solid sorbent with flue gas from a power plant. The second column is

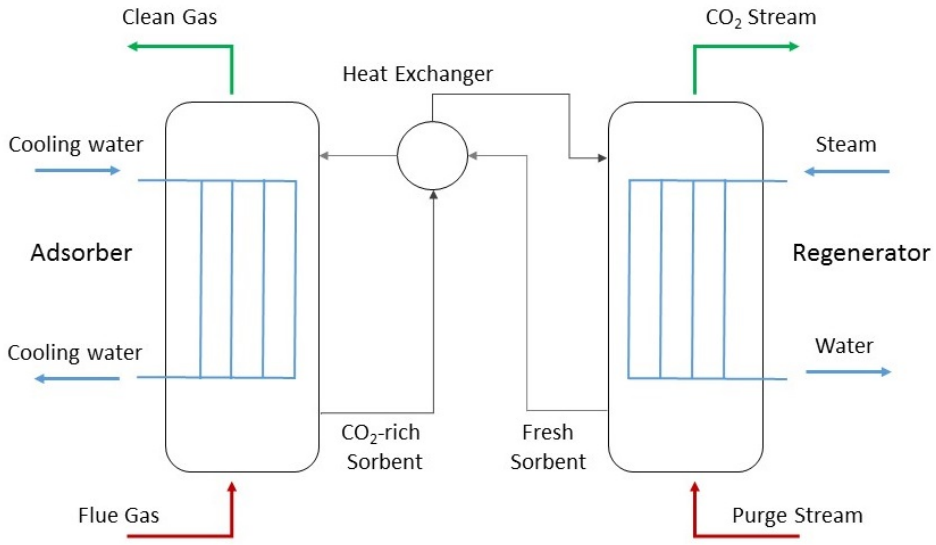


Figure 6.4: Carbon capture system, regenerator column is treated as black box

the regenerator, which is used to remove the CO_2 from the solid sorbent. The CO_2 can then be sent for sequestration or utilization. The adsorber is cooled with cooling water to promote CO_2 adsorption, while the regenerator is heated with steam to promote desorption. The sorbent is transported between the two columns with a heat exchanger in between to preheat (precool) the sorbent before entering the regenerator (adsorber).

Both the adsorber and regenerator are modeled as bubbling fluidized bed reactors. The model was developed by NETL [96, 97] to predict axial and temporal variations in concentrations and temperatures. For this case study, the adsorber is modeled as a fully equation oriented glass box model, while the regenerator is considered to be black box. In the adsorber model, solid sorbent is added at the top of the column, while flue gas is added at the bottom. The amine adsorption reactions occur in the solid phase:

$$\begin{aligned} &\mathrm{H_2O_{(g)}} \rightleftharpoons \mathrm{H_2O_{(phys)}} \\ &2\,\mathrm{R_2NH^+CO_2} \rightleftharpoons \mathrm{R_2NH_2}^+ + \mathrm{R_2NCO_2}^- \\ &\mathrm{R_2NH + CO_2 + H_2O_{(phys)}} \rightleftharpoons \mathrm{R_2NH_2}^+ + \mathrm{HCO_3}^- \end{aligned} \tag{6.24}$$

where R is an organic functional unit. Flow within the column is modeled using three flow regimes: a downward flowing emulsion region (containing solids), upward flowing bubble region (containing no solids), and upward flowing cloud wake region (containing some solids). Heat and mass transfer among these regions are described by a set of 20 partial differential equations. The full model is presented elsewhere [97], but as an example consider the gas phase mass balance for the bubble region:

$$\frac{\partial c_{b,j}}{\partial t}\delta A_x = -\frac{\partial G_b y_{b,j}}{\partial x} - A_x \delta K_{bc,j}(c_{b,j} - c_{c,j}) + K_{g,bulk,j} \tag{6.25}$$

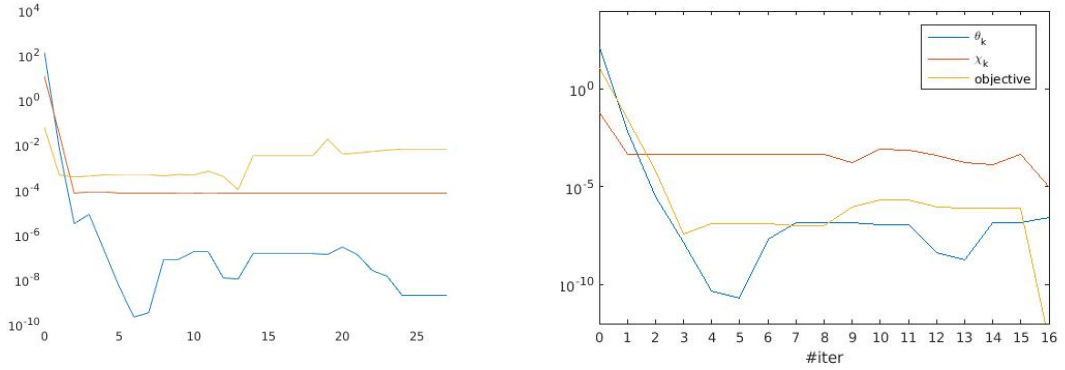
The left hand side accounts for the accumulation of component j in the bubble region, where $c_{b,j}$ is the concentration of component j in the bubble region, δ is the volume fraction of the bubble region, and A_x is the cross-sectional area. The first right hand side represents

the upward gas flow in the bubble region, where G_b is the axial flow rate and $y_{b,j}$ is the gas mole fraction for component j in the bubble region. The second term on the right hand side represents the mass transfer between the bubble and the cloud-wake regions, where $K_{bc,j}$ is the mass transfer coefficient and $c_{c,j}$ is the concentration of component j in the cloud-wake region. The final term $K_{g,bulk,j}$ represents the bulk flow of component j from the emulsion region into the bubble region.

In addition to partial differential equations, the model contains algebraic equations, including empirical correlations to describe the hydrodynamics, heat and mass transfer coefficient relations, gas phase properties including viscosity, thermal conductivity, heat capacity, empirical correlations to describe the cooling tubes within the adsorber, and detailed nonlinear reaction kinetics for reactions (6.24). In this work, the model is considered in steady-state mode (all time derivatives are set to zero). The resulting differential algebraic equations are discretized using collocation on finite elements. Full details of the model and discretization scheme can be found in [96, 97, 98].

The structure of the regenerator model is the same except with different boundary conditions described in [97]. This regenerator model is solved independently as a black box and the adsorber/regenerator system creates a process flowsheet where glass and black box portions interact through the cycling of the solid sorbent. The model is used for minimization of utility usage subject to a 48% CO_2 capture constraint. The size of the problem is given as follows: Black box inputs $n_w = 6$ black box outputs $n_y = 5$ and other variables $n_z = 5144$. The reduced models were constructed by linear interpolation.

Both Algorithms 1 and 2 were applied to the BFB system to compare their performance on a large-scale chemical process example. Algorithm 1 was stopped after 26 iterations because progress to the solution was too slow (step size below tolerance of 10^{-6} for two consecutive iterations). When Algorithm 2 was applied to the carbon capture system, the



(a) Algorithm 1

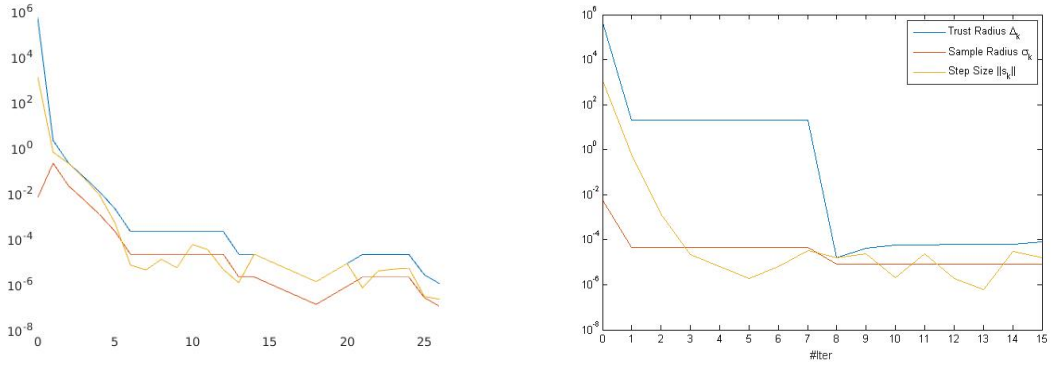
(b) Algorithm 2

Figure 6.5: Convergence metrics

optimal solution was found in 16 iterations. The performance of the two algorithms can be compared with Figures 6.5 and 6.6. Figure 6.5 focuses on the NLP convergence metrics, while Figure 6.6 focuses on the sequence of step sizes. In both figures, the abscissa represents the iteration count and the ordinate is the value for the metrics. All the values are shown in log scale for a better sense of the convergence behavior.

