Algorithms and Theory of Derivative-Free Optimization

A dissertation submitted to

University of Chinese Academy of Sciences

in partial fulfillment of the requirement

for the degree of

Doctor of Philosophy

in Computational Mathematics

By

Pengcheng Xie

Supervisor: Professor Ya-xiang Yuan

Academy of Mathematics and Systems Science, Chinese Academy of Sciences

June, 2024

Abstract

Most optimization algorithms depend on the derivative information of the problem. However, in the real world, the derivative information of objective functions in many practical problems in engineering computations, design optimization, data science, artificial intelligence, and other fields is either unavailable or prohibitively expensive. In these application scenarios, it is difficult for us to obtain and utilize precise derivative information of the problem. This motivates us to study derivative-free optimization (DFO) methods. Derivative-free optimization is one of the most important and challenging areas in scientific computing and engineering, with significant research demands and potential.

The trust-region methods based on the under-determined interpolation quadratic models is an efficient class of derivative-free optimization methods. Updating the quadratic model using different techniques will derive different models. This thesis proposes a new method to update the quadratic model, which is achieved by minimizing the H^2 norm of the change between neighboring quadratic models. We give the motivation for applying the H^2 norm and the theoretical properties of the proposed new updating method. Such a model is determined by calculating the coefficients using the KKT conditions. Numerical results show our new model's numerical advantages in solving the considered test set. We also propose the least weighted H^2 norm updating quadratic model and discuss the best weight coefficients. This thesis gives a new perspective based on the property of trust-region iteration to analyze the famous least norm type under-determined quadratic interpolation model. We find the non-determinacy of a coefficient in the optimality condition when constructing a quadratic model considering the trust-region iteration in some cases. The consequent non-uniqueness of the quadratic model leads us to propose a new model to improve the model. In detail, we selectively treat the previous under-determined quadratic model as a quadratic model or a linear model. We give an improved under-determined quadratic interpolation model, and it considers the optimality of the model based on the trust-region iteration. We consequently give a new derivative-free method. This thesis gives the theoretical motivation, analysis, and computational details. Our quadratic model's formula is implementation-friendly. The numerical results show the advantages of using our quadratic model in the derivative-free optimization methods. To the best of our knowledge, we provide the first work considering the property of trust-region iteration and the model's optimality when constructing the under-determined quadratic model for derivative-free methods. In addition, we give the conditions of distance reduction between the minimizers of non-convex quadratic functions in the trust region and the corresponding numerical ex-

amples.

This thesis proposes derivative-free optimization with transformed objective functions (DFOTO) and gives a model-based trust-region method with the least Frobenius norm updating quadratic model. The model updating formula is based on Powell's formula, and it can be easily implemented. Our method has the same framework as the methods for solving problems without transformations, and its query scheme is also given. We propose the definitions related to optimality-preserving transformations to understand the interpolation model in our method when minimizing transformed objective functions. We prove the existence of model optimality-preserving transformations beyond translation transformations. We give the corresponding necessary and sufficient condition. We also analyze the corresponding model and its interpolation error when the objective function is affinely transformed. The convergence property of a provable algorithmic framework related to the transformed objective functions is given in this thesis. Numerical results show that our method can successfully solve most test problems with objective optimality-preserving transformations. To the best of our knowledge, this is the first work providing the model-based derivative-free algorithm and analysis for transformed problems with the function evaluation oracle.

In addition, we propose a novel method named 2D-MoSub. It is a 2-dimensional model-based subspace derivative-free method. 2D-MoSub especially aims to solve large-scale derivative-free problems. 2D-MoSub combines 2-dimensional quadratic interpolation models and trust-region techniques to iteratively update the points and explore the 2-dimensional subspace. We introduce its framework and computational details, including initialization, the interpolation set, the quadratic interpolation model, trust-region trial steps, and the updating of trust-region radius and subspace. We discuss the poisedness and quality of the interpolation set in the corresponding subspace and analyze some properties of our method, which include the model's approximation error, projection property and 2D-MoSub's convergence. Numerical results show the advantage of 2D-MoSub. Besides, this thesis proposes the derivative-free optimization algorithm SUSD-TR. The speeding-up and slowing-down (SUSD) direction is proved to converge to the gradient descent direction in some cases. Our SUSD-TR combines the SUSD direction based on the covariance matrix of interpolation points and the solution of the trust-region subproblem of the interpolation model function based on such points. We analyze the dynamics of the optimization process and the direction's properties of the algorithm SUSD-TR. We discuss the trial step and structure step. Numerical results show the advantage of SUSD-TR.

Key Words: derivative-free optimization, trust-region method, quadratic interpolation, large-scale problem, subspace method

Contents

List of Figures

List of Tables

List of Symbols

Characters

Symbol	Description
$\mathfrak{R}$	set of real numbers
$\mathfrak{R}^+$	set of positive real numbers
$\mathfrak{R}^n$	set of n-dimensional real vectors
$\mathfrak{R}^{m\times n}$	set of $m \times n$ real matrices
$\mathbb{Z}$	set of integers
$\mathbb{N}$	set of natural numbers
$\mathbb{N}^+$	set of positive integers
$\mathbf{X}^{-1}$	inverse of matrix $\mathbf{X}$
$\mathbf{X}^\top$	transpose of matrix $\mathbf{X}$
$\|\mathbf{X}\|_\mathrm{F}$	Frobenius norm of matrix $\mathbf{X}$
$\mathcal{B}_r(\boldsymbol{x}_0)$	$\left\{\boldsymbol{x} \in \mathfrak{R}^n : \left\|\boldsymbol{x} - \boldsymbol{x}_0\right\|_2 \leq r\right\}$
$\mathcal{V}_n$	volume of the n-dimensional ℓ_2 unit ball $\mathcal{B}_1(\boldsymbol{x}_0)$
$\boldsymbol{I}$	identity matrix
$\mathbf{0}_{nm}$	$n \times m$ zero matrix
$\boldsymbol{e}_t$	t-th column of the identity matrix $\boldsymbol{I}$
$\langle\cdot,\cdot\rangle$	inner product

Operators

Symbol	Description
min	minimize
max	maximize
∂	partial derivative
∇	gradient

Abbreviations

DFO	derivative-free optimization
SUSD	speeding up and slowing down
KKT	Karush–Kuhn–Tucker
s. t.	subject to

Chapter 1 Introduction

In the fields of science and engineering, optimization problems have always been a highly concerned and crucial research area. Whether in industrial production, logistics and transportation, financial investment, or artificial intelligence, solving optimization problems can directly affect the efficiency and performance of systems [1–8]. By solving optimization problems, we can achieve optimal utilization of resources, reduce costs, improve product quality, and even optimize decision-making processes.