In Figure 6.5, we plot infeasibility measure θ_k, the criticality measure χ_k as defined in (4.7) and $|f(x_k) - f(x^*)|$ as a measure of the convergence of objective function. Both algorithms converge to feasibility relatively quickly, but Algorithm 1 slows down significantly and does not make much progress towards the optimal solution. The iterates stay within feasibility tolerance $\epsilon_\theta = 10^{-6}$ for most iterations.

In Figure 6.6, the trust region radius Δ_k, sample region radius σ_k and the step size $\|s_k\|$ are plotted to show the interactions. In Figure 6.6b, the sample radius is shown to successfully control the accuracy of the surrogate model while trust region allows larger step sizes. In many iterations, step size $\|s_k\|$ is much larger than the sample radius σ_k while still staying in the trust region Δ_k. The trust region constraint is only active at iteration



(a) Algorithm 1

(b) Algorithm 2

Figure 6.6: Trust and sample region radius shown together with step length $\|s_k\|$

8. This is because the algorithm takes mostly f-type steps (when very close to feasibility, f-type steps only require small decrease in the objective function). Since the trust region radius does not shrink on f-type steps, progress continues without updating the trust region. When the trust region is updated, for example at iteration 8, the trust region update is based on the step size. This results in a large change in the trust region radius in one iteration. The trust region update also requires the sample region to adjust so that the sample region lies within the trust region.

By contrast, Figure 6.6a shows that the step size in Algorithm 1 is restricted by the trust region radius. The "sample radius" in this case represents the perturbation size used to construct a surrogate model, and is directly tied to the trust region radius. Hence, the trust region constraint is consistently active in Algorithm 1, resulting in small step sizes and very slow convergence, as shown in Figure 6.5a for the objective function and criticality measure χ_k.

6.5 Surrogate Equations of State

Besides computationally expensive unit models, another common use case of surrogate models is for physical property models. Many equations of state exhibit several volume roots, which need to be either identified with their corresponding phases or ignored as physically irrelevant. Sequential modular approaches attempt to find all roots and use logical conditions to identify the phases, but the equation-oriented approach requires a customized model and implementation for the specific form of an equation of state. For cubic equations of state, [99] and [86] develop phase identification conditions using the sign of the first and second derivatives of the cubic equation. These conditions can then be included in the optimization models as inequality constraints to automatically enforce correct phase identification. However, for many fluids, such as polar or large-chain fluids, cubic equations of state are not always satisfactory. The IUPAC reference [100] reviews these methods in depth. However, when moving to more complex functional forms, the problem of phase identification becomes more serious. For example, the PC-SAFT equation can have up to five real roots for pure components [101]. Even outside of the equation-oriented context phase identification is non-trivial.

Sometimes, these issues have been avoided by using surrogate models of physical properties. However, these models are inherently local in nature, and the construction of suitable approximations becomes significantly more difficult as the number of components in a mixture increases. In addition, by independently fitting thermodynamic state variables, the laws of thermodynamics relating these state variables are often ignored, so extrapolation is limited. We propose instead to use surrogate equations of state (SEOS) fit within the trust region framework. This allows the properties derived from the surrogate equation of state to follow from basic laws of thermodynamics and have better extrapolation potential. We note that for this application, the key motivation of surrogate models is not for com-

putational efficiency, but rather to enable EO optimization technology for properties that typically are calculated procedurally.

It is desired to base the surrogate equations of state on well-known simple functional forms for equations of state. The most logical choices for the simple functional form are the virial equation and cubic equations of state. We will consider cubic equations with the Peng-Robinson form of the attractive term. This functional form will be tuned to match thermodynamic data (either generated from a complex equation of state or experimental data) in a local space, thus providing a surrogate equation of state. Of course, the simple functional form of the SEOS is not able to accurately describe all possible states. Instead, we fit over a local region of temperature, pressure, and concentration, and rely on the trust region filter approach introduced in 4 to control the error and guarantee accurate solutions.

6.5.1 Cubic Equations of State

The basic cubic functional form first proposed by van der Waals has been modified to produce many equation of state models. A review of these variations is given in [102], showing the great flexibility of cubic equations of state to predict the behavior of many systems. In this work, we will choose the Peng-Robinson equation of state functional form, although the methods can easily be used for other cubic equations of state.

The Peng Robinson equation of state for a pure component takes the form:

$$P = \frac{RT}{v-b} - \frac{\alpha a}{v^2 + 2bv - b^2} \tag{6.26}$$

where

$$\alpha = \left[1 + (0.37464 + 1.54226\omega - 0.26992\omega^2)\left(1 - \sqrt{\frac{T}{T_c}}\right)\right]^2 \tag{6.27}$$

was modified by Peng and Robinson from the SRK equation and ω is the Pitzer accentric factor.

Equation (6.26) can be reformulated to the standard cubic form in Z by multiplying through by the denominator and rearranging terms.

The parameters a and b in the PR equation (6.26) can be uniquely determined by imposing the criticality conditions. The criticality conditions state that at the critical point, we know that

$$\left(\frac{dP}{dv}\right)_{T_c} = \left(\frac{d^2P}{dv^2}\right)_{T_c} = 0 \tag{6.28}$$

Using these equations, we can show that

$$a = 0.45724\frac{R^2T_c^2}{P_c}$$

and

$$b = 0.07780\frac{RT_c}{P_c}$$

From fitting the critical conditions, we can see how the cubic equation of state follows the principle of corresponding states. This principle states that if the reduced conditions T/T_c and P/P_c and the accentric factor ω are the same, then the thermodynamic properties are (nearly) the same, regardless of the type of fluid. This means that a fluid is completely parameterized by its critical temperature, critical pressure, and accentric factor. This principle can be derived from molecular thermodynamics with assumptions including universal pairwise potential functions (see [103]). However, real fluids can vary in behavior depending on quantum effects, polarity, etc. To capture these deviations, we propose tuning the critical properties to fit the observed behavior of a fluid locally. For mixtures, we also tune binary interaction parameters $k_{i,j}$.

6.5.2 Parameter estimation formulation

In this section, a parameter estimation formulation for tuning cubic equations of state for vapor-liquid equilibrium calculations is proposed. Note that we redefine a few symbols

for ease of presentation, see Table 6.6 for details. Because cubic equations of state are parameterized by critical properties, we seek adjusted values for the pure component critical temperatures T_i, pure component critical pressures P_i, and binary interaction parameters $k_{i,j}$ to minimize the squared error in fugacity coefficient for one phase. This requires VLE data as input; for the purposes here the input data is generated by the PC-SAFT equation of state. The values associated with each data point k are system temperature $\hat{T}_k$, system pressure $\hat{P}_k$, composition $\hat{x}_{i,k}$, compressibility factor $\hat{Z}_k$, and fugacity coefficient $\hat{\phi}_{i,k}$. Note that the compositions, compressibility factor, and fugacity coefficient are specific to one phase, so the parameter estimation problem is separately applied to both liquid and vapor data. The subscript i represents the components and subscript k represents the data points. All symbols with a hat, for example $\hat{Z}_k$ are parameters associated with a data point. The parameter estimation model is presented as follows:

$$\min \quad \sum_k \sum_i (\hat{\phi}_{i,k} - \phi_{i,k})^2 + \rho(f(\hat{Z}_k))^2 \tag{6.29}$$