Optimization problems in practical applications often exhibit complex characteristics [9–19], including nonlinearity, nonconvexity, and multivariable dependence. These features make the solution of optimization problems more difficult and complicated. For example, in engineering design, we often need to consider various physical constraints as well as the nonlinear behavior of the system; in the financial field, investment optimization problems often involve nonlinear relationships between the returns and risks of multiple assets; in the medical field, drug formulation optimization problems may involve interactions among multiple drug components and their effects on the patient' s physiological state. Such complexities motivate us to search for more flexible and efficient numerical algorithms to solve these nonlinear optimization problems.

This dissertation focuses on the solution of unconstrained problems, considering that such problems not only have wide applications themselves but also that the corresponding methods and ideas can be extended to constrained problems. At the same time, some constrained problems can be transformed into unconstrained problems. Therefore, the unconstrained optimization methods discussed in this dissertation are fundamental to optimization methods. Most unconstrained optimization methods require the derivatives of the objective function. However, in some practical cases, the objective function is costly to evaluate, and its derivatives are unavailable. A typical example is when the objective function is not expressed analytically but obtained through a "black box," such as a chemical process or computer simulation. The optimization of such problems does not use the derivatives of the objective function and is therefore called derivative-free optimization. That is, derivative-free optimization methods are a class of numerical methods that do not require the first- or higher-order derivatives of the objective function. For more introductions to derivative-free optimization, one can refer to the monograph by Conn, Scheinberg, and Vicente [20], as well as the monograph by Audet and Hare [21].

As the introduction, this chapter mainly presents the research background, objectives, and organizational structure of this dissertation. First, in Section 1.1, we discuss the importance and application fields of derivative-free optimization. Section 1.1.1

examines specific applications of derivative-free optimization. Section 1.1.2 provides a classification and overview of derivative-free optimization methods. Subsequently, Section 1.2 gives a detailed introduction to model-based derivative-free optimization methods, which are the focus of this dissertation. In Section 1.2.1 and Section 1.2.2, we present the relevant algorithms and the fundamental concepts of interpolation models. Section 1.3 mainly introduces evaluation methods for derivative-free optimization algorithms, where we provide criteria and schemes for assessing their performance. Finally, Section 1.4 summarizes the main content and organizational structure of the dissertation.

1.1 Derivative-Free Optimization

This dissertation considers and studies unconstrained derivative-free optimization problems

$$\min_{\boldsymbol{x}\in\Re^n} \; f(\boldsymbol{x}), \tag{1-1}$$

where f is a real-valued function with no available first- or higher-order derivative information. We know that derivative-free optimization methods are a class of numerical methods that do not use the true or exact derivative information of the objective function in the optimization process. Since only function values are required, derivative-free optimization methods have wide applications. At the same time, due to the lack of derivative information, derivative-free methods have some disadvantages. For example, it is difficult to obtain a good approximation of the original objective function in the optimization problem, and we can hardly capture the shape of the black-box function accurately. In the discussion of this dissertation, we assume that it is difficult to obtain a perfectly accurate approximation of the objective function or its derivatives. Note that, for this reason, the theoretical properties of derivative-free optimization methods are usually hard to analyze.

1.1.1 Applications of Derivative-Free Optimization

Derivative-free optimization problems are very common in practical applications, and they are widely used in engineering. In the early stage of optimization algorithm development, although derivative information of nonlinear optimization problems was known in some contexts, the corresponding theoretical framework was not yet mature, and effective derivative-based methods were lacking. In such cases, many derivative-free optimization methods were favored by users at that time because of their simplicity and ease of use. With the rapid development of scientific computing technology and the increasing scale and complexity of problems, the original derivative-free methods began to show limitations in solving problems, which promoted the continuous development and refinement of more complex derivative-based methods. These methods usually re-

quire users to provide derivative information of the objective function. However, this is not always feasible, because the required function values in many practical engineering scenarios may come from measurements in physical, biological, or computer experiments. Such processes for obtaining function values can only be regarded as a black-box system.

Some application examples of derivative-free optimization include parameter tuning of numerical algorithms [22], optimization of neural networks [23], automatic error analysis [24], dynamic pricing [25], and optimal design in engineering design [26].

It is worth noting that derivative-free optimization methods have been widely applied in industry and engineering [27–29], especially playing an important role in solving problems involving complex models or experiments. These problems come from various fields, such as wing design [30], aerodynamic shape design [31, 32], fluid dynamics design [33], and circuit design [34, 35]. Any optimization problem that requires optimizing complex models, simulations, or experiments may involve derivative-free optimization. Notably, in industry, multidisciplinary design optimization [36] is a well-known specific application of derivative-free optimization. Since multidisciplinary design optimization problems usually involve simulations or experiments in industrial production, derivative-free optimization methods are often required to solve the corresponding black-box problems [37].

Many derivative-free optimization and black-box optimization problems related to data science and machine learning have also emerged [38, 39]. For example, black-box attacks on neural networks [40] can be regarded as solving a derivative-free optimization problem. In most cases, hyperparameter tuning is also a black-box problem [22]. For instance, using derivative-free optimization methods to improve parameter selection in climate modeling [41]. Below, we introduce some specific application cases of derivative-free optimization in detail.

Example 1.1 (Parameter Optimization)**.** Optimizing parameters in numerical algorithms is a very meaningful research direction [22]. We know that most numerical algorithms depend on a set of pre-specified parameters. At the same time, these parameters may be recommended values provided based on the experience of algorithm developers, or they may need to be set by the users through trial and error. The selection of these parameters has a significant impact on the performance of the algorithm. An effective parameter selection method is to obtain good parameters by solving a black-box optimization problem. We consider treating the parameters as variables, with the performance of the algorithm on the test set (for example, evaluated by CPU running time or number of iterations) as the objective function. Usually, these parameters are subject to lower and upper bound constraints. Therefore, the optimization problem can be

simply formulated as

$$\min_{\boldsymbol{p}\in\Re^n} f(\boldsymbol{p}) = \text{Performance}(\boldsymbol{p})$$
$$\text{s. t. } l_i \le \boldsymbol{p}_i \le u_i,\ \forall\, i = 1, \cdots, n,$$

where $\boldsymbol{p}$ denotes the parameters to be tuned, and $\boldsymbol{p}_i$ denotes its elements. Note that the objective function of such problems usually cannot be expressed analytically or have derivatives computed [42].