$$\text{s.t.} \quad a_{i,k} = 0.45724 \frac{m_{i,k} R^2 T_i^2}{P_i} \tag{6.30}$$

$$b_i = 0.07780 \frac{RT_i}{P_i} \tag{6.31}$$

$$m_{i,k} = \left(1 + (0.37464 + 1.54226\omega_i - 0.26992\omega_i^2)\left(1 - \sqrt{\frac{\hat{T}_k}{T_i}}\right)\right)^2 \tag{6.32}$$

$$a_k^m = \sum_i \sum_j \hat{x}_{i,k} \hat{x}_{j,k} \sqrt{a_{i,k} a_{j,k}} (1 - k_{i,j}) \tag{6.33}$$

$$b_k^m = \sum_i \hat{x}_{i,k} b_i \tag{6.34}$$

$$\alpha_k = \frac{a_k^m \hat{P}}{R^2 \hat{T}^2} \tag{6.35}$$

$$\beta_k = \frac{b_k^m \hat{P}}{R\hat{T}} \tag{6.36}$$

$$f(\hat{Z}_k) = \hat{Z}_k^3 + (\beta_k - 1)\hat{Z}_k^2 + (\alpha_k - 2\beta_k - 3\beta_k^2)\hat{Z}_k - \alpha_k\beta_k + \beta_k^2 + \beta_k^3 \tag{6.37}$$

$$\ln(\phi_{i,k}) = \frac{b_i}{b_k^m}(\hat{Z}_k - 1) - \ln(\hat{Z}_k - \beta_k) - \frac{\alpha_k}{2\sqrt{2}\beta_k}\left(\frac{2\sqrt{a_{i,k}}}{a_k^m}\left(\sum_j \hat{x}_{j,k}\sqrt{a_{j,k}}(1 - k_{i,j})\right) - \frac{b_i}{b_k^m}\right) \times \ln\left(\frac{\hat{Z}_k + (1 + \sqrt{2})\beta_k}{\hat{Z}_k + (1 - \sqrt{2})\beta_k}\right) \tag{6.38}$$

The objective function consists of two terms. The first term minimizes the squared relative deviation in fugacity coefficient, while the second term enforces the root of the cubic equation using a penalty parameter ρ. Note that the compressibility factors in the cubic equation are parameters given by data; it is the cubic equation itself that changes during optimization. If instead a constraint was added to enforce $f(\hat{Z}_k) = 0$ for all k, the problem

would be over-specified as there may be many more data k than degrees of freedom.

An alternative formulation could define a new variable Z_k for each datum k and add constraints to require each Z_k to be a root of the cubic equation at the corresponding temperature and pressure. However, this introduces the issue of root selection and the large number of nonlinear constraints can be difficult to converge. By contrast, the presented formulation can be solved as an unconstrained optimization problem through simple variable elimination and is found to be quite robust. Although the problem is nonconvex and the existence of several local minima has been observed, very good initialization can be derived by using the literature-reported critical temperatures, pressures, and binary interaction parameters. Formulation (6.29) has been implemented in Pyomo, and the parameter estimation problem is solved using IPOPT.

Note that the objective function of (4.2) can be re-scaled as follows, which can give better performance in practice:

$$\min \quad \sum_k \sum_i (1 - \phi_{i,k}/\hat{\phi}_{i,k})^2 + \rho(f(\hat{Z}_k))^2$$

6.5.3 Numerical testing

To use cubic equations of state as surrogates in an optimization framework, they must be sufficiently accurate to recover correct solutions. With the aim of showing the κ-fully linear property can hold, several empirical numerical tests are run with using cubic equations in place of detailed thermodynamics. A case study is taken from the polyethylene production process. This flash model is used to separate unreacted monomer (ethylene) and chain transfer agent (hydrogen) from solvent (hexane) for recycle to the reactor. VLE data were gathered using a PC-SAFT flash model with a 3-level full factorial design on temperature, pressure, and component flowrates. The levels of each variable are shown in Table 6.7.

Table 6.6: Nomenclature Table

Symbol	Explanation
$Z = PV/RT$	Compressibility factor
T_i	Critical temperature of component i
P_i	Critical pressure of component i
$k_{i,j}$	Binary interaction parameters
$f(Z)$	Cubic equation of state
ϕ_i	Fugacity coefficient for component i

Out of the 243 points in the full factorial design, two fell outside of the two-phase region and were discarded. The flash model outputs both liquid and vapor phase concentrations, compressibility factors, flowrates, and fugacity coefficients. Formulation (6.29) was applied to fit SEOS for both liquid and vapor phases. The root error fitting parameter ρ was set to 100000. Both problems were initialized with the literature values for critical properties and binary interaction coefficients. In the liquid phase, there was insufficient information in the data (degeneracy in the reduced Hessian) to obtain unique values for hydrogens critical properties, so these were fixed at their literature values. The resulting parameters are shown in Table 6.8.

Figure 6.7 compares the fugacity coefficients varying with temperature for PC-SAFT, the SEOS, and untuned Peng-Robinson when the pressure is fixed at 3.85 bar. The SEOS matches very well over the temperature range, but the untuned PR deviates significantly. Another comparison was made with the flash temperature kept constant at 378K and the pressure was varied. This result is shown in Figure 6.8

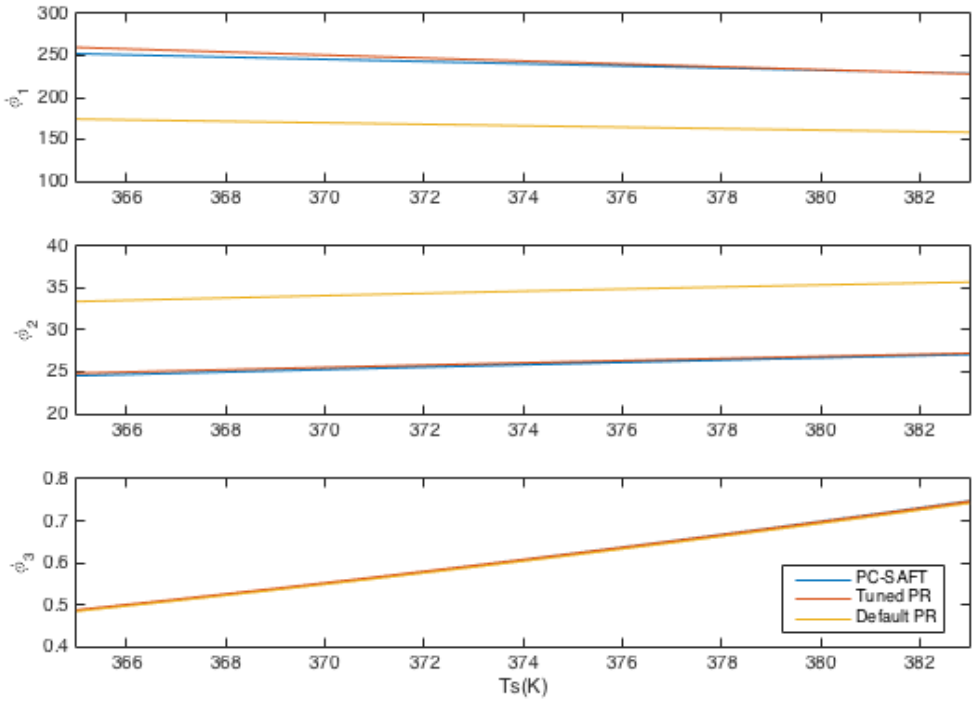


Figure 6.7: Change in liquid fugacity coefficient as flash temperature changes

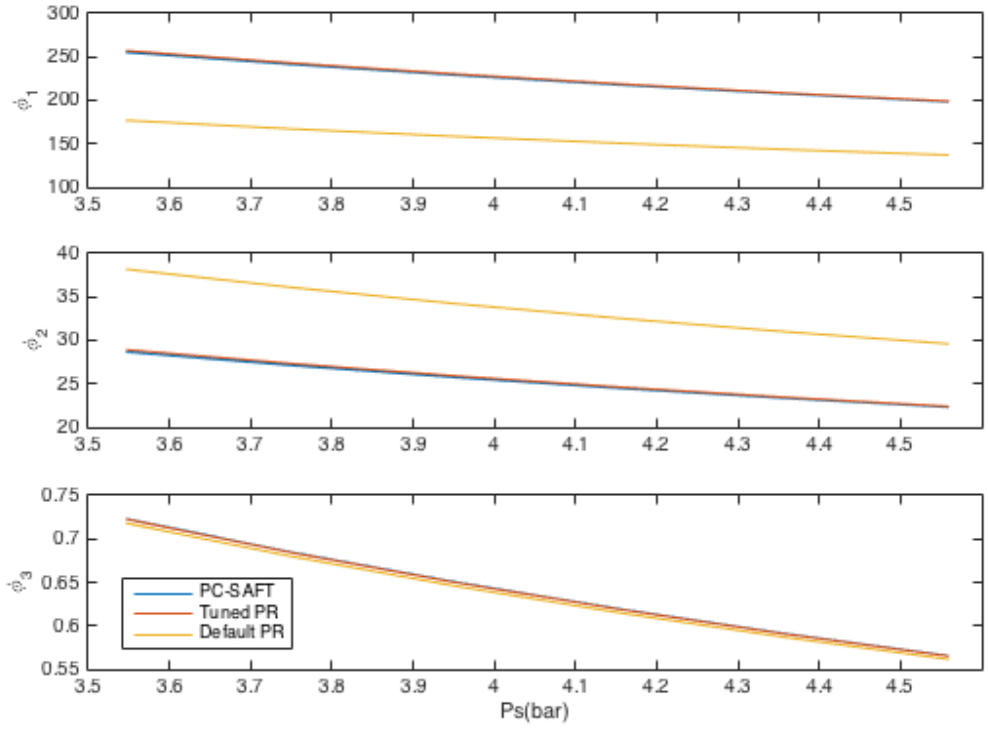


Figure 6.8: Change in liquid fugacity as flash pressure changes

Table 6.7: Full factorial design on flash input conditions

Input variable	Values	Explanation
$\hat{T}_k$	[362.15, 372.15, 382.15]	Flash temperature (K)
$\hat{P}_k$	[3.5, 4.0, 4.5]	Flash pressure (bar)
F_1	[0.093, 0.186, 0.373]	Input flowrate hydrogen
F_2	[3.01, 6.02, 12.05]	Input flowrate ethylene
F_3	[112.8, 141.0, 169.2]	Input flowrate hexane

6.5.4 Flash model

The resulting surrogate equations of state are used in an optimization formulation of a flash drum. The objective is to minimize the amount of monomer in the liquid phase while maintaining a lower bound on mass flow of this stream such that the polymer product (not included in equilibrium calculations) remains a slurry. The model is given as:

$$\min_{T,P} \quad x_{C2H4} L \tag{6.39}$$

$$s.t. \quad \tilde{F} = L + V \tag{6.40}$$

$$\tilde{F}\tilde{z}_i = L x_i + V y_i \tag{6.41}$$

$$0 = \sum_i (y_i - x_i) \tag{6.42}$$

$$\sum_i MW_i x_i \geq c \tag{6.43}$$

$$y_i = \frac{\phi_{L,i}}{\phi_{V,i}} x_i \tag{6.44}$$

where MW_i is the molecular weight of component i and c is a constant to keep the product polymer suspended in slurry. Feed flowrate $\tilde{F}$ and feed composition $\tilde{z}$ are fixed parame-

Table 6.8: Results of fitting SEOS

Parameter	Literature	Liquid fit	Vapor fit
T_{H2} (K)	33.0	33.0	76.54
T_{C2H4} (K)	283	347.5	342.3
$T_{n-hexane}$ (K)	507.5	506.9	649.7
P_{H2} (bar)	13.2	13.2	10.1
P_{C2H4} (bar)	51.2	62.4	84.8
$P_{n-hexane}$ (bar)	30.1	30.0	61.9
$k_{H2,C2H4}$	0.022	2.8×10^{-6}	0.509
$k_{H2,n-hexane}$	0.02917	0.513	0.972
$k_{2H4,n-hexane}$	0.1144	0.179	0.158

ters. $\phi_{L,i}$ and $\phi_{V,i}$ are functions as follows:

$$\phi_{L_i} := \phi_{L_i}(T, P, x_i, T_{L,i}, P_{L,i}, k^L_{i,j})$$

$$\phi_{V_i} := \phi_{V_i}(T, P, y_i, T_{V,i}, P_{V,i}, k^V_{i,j})$$

T, P are the system temperature and pressure (decision variables for the optimization). $T_{L,i}, P_{L,i}, k^L_{i,j}, T_{V,i}, P_{V,i}, k^V_{i,j}$ are the tuned parameters for the surrogate equations of state, and Z_L and Z_V are the compressibility factors. The value of $\phi_{V,i}$ and ϕ_{L_i} can be determined by solving the following system of equations. In fact, these equations can be substituted out to form one large expression, so the functional relationship to determine the ϕ values is well defined.

$$a_{L,i} = 0.45724 \frac{m_{L,i} R^2 T_{L,i}^2}{P_{L,i}} \tag{6.45}$$

$$a_{V,i} = 0.45724 \frac{m_{V,i} R^2 T_{V,i}^2}{P_{V,i}} \tag{6.46}$$

$$b_{L,i} = 0.07780 \frac{R T_{L,i}}{P_{L,i}} \tag{6.47}$$

$$b_{V,i} = 0.07780 \frac{R T_{V,i}}{P_{V,i}} \tag{6.48}$$

$$m_{L,i} = \left(1 + (0.37464 + 1.54226\omega_i - 0.26992\omega_i^2)\left(1 - \sqrt{\frac{\tilde{T}}{T_{L,i}}}\right)\right)^2 \tag{6.49}$$

$$m_{V,i} = \left(1 + (0.37464 + 1.54226\omega_i - 0.26992\omega_i^2)\left(1 - \sqrt{\frac{\tilde{T}}{T_{V,i}}}\right)\right)^2 \tag{6.50}$$

$$a_L^m = \sum_i \sum_j x_i x_j \sqrt{a_{L,i} a_{L,j}} (1 - k_{i,j}^L) \tag{6.51}$$

$$a_V^m = \sum_i \sum_j y_i y_j \sqrt{a_{V,i} a_{V,j}} (1 - k_{i,j}^V) \tag{6.52}$$

$$b_L^m = \sum_i x_i b_{L,i} \tag{6.53}$$

$$b_V^m = \sum_i y_i b_{V,i} \tag{6.54}$$

$$\tag{6.55}$$

$$\ln(\phi_{L,i}) = \frac{b_{L,i}}{b_L^m}(Z_L - 1) - \ln(Z_L - \beta_L) - \frac{\alpha_L}{2\sqrt{2}\beta_L}\left(\frac{2\sqrt{a_{L,i}}}{a_L^m}\left(\sum_j x_j \sqrt{a_{L,j}}(1 - k_{i,j}^L)\right) - \frac{b_{L,i}}{b_L^m}\right) \times \ln\left(\frac{Z_L + (1+\sqrt{2})\beta_L}{Z_L + (1-\sqrt{2})\beta_L}\right) \tag{6.56}$$

$$\ln(\phi_{V,i}) = \frac{b_{V,i}}{b_V^m}(Z_V - 1) - \ln(Z_V - \beta_V) - \frac{\alpha_V}{2\sqrt{2}\beta_V}\left(\frac{2\sqrt{a_{V,i}}}{a_V^m}\left(\sum_j y_j \sqrt{a_{V,j}}(1 - k_{i,j}^V)\right) - \frac{b_{V,i}}{b_V^m}\right) \times \ln\left(\frac{Z_V + (1+\sqrt{2})\beta_V}{Z_V + (1-\sqrt{2})\beta_V}\right) \tag{6.57}$$