Example 1.2 (Climate Modeling and Prediction)**.** Global climate numerical models are generally extremely complex computer programs, constructed from complicated code, essentially a black box, whose purpose is to model and predict the Earth's climate change by simulating the complex interactions among the ocean, atmosphere, and land. These models combine numerous input variables, such as the concentration of greenhouse gases in the atmosphere, solar radiation, ocean circulation, and surface cover types, to simulate, model, and predict the behavior of the Earth's climate system. Because global climate models must process massive amounts of data with many variables, they need to run on high-performance computers, which further increases research cost and complexity. At the same time, these models cover long and complex time spans. In summary, parameter selection in such models can be regarded as an expensive black-box optimization problem [41].

Example 1.3 (Black-Box Attacks)**.** In the research and application of artificial intelligence, a black-box attack is a behavior aimed at undermining neural network recognition systems. Its core strategy is to add noise to the input data of a neural network to mislead it into producing incorrect outputs. For example, an attacker may introduce subtle perturbations into image data. Although these perturbations are almost imperceptible to the human eye, they can cause the neural network to make incorrect recognition decisions. For instance, by adding carefully designed noise to a picture of a panda, an attacker can trick the neural network into misclassifying it as a gibbon. The particularly challenging aspect of such attacks is that attackers usually do not have access to the internal structure and mechanisms of the corresponding neural network, that is, they are in a so-called black-box environment. In other words, the attacker must find noise that can effectively interfere with the network's output without any knowledge of its internal parameters or architecture. This essentially transforms the problem into a black-box optimization problem [40].

Example 1.4 (Molecular Geometry)**.** In scientific research in chemistry and physics, one striking application area is the optimization of molecular geometry. When we consider a molecule or atomic cluster containing multiple atoms, its geometric configuration can be described by certain degrees of freedom or variables. Specifically, our goal is to find a good geometric configuration such that the potential energy of the entire molecule

or cluster is minimized, i.e., achieving the lowest possible energy. Although gradients for such optimization problems can sometimes be obtained, computing them may be costly and noisy. In such cases, derivative-free optimization methods, especially direct search strategies, have proven to be effective tools [43, 44].

In addition, there are also some optimal strategy problems that fall under derivative-free optimization. For example, Wild and Shoemaker described the problem of determining the best pumping strategy to reduce harmful contaminants in groundwater. Simply put, the problem involves a set of wells operating at certain pumping rates, which can inject clean water or remove and treat contaminated water. Suppose we want to explore: for 15 wells operating for more than 30 years, what is the optimal, lowest-cost pumping strategy to ensure that the concentration of harmful substances in the aquifer meets environmental standards. Here, a single evaluation (under the conditions at that time) requires more than 45 minutes of groundwater flow simulation. Moreover, the simulation code is too complex to obtain automatic differentiation. This is a typical black-box derivative-free optimization problem [45].

1.1.2 Classification and Overview of Derivative-Free Optimization Methods

Derivative-free methods, with a long research history, have different types: for example, direct search methods, line search methods, model-based methods, heuristic algorithms, and so on. In this section, we provide a brief introduction. The articles by Ding [42] and Zhang [46] also contain detailed introductions, and we refer to their summaries and discussions here.

Direct Search Methods

Direct search methods are a large class of derivative-free methods [47–49], which mainly construct certain geometric structures to search within the feasible region of variables. In general, direct search methods neither use derivative values of the original objective function nor use approximate derivative information obtained by finite differences. These methods mainly rely on low-dimensional geometric intuition and lack rich, rigorous mathematical theory. Historically, the term “direct search” was first proposed by Hooke and Jeeves in 1961. In their study, one core feature of direct search was that it only required comparing the relative magnitudes of function values at trial points, without depending on their specific values. In other words, any decrease in the objective function value would be accepted. Clearly, this makes the method relatively simple and intuitive.

Specifically, direct search methods include pattern search methods, simplex methods, directional direct search methods, and mesh adaptive direct search methods. Some examples are the Hooke–Jeeves method [50], the Nelder–Mead method [51], improved simplex methods [52], and generating set search methods [49]. The pattern search meth-

ods mentioned here are one type of direct search method [46], including the Fermi–Metropolis method [53], the Evolutionary Operation method [54], the Hooke–Jeeves method [50], multidirectional search methods [55–57], generalized pattern search methods [58–62], asynchronous parallel pattern search methods [63, 64], and mesh adaptive direct search (MADS) methods [65, 66]. In fact, direct search in the modern sense can at least be traced back to the research report of Fermi and Metropolis [53].

To give an intuitive description, minimizing a bivariate function can be likened to starting from a point on a mountain and searching for the lowest altitude. One can imagine that a basic approach is to find a valley and move forward along it. The main idea of pattern search methods is to use exploratory steps to obtain information for finding valleys, and then use pattern steps to advance along the valley [67].

In addition, simplex methods [20] are also a class of direct search methods. The Nelder–Mead simplex method [51] is a representative of this class [46]. This method has been widely used in practice. Briefly, for an n-dimensional problem, the Nelder–Mead method starts from a simplex formed by $n + 1$ initial points and iterates by reflecting, expanding, and contracting the simplex according to the function values at its vertices. The basic principle of the algorithm is that the simplex iteratively approximates the shape of the objective function locally [51], eventually converging successfully. Unfortunately, the Nelder–Mead method does not have a solid convergence theory (even for strictly convex functions [68]). In addition, scholars have proposed modifications and improvements to the Nelder–Mead method [69–71].

Here, we give another example of a specific direct search algorithm: directional direct search methods. Briefly, at each iteration, a finite set of points is generated near the current point $\boldsymbol{x}_k$. These candidate points are generated by moving from $\boldsymbol{x}_k$ in directions $\alpha_k \boldsymbol{d}$, where α_k is a positive step length and the direction $\boldsymbol{d}$ is chosen from a finite set of directions corresponding to the current step. Then the method evaluates the objective function at all or some of these candidate points, and sets $\boldsymbol{x}_{k+1}$ as a point that may decrease the function value and possibly increase the step length. Note that if the algorithm finds that none of the candidate points provide sufficient decrease, then $\boldsymbol{x}_{k+1}$ is set to $\boldsymbol{x}_k$ and the step length is reduced. Kolda et al. [49] proposed the term "generating set search methods" to define this class.