$$\alpha_L = \frac{a_L^m \tilde{P}}{R^2 \tilde{T}^2} \tag{6.58}$$

$$\alpha_V = \frac{a_V^m \tilde{P}}{R^2 \tilde{T}^2} \tag{6.59}$$

$$\beta_L = \frac{b_L^m \tilde{P}}{R\tilde{T}} \tag{6.60}$$

$$\beta_V = \frac{b_V^m \tilde{P}}{R\tilde{T}} \tag{6.61}$$

$$0 = Z_L^3 + (\beta_L - 1)Z_L^2 + (\alpha_L - 2\beta_L - 3\beta_L^2)Z_L - \alpha_L\beta_L + \beta_L^2 + \beta_L^3 \tag{6.62}$$

$$0 = Z_V^3 + (\beta_V - 1)Z_V^2 + (\alpha_V - 2\beta_V - 3\beta_V^2)Z_V - \alpha_L\beta_V + \beta_V^2 + \beta_V^3 \tag{6.63}$$

$$0 \leq 3Z_L^2 + 2(\beta_L - 1)Z_L + \alpha_L - 2\beta_L - 3\beta_L^2 \tag{6.64}$$

$$0 \leq 3Z_V^2 + 2(\beta_V - 1)Z_V + \alpha_V - 2\beta_V - 3\beta_V^2 \tag{6.65}$$

$$0 \geq 6Z_L + 2(\beta_L - 1) \tag{6.66}$$

$$0 \leq 6Z_V + 2(\beta_V - 1) \tag{6.67}$$

Optimization problem (6.39)-(6.44) was run in three different modes: using un-tuned Peng-Robinson (the above model with literature critical properties and binary interaction parameters), the tuned SEOS model, and a model from [104] where the fugacity coefficients are calculated with PC-SAFT directly. We note however that the PC-SAFT equation of state is difficult to use in equation-oriented optimization due to the presence of multiple volume roots. Through careful initialization, a result was obtained that was validated against Aspen Plus to have found the correct roots of PC-SAFT.

Table 6.9 shows that the results of using Peng-Robinson and PC-SAFT can vary by as much as 36%. Using the SEOS parameters shown in Table 6.8, the maximum error relative to PC-SAFT at the optimal solution is reduced to 6%. Then, more data were gathered from PC-SAFT, in a full factorial design centered at the solution of the first SEOS optimization problem, with the width of data range cut in half. This simulates the behavior of the trust region subproblem, where a smaller trust region should increase optimization accuracy.

Table 6.9: Flash optimization results, errors are measured relative to PC-SAFT

	PC-SAFT	PR	Error	SEOS	Error	Re-tuned SEOS	Error
x_{H2}	4.6210-5	6.310-5	36%	4.3410-5	6%	4.6210-5	0.03%
x_{C2H4}	0.0127	0.0974	23%	0.0125	1.86%	0.0127	0.04%
x_{hexane}	0.9872	0.9902	0.3%	0.9875	0.02%	0.9873	<0.01%
T	351.67	349.68	0.6%	351.38	0.08%	351.64	<0.01%
P	2.5	2.5	0%	2.5	0%	2.5	0%

Then the SEOS is re-tuned using the data in the region of the solution, and the maximum error is reduced to 0.04%. This indicates the potential of using automatic update strategies in a trust region framework to control the optimization error introduced by the SEOS approximation.

These results demonstrate the potential of using a cubic equation of state to locally represent thermodynamic behavior directly from data. Tuning parameters for the cubic equation are selected based on their interpretation in terms of the principle of corresponding states. Then, an effective and reliable parameter estimation formulation is proposed. The formulation can be naturally converted to an unconstrained optimization problem for ease of initialization and solution. Finally, SEOS models are demonstrated for optimization of a flash unit in a polymerization system, and the SEOS gives accurate results with the potential for refinement by gathering more data.

More broadly, this suggests that the framework of κ-fully linear models with trust regions may be applicable outside of the typical domain of polynomial approximations of black box functions. As presented, using cubic equations of state and a parameter estimation formulation, the κ-fully linear prop The SEOS concept is not restricted to cubic functional forms and it may be worthwhile to consider virial-type equations as well. In

addition, it is important to consider what theoretical guarantees can be made on the SEOS. One approach could be through the use of thermodynamics. It may be possible to posit functional forms for equations of state that allow for first-order consistent matching of a thermodynamic property of interest. This can guarantee that the surrogate model is κ-fully linear and then the models can be freely used in the trust region framework.

6.6 Conclusions

In this chapter, we have summarized our experiences applying Algorithms 1 and 2 to practical problems in engineering. The Williams-Otto problem allowed us to compare the glass box/black box approach via the trust region filter method against other optimization approaches for this problem. The ammonia process further verified that the algorithm converges quickly and reliably on larger problems. Then, the power plant optimization case study applied the TRF method to a problem that could not be handled with previously existing optimization solvers. Integrating the detailed boiler model with process models through the TRF method allowed for significant efficiency improvements in the oxycombustion power plant. Next, a solid sorbent carbon capture process was used to demonstrate the improvements of Algorithm 2. Finally, the extension of this framework to physical properties is discussed. The complexity of physical property models is often a barrier to building and using optimization models. Surrogate models combined with a trust region are shown to be promising in reducing model building complexity and optimization robustness.

Chapter 7

Conclusions

This thesis can be summarized as follows. In Chapter 1, the concept of glass box / black box optimization is introduced along with motivation in engineering. A common bottleneck in applying large-scale optimization is the near-requirement that models be conceived in equation oriented form and implemented in algebraic modeling languages. When this is true, modern automatic differentiation methods provide accurate first and second derivatives. Nonlinear programming solvers may utilize these derivatives and exploit sparsity to provide fast and accurate optimization.

However, equation oriented representations are not always available for all models in a system. Examples of non-EO models include thermodynamic function calls, models of proprietary unit operations, or inherently procedural models. Ideally these closed models could be opened, i.e. transformed into an equation-based representation and included in the EO process model. For example, partial differential algebraic equations can be discretized and transformed into algebraic equations. But often this transformation is not possible or not desirable. Moreover, highly specialized (often procedural) methods exist for solving computational fluid dynamics or molecular simulation problems. For reliable and accurate solutions, these systems must be solved outside of the optimization modeling framework, and derivative information is often unavailable. This gives rise to a class of problems we term glass box/black box optimization, or gray-box optimization.

In Chapters 2 and 3, we review background material about nonlinear optimization and

surrogate modeling. Past efforts in the realm of surrogate-based optimization are reviewed. Researchers have approached this question from several angles. Applied mathematicians working in the field of numerical optimization are currently interested in the topic of constrained derivative free optimization. A good method for constrained DFO should be able to solve a glass box / black box problem. However, these works normally ignore the potential to exploit the glass box structure and speed practical solution methods.

On the other hand, engineers often encounter problems that could be solved with this type of optimization. The most common approach in practice is the use of surrogate models, or reduced order models, to transform the glass box / black box problem into a fully equation oriented form suitable for NLP solvers. Aerospace engineers approach from the direction of multi-fidelity optimization, where a large scale PDE model is discretized at both coarse and fine resolution. The coarse grain discretization can act as a surrogate for the fine grain, and correction terms may be added to help performance in optimization. Chemical engineers often face this problem when dealing with specialized unit operation models. Reactor models may be tailor-made for a process, and the reaction and transport equations may require specialized simulation techniques. These simulations may be treated as black boxes, and through computational experiment data may be generated and surrogate models fit. Popular forms include polynomials, neural networks, and Kriging and Gaussian process regression.

However, problems are often encountered when using surrogate models in optimization as the optimizer can exploit errors in the surrogate to artificially improve the objective. The most common approach to regulate this behavior is through the use of trust region methods. However, past efforts usually required derivative information from the black box to verify convergence properties. Finally, concepts from derivative free optimization were reviewed. The κ-fully linear property provides a general yet powerful method to drive

convergence of surrogate-based optimization without explicit access to derivative information. However, few works have considered the use of κ-fully linear surrogate models in constrained derivative free optimization or glass box / black box optimization.