It is worth noting that during the study of direct search algorithms, scholars gradually established and developed systematic theoretical foundations for the mathematical concept of positive bases [20, 21].

Line Search Methods

Another class of derivative-free methods are line search methods that do not use derivative information of the original objective function. Scholars observed that direct search methods rely on comparing function values at grid points or simplex vertices

without considering properties such as continuity and smoothness, which leads to slow convergence. When one-dimensional search techniques were introduced, the efficiency of these methods was significantly improved [42]. Specifically, the classical and basic line search framework for solving the unconstrained optimization problem (1-1) is shown in Algorithm 1 [46]. Depending on the choice of search directions, there are generally three types of line search methods: alternating direction methods (e.g., the Rosenbrock method [72] and its improved versions [73]), conjugate direction methods [74, 75], and approximate gradient-based methods (e.g., finite-difference quasi-Newton methods [76–78]). In addition, there are some methods based on stochastic gradient approximations [79–83]. In general, once a search direction is given, an appropriate step length can be chosen using methods such as interval division [2, 84].

Algorithm 1 Framework of Line Search Methods

Step 1. (Initialization) Obtain the initial point $\boldsymbol{x}_1$, and let $k = 1$.

Step 2. (Choose search direction) Select the search direction $\boldsymbol{d}_k$.

Step 3. (Choose step length) Solve

$$\min_{\alpha \geq 0} \ f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k),$$

to obtain the step length α_k.

Step 4. (Update) Let $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k$, set $k = k + 1$, and go to Step 2.

As a class of derivative-free line search methods, the basic principle of coordinate rotation methods is to use each coordinate direction in turn as the search direction. One drawback is that zigzagging may occur. A way to overcome this difficulty is to introduce the pattern search of Hooke and Jeeves [50]. Note that a simple extension of coordinate rotation is to extend coordinate directions to any orthogonal basis. The Rosenbrock method [72], also called the rotating axes method, uses a similar idea. Yuan Yaxiang's monograph [2] presents results on the convergence of alternating direction methods.

In addition, conjugate direction methods iteratively generate conjugate directions as search directions during the solution process. The earliest conjugate direction method was proposed by Smith [74]. Powell [75] and Zagwill [85] also studied conjugate direction methods.

The idea of gradient-approximation-based line search methods is to obtain numerical approximate gradients in some way, and to build algorithmic frameworks analogous to classical derivative-based methods. One of the simplest and most direct approaches to approximate gradients is finite differences. The difference quasi-Newton method [76] is a quasi-Newton method based on finite-difference approximations. Gill and Murray [77] studied Broyden family methods based on finite differences. Implicit filtering methods [86–88] can be regarded as quasi-Newton methods based on simplex gradients.

In addition, there are derivative-free quasi-Newton methods [89, 90], which formulate quasi-Newton conditions based only on function values, where the gradient and Hessian satisfy certain variational minimal properties [46]. In recent years, research and methods on stochastic gradient approximations [79–83] have also matured, including single-point stochastic gradient approximation, two-point stochastic approximation, and multi-point stochastic approximation.

Model-Based Methods

Model-based methods are an important class of derivative-free methods. Specifically, quadratic interpolation model methods, underdetermined quadratic interpolation model methods, and regression model methods all use polynomial models [91–99]. Another type of model-based derivative-free method is based on radial basis function interpolation models [100, 101]. In fact, there are also some hybrid model-based methods, such as those combining trust-region methods and line search methods [102]. Scholars have also discussed probabilistic model-based trust-region methods [103, 104]. Model-based derivative-free optimization methods are the focus of this dissertation, and we will provide a detailed discussion in Section 1.2.

Heuristic Algorithms

In fact, most modern heuristic algorithms also do not use derivative information. Simulated annealing algorithms [105] and genetic algorithms [21] belong to this class [42].

Genetic algorithms are derived from the principles of natural selection and genetics. This algorithm was first proposed by Holland in 1975 [106–108]. It is a search technique that simulates the principle of "survival of the fittest" and the random gene exchange mechanism in biological evolution. The method imitates and refers to the laws of biological evolution and heredity. After encoding, the algorithm starts from an initial population (initial feasible solutions) and iteratively applies reproduction, crossover, and mutation. By repeatedly applying these operations across generations, the algorithm eliminates inferior solutions and cultivates new solutions, eventually finding the optimal solution. Its core idea is to use the "survival of the fittest" rule to eliminate poor solutions and breed new ones for the optimization problem. To remain consistent with biological terminology, search points are usually called individuals, collections of points form populations, information within a point is encoded as a chromosome, and new search points are obtained by using processes analogous to biological operations [21].

Simulated annealing algorithms are search algorithms developed from metal heat treatment. In 1983, Kirkpatrick et al. [109, 110] applied this idea to solve optimization problems. Its basic idea is to treat an optimization problem as analogous to a metallic object, where the objective function, the solutions, and the optimal solution correspond

to the energy of the object, its state, and the lowest-energy state, respectively. Then the annealing process of the metallic object is simulated: starting from a sufficiently high temperature and gradually lowering the temperature, so that the molecules of the object reach the ideal state of minimum energy, thereby finding the solution of the optimization problem.

Other Methods and Some Numerical Software

Derivative-free methods also include hybrid methods, such as implicit filtering methods [87, 88] and adaptive regularization methods [111]. Below we briefly introduce one hybrid method.

Implicit filtering is a class of hybrid methods that can be understood as a mixture of grid search algorithms and local quasi-Newton optimization methods, where the corresponding finite-difference parameters are adjusted adaptively.

In addition, evolutionary algorithms [112] form a large class of methods, many of which do not require derivative information of the objective function. The covariance matrix adaptation evolution strategy (CMA-ES) [113–115] is one such evolutionary strategy algorithm. Briefly, it samples from a Gaussian distribution in the solution space of the optimization problem, and updates the Gaussian distribution according to a certain sample selection mechanism. By iteratively repeating the sampling and updating process, this method eventually finds a satisfactory solution.

Inspired by the behavior of fish schools searching for darker regions in their environment, a distributed source-seeking strategy has recently emerged. This strategy generates a speeding-up slowing-down (SUSD) behavior [116, 117], which is very similar to the behavior observed in fish schools. This strategy can be regarded as a particle swarm (bio-inspired) optimization algorithm, allowing each search point to measure the field value (corresponding to the objective function value in the optimization setting) in real time, and collectively move toward an approximate negative gradient direction. Here, the movement speed of each search point is designed to be proportional to its field measurement. This method will also be introduced and improved later in the dissertation.