To address this gap in the literature, a novel trust region filter algorithm for glass box / black box optimization was presented in Chapter 4. The algorithm is based on trust region filter methods from sequential quadratic programming, except the quadratic programming subproblems are replaced by surrogate-based nonlinear optimization problems. The main contributions are as follows:

1. Proposed a novel trust region filter method for glass box / black box optimization
2. Proved convergence, using the original trust region filter convergence proof modified to support κ-fully linear models through use of a unique criticality phase.

In Chapter 5, the practical behavior of the trust region filter method is examined in more detail. Several algorithmic modifications/enhancements are introduced and numerical performance is tested. The main contributions are as follows:

1. Proposed the sampling region concept for the trust region filter. By not requiring the trust region to tend to zero, behavior of the algorithm is improved near the solution.
2. Modified step acceptance and trust region update criteria. This helps keep the trust region at an appropriate radius such that the algorithm quickly recovers after one rejected step.
3. Implemented TRF algorithm in Pyomo. The software automatically locates and replaces external function calls with surrogate models and applies the TRF algorithm.
4. Developed a test problem set for glass box / black box optimization. These 63 problems may be used to compare the TRF code against alternative methods.
5. Results show the advantage of the sampling region, modified step acceptance, and trust region update criteria.

In Chapter 6, the trust region code was applied to a variety of case studies. These studies are designed to both examine the performance of the algorithm as well as demonstrate its capability to solve practical problems. Key results are:

1. Demonstrated the benefit of a specialized glass box / black box solver over a fully DFO approach on the Williams-Otto flowsheet.
2. Showed fast convergence on a full-scale process optimization of the ammonia process.
3. Successfully optimized a large-scale power plant model in both carbon capture and conventional configurations.
4. Identified oxycombustion systems design reducing the carbon capture efficiency penalty to 5.7%.
5. Analyzed the effect of the sampling region on a solid-sorbent carbon capture process optimization problem.
6. Proposed the use of trust region concepts with physical property calculations through the framework of surrogate equations of state.

7.1 Recommendations for Future Work

In this section, we conclude with thoughts on interesting directions for future work.

7.1.1 Multiple black boxes

The first direction of future work would be to address the case when the black box function $d(w)$ is in fact made of several black box functions $d_i(w_i)$. Each of these black boxes may be scaled differently, and different surrogate construction strategies should be applied to each. This impacts the geometry of samples, because now samples need to be poised in

subspaces rather than the full space $\mathbb{R}^m$. This also suggests the use of separate trust regions and sampling regions to govern each black box d_i.

A theoretical framework to accomplish this would be through the use of uniformly equivalent norms [78]. This means that there exists an $\alpha \geq 1$ such that for all k, $\max_i \Delta_{i,k} \leq \alpha \min_i \Delta_{i,k}$. If this condition is satisfied, and as long as $\lim_{k\to\infty} \sigma_{i,k} = 0$, there is considerable flexibility to design an effective update strategy.

7.1.2 Building SEOS

The concept of a surrogate equation of state, introduced in Chapter 6, has great potential to ease the effort in building thermodynamic models for optimization. Through allowing a fixed functional form to be tuned to data, the task of formulating the thermodynamic model only needs to be completed once and then many systems can easily be handled. However, the question of what that functional form should be remains somewhat open. This thesis used cubic equations of state for simplicity, but there may be thermodynamic justification for another functional form.

The κ-fully linear property can be considered from the thermodynamic perspective. In other words, the effect of the equation of state on the various properties and their partial derivatives can be considered in such a way that there is some assurance that the SEOS can be tuned sufficiently well to match the first order behavior of a physical property of interest.

7.1.3 Noise in function calls

One of the greatest challenges areas for future work in this field is on the question of noise in the black box calls. Optimization of noisy black box were thoroughly reviewed in [106]. However, little work has been done extending this to the context of constrained black box

optimization. The concept of including a stochastic function in a constraint is not on the surface well-posed. Statements need to be made about whether the constraint should hold in expectation, or perhaps at a confidence level. With additional inputs from the DFO community in this area (see e.g. [107]), this is an active area of research. However, the appropriate strategies do seem highly dependent on the application at hand.

A closely related but distinct challenge is that of "deterministic" noise. This is seemingly noisy behavior caused by numerical tolerances used in black box simulations. In this thesis, black box simulations were modified to tighten internal tolerances so that a relatively smooth response was obtained. However, this may not always be possible in practice. Smoothing or weighted regression methods may be considered to try to find approximations of the smooth function underlying the numerical noise.

Bibliography

[1] F. Gould and J. W. Tolle, "A necessary and sufficient qualification for constrained optimization," *SIAM Journal on Applied Mathematics*, vol. 20, no. 2, pp. 164–172, 1971.

[2] L. T. Biegler, *Nonlinear programming: concepts, algorithms, and applications to chemical processes*. Philadelphia: SIAM, 2010.

[3] A. Forsgren, P. E. Gill, and M. H. Wright, "Interior methods for nonlinear optimization," *SIAM review*, vol. 44, no. 4, pp. 525–597, 2002.

[4] R. Andreani, J. M. Martinez, A. Ramos, and P. J. Silva, "A cone-continuity constraint qualification and algorithmic consequences," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 96–110, 2016.

[5] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[6] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization*. Academic press, 1981.

[7] W. Murray, "Sequential quadratic programming methods for large-scale problems," *Computational Optimization and Applications*, vol. 7, no. 1, pp. 127–142, 1997.

[8] A. V. Fiacco and G. P. McCormick, *Nonlinear programming: sequential unconstrained minimization techniques*. SIAM, 1990.

[9] J. E. Dennis Jr and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, vol. 16. SIAM, 1996.

[10] R. Fletcher and S. Leyffer, "Nonlinear programming without a penalty function," *Mathematical programming*, vol. 91, no. 2, pp. 239–269, 2002.

[11] A. Drud, "Conopt: A GRG code for large sparse dynamic nonlinear optimization problems," *Mathematical Programming*, vol. 31, no. 2, pp. 153–191, 1985.

[12] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[13] R. H. Byrd, J. Nocedal, and R. A. Waltz, "Knitro: An integrated package for nonlinear optimization," in *Large-scale nonlinear optimization*, pp. 35–59, Springer, 2006.

[14] R. H. Byrd, M. E. Hribar, and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 877–900, 1999.

[15] A. R. Conn, G. Gould, and P. L. Toint, *LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A)*, vol. 17. Springer Science & Business Media, 2013.

[16] R. J. Vanderbei, "LOQO: An interior point code for quadratic programming," *Optimization methods and software*, vol. 11, no. 1-4, pp. 451–484, 1999.

[17] B. A. Murtagh and M. A. Saunders, "Minos 5.0 user's guide.," tech. rep., STANFORD UNIV CA SYSTEMS OPTIMIZATION LAB, 1983.

[18] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, "User's guide for npsol (version 4.0): A fortran package for nonlinear programming.," tech. rep., STANFORD UNIV CA SYSTEMS OPTIMIZATION LAB, 1986.

[19] P. Gill, W. Murray, and M. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[20] D. M. Gay, "Hooking your solver to AMPL," tech. rep., Technical Report 93-10, AT&T Bell Laboratories, Murray Hill, NJ, 1993, revised, 1997.

[21] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[22] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to derivative-free optimization*. Philadelphia: SIAM, 2009.

[23] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[24] M. J. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in optimization and numerical analysis*, pp. 51–67, Springer, 1994.

[25] C. Audet and J. Dennis, Jr., "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 188–217, 2006.

[26] C. Audet, S. Le Digabel, and C. Tribes, "NOMAD user guide," Tech. Rep. G-2009-37, Les cahiers du GERAD, 2009.

[27] P. R. Sampaio and P. L. Toint, "A derivative-free trust-funnel method for equality-constrained nonlinear optimization," *Computational Optimization and Applications*, vol. 61, no. 1, pp. 25–49, 2015.

[28] F. Boukouvala, R. Misener, and C. A. Floudas, "Global optimization advances in mixed-integer nonlinear programming, minlp, and constrained derivative-free opti-

mization, cdfo," *European Journal of Operational Research*, vol. 252, no. 3, pp. 701–727, 2016.

[29] C. A. Floudas, A. M. Niziolek, O. Onel, and L. R. Matthews, "Multi-scale systems engineering for energy and the environment: Challenges and opportunities," *AIChE Journal*, vol. 62, no. 3, pp. 602–623, 2016.

[30] Y.-d. Lang, A. Malacina, L. T. Biegler, S. Munteanu, J. I. Madsen, and S. E. Zitney, "Reduced order model based on principal component analysis for process simulation and optimization," *Energy & Fuels*, vol. 23, no. 3, pp. 1695–1706, 2009.

[31] A. Barrett and J. Walsh, "Improved chemical process simulation using local thermodynamic approximations," *Computers & Chemical Engineering*, vol. 3, no. 14, pp. 397 – 402, 1979.

[32] J. Boston and H. Britt, "A radically different formulation and solution of the single-stage flash problem," *Computers & Chemical Engineering*, vol. 2, no. 2-3, pp. 109–122, 1978.

[33] M. Leesley and G. Heyen, "The dynamic approximation method of handling vapor-liquid equilibrium data in computer calculations for chemical processes," *Computers & Chemical Engineering*, vol. 1, no. 2, pp. 103–108, 1977.

[34] A. C. Antoulas, D. C. Sorensen, and S. Gugercin, "A survey of model reduction methods for large-scale systems," *Contemporary mathematics*, vol. 280, pp. 193–220, 2001.

[35] W. Marquardt, "Nonlinear model reduction for optimization based control of transient chemical processes," in *AIChE Symposium Series*, pp. 12–42, New York; American Institute of Chemical Engineers; 1998, 2002.

[36] T. Bui-Thanh, M. Damodaran, and K. E. Willcox, "Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition," *AIAA journal*, vol. 42, no. 8, pp. 1505–1516, 2004.

[37] M. Couplet, C. Basdevant, and P. Sagaut, "Calibrated reduced-order POD-Galerkin system for fluid flow modelling," *Journal of Computational Physics*, vol. 207, no. 1, pp. 192–220, 2005.

[38] B. Galletti, C. Bruneau, L. Zannetti, and A. Iollo, "Low-order modelling of laminar flow regimes past a confined square cylinder," *Journal of Fluid Mechanics*, vol. 503, pp. 161–170, 2004.

[39] H. V. Ly and H. T. Tran, "Modeling and control of physical processes using proper orthogonal decomposition," *Mathematical and computer modelling*, vol. 33, no. 1-3, pp. 223–236, 2001.

[40] J. Moehlis, T. Smith, P. Holmes, and H. Faisst, "Models for turbulent plane couette flow using the proper orthogonal decomposition," *Physics of Fluids*, vol. 14, no. 7, pp. 2493–2507, 2002.

[41] H. Park and D. Cho, "The use of the karhunen-loeve decomposition for the modeling of distributed parameter systems," *Chemical Engineering Science*, vol. 51, no. 1, pp. 81–98, 1996.

[42] S. Shvartsman, C. Theodoropoulos, R. Rico-Martınez, I. Kevrekidis, E. Titi, and T. Mountziaris, "Order reduction for nonlinear dynamic models of distributed reacting systems," *Journal of Process Control*, vol. 10, no. 2-3, pp. 177–184, 2000.

[43] K. Willcox and J. Peraire, "Balanced model reduction via the proper orthogonal decomposition," *AIAA journal*, vol. 40, no. 11, pp. 2323–2330, 2002.

[44] T. Yuan, P. Cizmas, and T. O'Brien, "A reduced-order model for a bubbling fluidized bed based on proper orthogonal decomposition," *Computers & chemical engineering*, vol. 30, no. 2, pp. 243–259, 2005.

[45] T. A. Brenner, R. L. Fontenot, P. G. Cizmas, T. J. OBrien, and R. W. Breault, "A reduced-order model for heat transfer in multiphase flow and practical aspects of the proper orthogonal decomposition," *Computers & Chemical Engineering*, vol. 43, pp. 68–80, 2012.

[46] T. W. Simpson, J. Poplinski, P. N. Koch, and J. K. Allen, "Metamodels for computer-based engineering design: survey and recommendations," *Engineering with computers*, vol. 17, no. 2, pp. 129–150, 2001.

[47] R. H. Myers, D. C. Montgomery, G. G. Vining, C. M. Borror, and S. M. Kowalski, "Response surface methodology: a retrospective and literature survey," *Journal of quality technology*, vol. 36, no. 1, p. 53, 2004.

[48] J. P. Kleijnen, "Kriging metamodeling in simulation: A review," *European journal of operational research*, vol. 192, no. 3, pp. 707–716, 2009.

[49] J. Eason and S. Cremaschi, "Adaptive sequential sampling for surrogate model generation with artificial neural networks," *Computers & Chemical Engineering*, 2014.

[50] S. Razavi, B. A. Tolson, and D. H. Burn, "Review of surrogate modeling in water resources," *Water Resources Research*, vol. 48, no. 7, 2012.

[51] A. Bhosekar and M. Ierapetritou, "Advances in surrogate based modeling, feasibility analysis and and optimization: A review," *Computers & Chemical Engineering*, 2017.

[52] A. Cozad, N. V. Sahinidis, and D. C. Miller, "Learning surrogate models for simulation-based optimization," *AIChE Journal*, vol. 60, no. 6, pp. 2211–2227, 2014.

[53] Z. T. Wilson and N. V. Sahinidis, "The alamo approach to machine learning," *Computers & Chemical Engineering*, vol. 106, pp. 785–795, 2017.

[54] L. T. Biegler, I. E. Grossmann, and A. W. Westerberg, "A note on approximation techniques used for process optimization," *Computers & Chemical Engineering*, vol. 9, no. 2, pp. 201–206, 1985.

[55] L. T. Biegler, Y. Lang, and W. Lin, "Multi-scale optimization for process systems engineering," *Computers & Chemical Engineering*, vol. 60, pp. 17–30, 2014.

[56] N. M. Alexandrov, J. E. Dennis Jr, R. M. Lewis, and V. Torczon, "A trust-region framework for managing the use of approximation models in optimization," *Structural Optimization*, vol. 15, no. 1, pp. 16–23, 1998.

[57] A. A. Giunta and M. S. Eldred, "Implementation of a trust region model management strategy in the dakota optimization toolkit," in *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA*, 2000.

[58] E. Arian, M. Fahl, and E. W. Sachs, "Trust-region proper orthogonal decomposition for flow control," Tech. Rep. ICASE Report No. 2000-25, Institute for Computer Applications in Science and Engineering, 2000.

[59] S. M. Wild, R. G. Regis, and C. A. Shoemaker, "ORBIT: Optimization by radial basis function interpolation in trust-regions," *SIAM Journal on Scientific Computing*, vol. 30, no. 6, pp. 3197–3219, 2008.

[60] A. March and K. Willcox, "Provably convergent multifidelity optimization algorithm not requiring high-fidelity derivatives," *AIAA journal*, vol. 50, no. 5, pp. 1079–1089, 2012.

[61] J. A. Caballero and I. E. Grossmann, "An algorithm for the use of surrogate models in modular flowsheet optimization," *AIChE journal*, vol. 54, no. 10, pp. 2633–2650, 2008.

[62] C. A. Henao and C. T. Maravelias, "Surrogate-based process synthesis," *Computer Aided Chemical Engineering*, vol. 28, pp. 1129–1134, 2010.

[63] C. A. Henao and C. T. Maravelias, "Surrogate-based superstructure optimization framework," *AIChE Journal*, vol. 57, no. 5, pp. 1216–1232, 2011.

[64] A. Agarwal and L. T. Biegler, "A trust-region framework for constrained optimization using reduced order modeling," *Optimization and Engineering*, vol. 14, no. 1, pp. 3–35, 2013.

[65] F. Boukouvala, M. F. Hasan, and C. A. Floudas, "Global optimization of general constrained grey-box models: new method and its application to constrained PDEs for pressure swing adsorption," *Journal of Global Optimization*, pp. 1–40, 2015.