There also exist derivative-free methods for special problems, such as methods for least-squares problems [118, 119], derivative-free methods for composite optimization [120, 121], and examples with special constraints, such as ellipsoidal-constrained derivative-free optimization [122] and distributed derivative-free optimization [123]. There are also other types of derivative-free optimization methods, such as Bayesian optimization methods [124], methods using stochastic techniques [103, 125, 126], and global optimization [127]. In addition to the monographs of Conn, Scheinberg, and Vicente [20] and Audet and Hare [21], some survey papers, such as those by Larson, Menickelly, and Wild [128], Zhang [129], and Rios and Sahinidis [130], also provide detailed introductions to various types of derivative-free optimization methods.

In addition, researchers have developed software and solvers for derivative-free optimization methods. Examples include CMA-ES [131], DFO [132], IMFIL [133], and SOLNP+ [134]. Powell developed a series of algorithms including TOLMIN [135], COBYLA [136], UOBYQA [137], NEWUOA [94], BOBYQA [138], and LINCOA [139]. In addition, there are DFO-LS [140] and DFBGN [141]. Recently, Ragonneau and Zhang developed PDFO, which provides a cross-platform interface for Powell's derivative-free solvers [142], and they also developed the COBYQA [143] algorithm. We developed MATLAB implementations [144, 145] and Python implementations of the NEWUOA and BOBYQA algorithms. In addition, Zhang's PRIMA [146] provides a modernized and improved reference implementation of Powell's methods.

1.2 Model-Based Derivative-Free Optimization Methods

1.2.1 Algorithms

Here we specifically introduce model-based derivative-free optimization methods. Model-based methods are a classical and efficient class of derivative-free methods. The approaches to obtaining polynomial models include linear interpolation [147], quadratic interpolation [91, 148], underdetermined quadratic interpolation [92], and regression [20]. Radial basis function interpolation [100] is another choice for constructing the employed model. At the same time, random models can also be used in trust-region methods [103, 104]. In addition, there are model-based methods designed for noisy problems [149], methods for minimizing transformed objective functions [150], and approaches using cubic models [151].

The main idea of model-based derivative-free optimization methods is to construct a model function to locally approximate the original black-box objective function at each iteration. Then the algorithm uses the information of the model function (including gradient) to gradually obtain iterates. Most model-based methods adopt a trust-region framework [152], generating new iterates by minimizing a quadratic model within a region around the current or best iterate (usually centered there), as shown in (1-2). For derivative-free cases, the corresponding model is usually constructed via interpolation of function values at interpolation points (polynomial interpolation). Such methods are called model-based (interpolation) derivative-free trust-region methods.

Specifically, quadratic interpolation model methods [91], underdetermined quadratic interpolation model methods [92, 93, 97, 102, 150, 153, 154] and regression model methods [95, 96] all use approximation models. The algorithms UOBYQA [137] and CONDOR [155] employ quadratic models. Algorithms NEWUOA [94], DFO [156], and MNH [157] are examples of methods using underdetermined quadratic interpolation models. Another type of model-based derivative-free method is the radial basis function interpolation model method [100], with BOOSTERS [158] and ORBIT

[101] being examples of such algorithms. There are also wedge trust-region methods [159], related methods for least squares problems (including algorithms DFBGN [141] and DFO-LS [140]), as well as algorithms using sparse models [153]. This dissertation will focus on the interpolation step of model-based derivative-free trust-region methods. The framework of a model-based derivative-free trust-region algorithm is shown in Algorithm 2. The basic convergence of derivative-free trust-region algorithms [96, 160] as well as improvements in flexibility and robustness [140] have been studied. Discussions on solving the quadratic model trust-region subproblem (1-2) can be found in Chapter 7 of the book by Conn, Gould, and Toint [152]. One of the common methods for solving the trust-region subproblem of a model function is the truncated conjugate gradient method [161–163].

Algorithm 2 Framework of a model-based derivative-free trust-region algorithm

Input: black-box objective function f and initial point $\boldsymbol{x}_{\text{int}}$.

Output: minimizer $\boldsymbol{x}^*$ and minimal function value.

Obtain/set the initial trust-region radius Δ_0 and other initial parameters.

Step 1. (Manage interpolation set)

Select an interpolation set $\mathcal{X}_k \subset \mathfrak{R}^n$. Most points in $\mathcal{X}_k$ have already been evaluated in previous iterations. Here, the algorithm evaluates all $\boldsymbol{y} \in \mathcal{X}_k$ whose function values are not yet known, obtaining $f(\boldsymbol{y})$.

Step 2. (Construct interpolation model)

Use a linear or quadratic interpolation model function (or radial basis function interpolation model) Q_k to approximate f.

Step 3. (Trust-region iteration)

Obtain $\boldsymbol{x}_k^+$ and function value $f(\boldsymbol{x}_k^+)$ by solving

$$\begin{aligned} &\min_{\boldsymbol{x}} Q_k(\boldsymbol{x}) \\ &\text{s. t. } \left\|\boldsymbol{x} - \boldsymbol{x}_k\right\|_2 \leq \Delta_k \end{aligned} \tag{1-2}$$

and update $\boldsymbol{x}_{k+1}$ and Δ_{k+1} accordingly. Specifically, by evaluating the function value at the new point and comparing the actual reduction of the objective with the predicted reduction of the model, the algorithm determines whether the iteration is successful, and updates the iterate and the trust region.

If the termination condition is not satisfied, set $k = k + 1$, go to **Step 1**.

In fact, constructing a good local approximation model for trust-region methods, with or without derivative information of the objective function, is very important. In derivative-free optimization, the most common approach to obtain a model is to deter-

mine model Q based on interpolation of sampled function values, with the form

$$Q(\boldsymbol{x}) = \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{H}\left(\boldsymbol{x} - \boldsymbol{x}_0\right) + \boldsymbol{g}^\top\left(\boldsymbol{x} - \boldsymbol{x}_0\right) + c,$$

where $\boldsymbol{x}_0$ is a given vector, symmetric matrix $\boldsymbol{H} \in \mathfrak{R}^{n\times n}$, $\boldsymbol{g} \in \mathfrak{R}^n$ and $c \in \mathfrak{R}$ together contain $\frac{1}{2}(n+1)(n+2)$ unknown coefficients to be determined. A quadratic function selected from the set of quadratics that satisfies the function value constraints

$$Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k \tag{1-3}$$

is denoted as Q, where $\mathcal{X}_k$ denotes the interpolation set at the k-th iteration (also called interpolation set or interpolation points set). We assume the m interpolation points in $\mathcal{X}_k$ are $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m$.

In this dissertation, a determined quadratic model [20] refers to the case where all $\frac{1}{2}(n+1)(n+2)$ independent parameters can be uniquely determined by the interpolation conditions (1-3). An underdetermined quadratic model (Chapter 5 of the book by Conn, Scheinberg, and Vicente [20]) refers to the case where after satisfying the interpolation conditions (1-3), the quadratic function still has remaining degrees of freedom, since $m < \frac{1}{2}(n+1)(n+2)$. In other words, an underdetermined quadratic model cannot be uniquely determined solely by the function value constraints (1-3). In derivative-free optimization, underdetermined quadratic models are one of the most important types of quadratic models used in trust-region algorithms.

It should be pointed out that, for the purpose of saving function evaluation costs when solving the algorithm, in interpolation-based methods, after the k-th iteration is completed, the points in the interpolation set $\mathcal{X}_k$ are usually not completely discarded. Most of them are inherited by $\mathcal{X}_{k+1}$. At the same time, in the vast majority of cases, the newly obtained iterate is also added into $\mathcal{X}_{k+1}$, unless doing so would seriously affect the well-poisedness of $\mathcal{X}_{k+1}$[1]. Considering that interpolation accuracy is usually good locally, the underdetermined quadratic models we wish to explore are often more suitable to represent the local properties of the objective function rather than its global properties, and are thus often used in trust-region frameworks.

There are three main reasons to consider using underdetermined quadratic interpolation functions within a trust-region framework. First, quadratic model functions can capture and describe the curvature information of the objective function locally. Second, using fewer interpolation points can reduce the number of function evaluations. Last but not least, the samples/interpolation points need to be reasonably close to the current iterate. However, quadratic models require $\mathcal{O}(n^2)$ interpolation points. As a result, in many cases, the number of useful interpolation points is fewer or much fewer than the number of elements in the polynomial basis.

[1] An introduction will be given later.

Note that, as mentioned earlier, the coefficients of the quadratic model function Q_k are the symmetric Hessian matrix $\nabla^2 Q_k$, the gradient vector ∇Q_k, and a constant term. Their total degrees of freedom are $\frac{1}{2}(n+1)(n+2)$, i.e., $\mathcal{O}(n^2)$. When the dimension n of the problem is large, if we directly determine the coefficients of the model function Q_k by solving the interpolation equations (1-3), the number of function evaluations will be very large. To reduce the number of function evaluations, we can use fewer interpolation points to construct the quadratic model function. To uniquely determine the coefficients of Q_k, Powell [94] suggested taking the solution of the subproblem

$$\begin{aligned} &\min_{Q \in \mathcal{Q}} \ \left\| \nabla^2 Q - \nabla^2 Q_{k-1} \right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}) = f(\boldsymbol{y}), \ \forall \ \boldsymbol{y} \in \mathcal{X}_k \end{aligned} \tag{1-4}$$

as the desired quadratic model Q_k, where the notation $\|\cdot\|_F$ denotes the Frobenius norm, i.e., for a given matrix $\boldsymbol{C} \in \mathfrak{R}^{n \times n}$, its Frobenius norm is defined as $\|\boldsymbol{C}\|_F = (\sum_{i,j} c_{ij}^2)^{\frac{1}{2}}$, where $c_{ij}, 1 \leq i, j \leq n$, are the elements of matrix $\boldsymbol{C}$. Note that here $\mathcal{X}_k$ denotes the interpolation set at the k-th iteration, and $\mathcal{Q}$ denotes the set of quadratic functions[2]. In other words, the quadratic model function Q_k obtained in this way satisfies: among all quadratic functions satisfying the interpolation conditions $Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i), \ i = 1, \cdots, m$, the corresponding $\nabla^2 Q_k - \nabla^2 Q_{k-1}$ has the smallest Frobenius norm, where $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m$ denote the current interpolation points, and $m < \frac{1}{2}(n+1)(n+2)$.

According to the convexity of the Frobenius norm, it can be proved that in the k-th iteration, the quadratic model Q_k can be uniquely determined. This type of model has the advantage of possessing a projection property when the {objective function is quadratic} [93], i.e.,

$$\left\| \nabla^2 Q_k - \nabla^2 f \right\|_F \leq \left\| \nabla^2 Q_{k-1} - \nabla^2 f \right\|_F$$

holds for such Q_k and any {quadratic function} f.

If in (1-4), $\nabla^2 Q_{k-1}$ is replaced by the zero matrix, this corresponds to the minimum Frobenius norm quadratic model [156, 157].

1.2.2 Related Concepts of Interpolation Models

As mentioned earlier, interpolation models are very important for model-based trust-region derivative-free optimization algorithms, and the interpolation set that provides interpolation conditions is also crucial for constructing interpolation models. We now introduce the relevant concepts of interpolation models and interpolation sets. The first thing to introduce is how we should define a measure of poisedness for an interpolation point set. In fact, given an interpolation set $\mathcal{X}$, a good poisedness measure should reflect

[2] We use the term "quadratic function" to refer to polynomials of nonnegative integer degree no greater than 2.

how the set "covers" the region of interest for interpolation. For example, in the linear case, a "good coverage" usually means that the points in $\mathcal{X}$ are affinely independent.

Clearly, such a metric will depend on $\mathcal{X}$ itself and on the region considered. For example, in the case of linear interpolation, the set $\mathcal{X} = \{(0,0)^\top, (0,1)^\top, (1,0)^\top\} \subset \Re^2$ is a well-posed set inside the ball $\mathcal{B}_1(\mathbf{0})$, but not a well-posed set inside the ball $\mathcal{B}_{10^6}(\mathbf{0})$[3]. In addition, the well-poisedness of the set $\mathcal{X}$ also depends on the corresponding polynomial space for interpolation.

We will use the following definition of a well-posed set, which is given by referring to Definition 3.6 in the monograph of Conn, Scheinberg, and Vicente [20].