[66] E. Davis and M. Ierapetritou, "A kriging method for the solution of nonlinear programs with black-box functions," *AIChE Journal*, vol. 53, no. 8, pp. 2001–2012, 2007.

[67] E. Davis and M. Ierapetritou, "A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions," *Journal of Global Optimization*, vol. 43, no. 2-3, pp. 191–205, 2009.

[68] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: a review of algorithms and comparison of software implementations," *Journal of Global Optimization*, vol. 56, no. 3, pp. 1247–1293, 2013.

[69] A. R. Conn, K. Scheinberg, and P. L. Toint, "Recent progress in unconstrained non-

linear optimization without derivatives," *Mathematical programming*, vol. 79, no. 1-3, p. 397, 1997.

[70] M. J. Powell, "UOBYQA: unconstrained optimization by quadratic approximation," *Mathematical Programming*, vol. 92, no. 3, pp. 555–582, 2002.

[71] M. J. Powell, "The newuoa software for unconstrained optimization without derivatives," in *Large-scale nonlinear optimization*, pp. 255–297, Springer, 2006.

[72] R. Fletcher, N. I. Gould, S. Leyffer, P. L. Toint, and A. Wächter, "Global convergence of a trust-region SQP-filter algorithm for general nonlinear programming," *SIAM Journal on Optimization*, vol. 13, no. 3, pp. 635–659, 2002.

[73] A. Walther and L. Biegler, "On an inexact trust-region SQP-filter method for constrained nonlinear optimization," *Computational Optimization and Applications*, pp. 1–26, 2014.

[74] S. M. Wild, *Derivative-free optimization algorithms for computationally expensive functions*. PhD thesis, Cornell University, 2009.

[75] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer New York, second ed., 1999.

[76] A. V. Fiacco, *Introduction to sensitivity and stability analysis in nonlinear programming*. Elsevier, 1983.

[77] S. M. Robinson, *Generalized equations and their solutions, Part II: Applications to nonlinear programming*. Springer, 1982.

[78] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust region methods*. SIAM, 2000.

[79] W. E. Hart, J.-P. Watson, and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in Python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

[80] W. E. Hart, C. Laird, J.-P. Watson, and D. L. Woodruff, *Pyomo–optimization modeling in Python*, vol. 67. Springer Science & Business Media, 2012.

[81] N. I. Gould, D. Orban, and P. L. Toint, "CUTEr and SifDec: A constrained and unconstrained testing environment, revisited," *ACM Transactions on Mathematical Software (TOMS)*, vol. 29, no. 4, pp. 373–394, 2003.

[82] E. D. Dolan, J. J. Moré, and T. S. Munson, "Benchmarking optimization software with COPS 3.0.," tech. rep., Argonne National Lab., Argonne, IL (US), 2004.

[83] D. P. Word, *Nonlinaer programming approaches for efficient large-scale parameter estimation with applications in epidemiology*. PhD thesis, Texas A& M University, 2013.

[84] M. Čižniar, M. Fikar, and M. Latifi, "Matlab dynamic optimisation code dynopt. Users guide," tech. rep., Technical Report, KIRP FCHPT STU Bratislava, Slovak Republic, 2005.

[85] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, "DACE - A Matlab Kriging toolbox, version 2.0," tech. rep., 2002.

[86] A. W. Dowling and L. T. Biegler, "A framework for efficient large scale equation-oriented flowsheet optimization," *Computers & Chemical Engineering*, vol. 72, pp. 3–20, 2015.

[87] A. Murase, H. L. Roberts, and A. O. Converse, "Optimal thermal design of an autothermal ammonia synthesis reactor," *Industrial & Engineering Chemistry Process Design and Development*, vol. 9, no. 4, pp. 503–513, 1970.

[88] U. M. Ascher and L. R. Petzold, *Computer methods for ordinary differential equations and differential-algebraic equations*. Philadelphia: SIAM, 1998.

[89] S. Vasantharajan and L. T. Biegler, "Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria," *Computers & chemical engineering*, vol. 14, no. 10, pp. 1083–1100, 1990.

[90] J. Ma, J. P. Eason, A. W. Dowling, L. T. Biegler, and D. C. Miller, "Development of a first-principles hybrid boiler model for oxy- combustion power generation system," *International Journal of Greenhouse Gas Control*, vol. 46, pp. 136–157, 2016.

[91] M. A. Duran and I. E. Grossmann, "Simultaneous optimization and heat integration of chemical processes," *AIChE Journal*, vol. 32, no. 1, pp. 123–138, 1986.

[92] J. Black, "Quality guidelines for energy systems studies: Process modeling design parameters," Tech. Rep. DOE/NETL-341/051314, National Energy Technology Laboratory, 2014.

[93] A. Darde, R. Prabhakar, J.-P. Tranier, and N. Perrin, "Air separation and flue gas compression and purification units for oxy-coal combustion systems," *Energy Procedia*, vol. 1, pp. 527–534, Feb. 2009.

[94] A. W. Dowling, J. P. Eason, J. Ma, D. C. Miller, and L. T. Biegler, "Equation-based design, integration, and optimization of oxycombustion power systems," in *Alternative Energy Sources and Technologies*, pp. 119–158, Springer, 2016.

[95] A. S. Drud, "CONOPT a large-scale GRG code," *ORSA Journal on Computing*, vol. 6, no. 2, pp. 207–216, 1994.

[96] A. Lee and D. C. Miller, "A one-dimensional (1-d) three-region model for a bubbling

fluidized-bed adsorber," *Industrial & Engineering Chemistry Research*, vol. 52, no. 1, pp. 469–484, 2012.

[97] S. Modekurti, D. Bhattacharyya, and S. E. Zitney, "Dynamic modeling and control studies of a two-stage bubbling fluidized bed adsorber-reactor for solid–sorbent co_2 capture," *Industrial & Engineering Chemistry Research*, vol. 52, no. 30, pp. 10250–10260, 2013.

[98] M. Yu, D. C. Miller, and L. T. Biegler, "Dynamic reduced order models for simulating bubbling fluidized bed adsorbers," *Industrial & Engineering Chemistry Research*, vol. 54, no. 27, pp. 6959–6974, 2015.

[99] R. S. Kamath, L. T. Biegler, and I. E. Grossmann, "An equation-oriented approach for handling thermodynamics based on cubic equation of state in process optimization," *Computers & Chemical Engineering*, vol. 34, no. 12, pp. 2085–2096, 2010.

[100] J. V. Sengers, R. Kayser, C. Peters, and H. White, *Equations of state for fluids and fluid mixtures*, vol. 5. IUPAC, 2000.

[101] R. Privat, R. Gani, and J.-N. Jaubert, "Are safe results obtained when the pc-saft equation of state is applied to ordinary pure chemicals?," *Fluid Phase Equilibria*, vol. 295, no. 1, pp. 76–92, 2010.

[102] J. O. Valderrama, "The state of the cubic equations of state," *Industrial & engineering chemistry research*, vol. 42, no. 8, pp. 1603–1618, 2003.

[103] J. M. Prausnitz, R. N. Lichtenthaler, and E. G. de Azevedo, *Molecular thermodynamics of fluid-phase equilibria*. Pearson Education, 1998.

[104] J. Kang, L. Zhu, S. Xu, Z. Shao, and X. Chen, "An equation-oriented approach for handling perturbed-chain saft equation of state in simulation and optimization of

polymerization processes," *Industrial & Engineering Chemistry Research*, 2018. In print.

[105] C. Zhang, Z. Shao, X. Chen, Z. Yao, X. Gu, and L. T. Biegler, "Kinetic parameter estimation of hdpe slurry process from molecular weight distribution: Estimability analysis and multistep methodology," *AIChE Journal*, vol. 60, no. 10, pp. 3442–3459, 2014.

[106] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: a review of algorithms and applications," *Annals of Operations Research*, vol. 240, no. 1, pp. 351–380, 2016.

[107] R. Chen, M. Menickelly, and K. Scheinberg, "Stochastic optimization using a trust-region method and random models," *Mathematical Programming*, pp. 1–41, 2015.