Definition 1.1. *Let $\Lambda > 0$, and let $\mathcal{B} \in \Re^n$. Let $\phi = \{\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \cdots, \phi_m(\boldsymbol{x})\}$ be a basis in the n-dimensional polynomial space $\mathcal{P}_n^d$ of degree no greater than d. The set $\mathcal{X} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_m\}$ is said to be Λ-poised in $\mathcal{B}$ (in the interpolation sense) if and only if*

1. For the basis of Lagrange polynomials[4] $l_i(\boldsymbol{x})$ associated with $\mathcal{X}$, we have

$$\Lambda \geq \max_{1 \leq i \leq m} \max_{\boldsymbol{x} \in \mathcal{B}} \left| l_i(\boldsymbol{x}) \right|,$$

or, equivalently,

2. For any $\boldsymbol{x} \in \mathcal{B}$, there exists $\boldsymbol{\lambda}(\boldsymbol{x}) \in \Re^m$ such that

$$\sum_{i=1}^{m} \lambda_i(\boldsymbol{x}) \phi(\boldsymbol{y}_i) = \phi(\boldsymbol{x}) \text{ and } \|\lambda(\boldsymbol{x})\|_\infty \leq \Lambda,$$

where λ_i denotes the i-th element of $\boldsymbol{\lambda}$ here.

Or, equivalently,

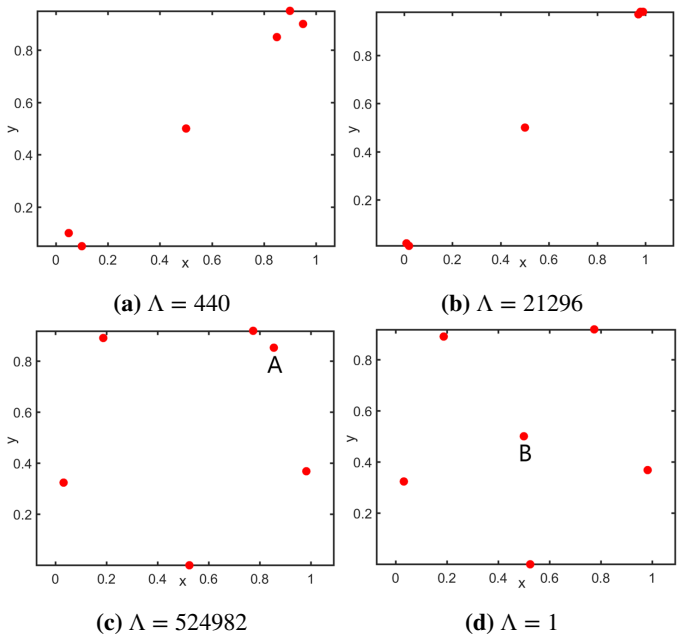
3. By replacing any point in $\mathcal{X}$ with any point $\boldsymbol{x}$ in $\mathcal{B}$, the corresponding volume of the set $\{\phi(\boldsymbol{y}_1), \phi(\boldsymbol{y}_2), \cdots, \phi(\boldsymbol{y}_m)\}$ can be enlarged by at most a factor of Λ.

To more intuitively illustrate the concrete strategy of improving an interpolation set by generating interpolation points, we present an example in Figure 1-1 [20], where six interpolation points are distributed over the region $[0,1] \times [0,1]$. We can observe that the poisedness constant Λ in subfigure 1-1d of Figure 1-1 is the smallest among all the subfigures. In addition, sometimes the poisedness constant Λ can be greatly improved after one of the six interpolation points is replaced by another point (for example, when point A in subfigure 1-1c of Figure 1-1 is replaced by point B in subfigure 1-1d).

Below we give an assumption on the objective function f for theoretical purposes. Note that this assumption is used for theoretical analysis and does not imply that derivative-free optimization methods can only solve problems that satisfy this assumption.

[3] The specific reason is omitted here.

[4] See the monograph of Conn, Scheinberg, and Vicente [20].



(a) $\Lambda = 440$ **(b)** $\Lambda = 21296$

(c) $\Lambda = 524982$ **(d)** $\Lambda = 1$

Figure 1-1 The poisedness constants Λ with different distributions on $[0,1] \times [0,1]$

Assumption 1.2. Suppose a set S and a radius $\Delta_{\max}$ are given. Assume that f is continuously differentiable in a suitable open neighborhood of $\bigcup_{\boldsymbol{x} \in S} \mathcal{B}_{\Delta_{\max}}(\boldsymbol{x})$, and that its gradient is Lipschitz continuous.

Next we present the definition of fully linear models that needs to be mentioned in this dissertation.

Definition 1.3 (Fully linear models, see Definition 6.1 in the monograph of Conn, Scheinberg, and Vicente [20])**.** *Given a function $f : \mathfrak{R}^n \to \mathfrak{R}$ satisfying Assumption 1.2. We call a class of model functions $\{Q : \mathfrak{R}^n \to \mathfrak{R},\ Q \in C^1\}$ a class of fully linear models if it satisfies the following conditions:*

1. There exist positive constants κ_{ef}, κ_{eg} and ν_Q such that for any $\boldsymbol{x} \in S$ and $\Delta \in (0, \Delta_{\max}]$, there exists a model function $Q(\boldsymbol{y})$ in the class whose gradient is continuous and whose Lipschitz constant is bounded above by ν_Q, and which satisfies:
the error between the model gradient and the gradient of f satisfies

$$\|\nabla f(\boldsymbol{y}) - \nabla Q(\boldsymbol{y})\|_2 \leq \kappa_{eg}\Delta,\ \forall\ \boldsymbol{y} \in \mathcal{B}_\Delta(\boldsymbol{x}),$$

and the error between the model and the function f satisfies

$$|f(\boldsymbol{y}) - Q(\boldsymbol{y})| \leq \kappa_{ef}\Delta^2,\ \forall\ \boldsymbol{y} \in \mathcal{B}_\Delta(\boldsymbol{x}).$$

Such a model Q is said to be fully linear on $\mathcal{B}_\Delta(\boldsymbol{x})$.
2. For this class, there exists an algorithm, which we call a "model-improvement" algorithm (see Chapter 6 of the monograph by Conn, Scheinberg, and Vicente [20]), that, in a finite number of steps uniformly bounded with respect to $\boldsymbol{x}$ and Δ, satisfies one

of the following:
- Determines that a given model Q in the class is fully linear on $\mathcal{B}_\Delta(\boldsymbol{x})$;
- Finds a model $\tilde{Q}$ in the class that is fully linear on $\mathcal{B}_\Delta(\boldsymbol{x})$.

1.3 Evaluation Methods for Derivative-Free Optimization Algorithms

This section introduces evaluation methods for derivative-free optimization algorithms. In fact, evaluating and comparing different algorithms is an important part of research on optimization algorithms, and it is related to how we judge the quality of algorithm design. To guide algorithm selection for practical applications and specific improvements in research, we need to establish and use reliable evaluation systems. Here we introduce approaches to evaluating derivative-free optimization methods beyond the traditional practice of observing curves of iterative function values. The methods introduced here will be used in subsequent chapters for algorithm comparison. It is important to note that, in derivative-free optimization, we usually focus on the number of function evaluations or the number of sampled points, rather than the number of algorithmic iterations.

The most common evaluation criteria for derivative-free optimization algorithms are the Performance Profile [21, 164, 165] and the Data Profile [21, 165]. They are the two most common ways to compare derivative-free algorithms by solving test problem sets. The two profiles help us display, in compact graphical form, information such as the convergence speed of derivative-free optimization algorithms and the proportion of problems solved successfully.

Assume $\boldsymbol{x}_N$ is the best point found by the algorithm after N function evaluations, $\boldsymbol{x}_{\text{int}}$ is the initial point, and $\boldsymbol{x}^*$ is a known best solution. Given an accuracy $\tau \in [0, 1]$, we define

$$T_{a,p} = \begin{cases} 1, \text{ if for some } N \text{ we have } f(\boldsymbol{x}_N) \leq f(\boldsymbol{x}^*) + \tau \left(f(\boldsymbol{x}_{\text{int}}) - f(\boldsymbol{x}^*) \right), \\ 0, \text{ otherwise,} \end{cases}$$

where a denotes the corresponding algorithm and p denotes the corresponding problem. Note that, in the comparisons of this dissertation, $\boldsymbol{x}^*$ is obtained through numerical experiments.

We first give the definition of the Performance Profile. In a Performance Profile, the function $\rho_a : [1, \infty) \mapsto [0, 1]$ corresponds to the proportion of problems in the test set $\mathcal{P}$ that algorithm $a \in \mathcal{A}$ solves successfully, and is defined as

$$\rho_a(\alpha) = \frac{1}{|\mathcal{P}|} \left| \left\{ p \in \mathcal{P} : r_{a,p} \leq \alpha \right\} \right|,$$

where

$$r_{a,p} = \begin{cases} \dfrac{N_{a,p}}{\min \left\{ N_{\tilde{a},p} : \tilde{a} \in \mathcal{A}, T_{\tilde{a},p} = 1 \right\}}, & \text{if } T_{a,p} = 1, \\ \infty, & \text{if } T_{a,p} = 0, \end{cases}$$

$$N_{a,p} = \min \left\{ N \in \mathbb{N}^+, f(\boldsymbol{x}_N) \leq f(\boldsymbol{x}^*) + \tau \left(f(\boldsymbol{x}_{\text{int}}) - f(\boldsymbol{x}^*) \right) \right\}.$$

Note that here the notation $|\cdot|$ denotes the number of elements in the corresponding set.

The above profile is created by plotting the function ρ_a for all algorithms in $\mathcal{A}$. It attempts to examine and present algorithmic efficiency and robustness. A characteristic of the Performance Profile is that a higher curve corresponds to better solution performance for that algorithm; here we use NF to denote the number of function evaluations used in solving.

In addition, we use the Data Profile [165] to provide some raw information (the Performance Profile focuses on comparing different algorithms, whereas the Data Profile reflects the number of function evaluations required to solve the test set). This is valuable as a reference for users who have a specific computational budget and need to choose an algorithm that may achieve a given amount of function value decrease. In a Data Profile,

$$\delta_a(\beta) = \frac{1}{|\mathcal{P}|} \left| \left\{ p \in \mathcal{P} \,:\, N_{a,p} \leq \beta \, (n+1) \, T_{a,p} \right\} \right|,$$

and the larger the value of $\delta_a(\beta)$, the more problems are solved successfully.

This dissertation will use these two evaluation methods to compare different algorithms, including newly designed ones. It should also be noted that, in some profiles in this dissertation, a logarithmic scale transformation will be applied to the horizontal axis in order to display the comparative content and regions that we are more concerned with.

1.4 Main Content of the Dissertation

This dissertation will attempt to analyze and address the following questions around derivative-free optimization.

(1) How can we design better approximation models for model-based derivative-free optimization methods?

(2) How are the models and model-based methods affected by transformed and noisy output function values?

(3) How can we solve large-scale derivative-free optimization problems more effectively?

(4) Can quadratic interpolation models improve line search methods that use approximate first-order derivatives?

Regarding question (3), we know that model-based trust-region methods are a mature class of algorithms for solving nonlinear programming problems. Most of these algorithms use quadratic models because quadratic models can effectively fit curvature information of the objective function. However, for existing model-based derivative-free optimization methods, solving large-scale problems remains a bottleneck, because

when the problem dimension is high, the computational cost of constructing local (polynomial) models and the interpolation error may be high, leading to poor practical performance. This can be regarded as the "curse of dimensionality" in derivative-free optimization. Traditional derivative-free optimization methods use quadratic interpolation models to approximate the objective function, but the limited available information of a black-box objective function leads to lower reliability of these models. Currently, some methods have been proposed and developed to handle large-scale problems; an important approach is to use subspace methods, whose main idea is to minimize the objective function in a low-dimensional subspace at each iteration to obtain the next iterate [35, 141, 166–169]. This dissertation will introduce research carried out around using subspace methods to solve large-scale derivative-free optimization problems.

Specifically, the main thread of the remaining content of this dissertation is as follows: Chapter 2 focuses on improvements to quadratic interpolation models in derivative-free trust-region algorithms. We introduce the minimum H^2-norm updated quadratic interpolation model, including the motivation for constructing minimum-norm quadratic models using the H^2 norm and the properties of the model, the formula of the minimum H^2-norm updated quadratic model, and present corresponding numerical results and summaries. At the same time, we discuss the minimum weighted H^2-norm updated quadratic interpolation model, including the minimum weighted H^2-norm updated quadratic model and the KKT matrix, the error of the KKT matrix, and the barycenter of the weight coefficient region of the minimum weighted H^2-norm updated quadratic model, and give corresponding numerical results and summaries. In addition, we introduce a derivative-free method using a new underdetermined quadratic interpolation model, including its background and motivation, model details that consider properties of the previous trust-region iteration, the convexity of the subproblem for obtaining the corresponding model and the computational formula of the model, as well as the corresponding numerical results. We also introduce conditions under which the distance between minimizers of nonconvex quadratic functions in a trust region decreases, and present corresponding numerical results and summaries.

Chapter 3 introduces derivative-free optimization with transformed objective functions and algorithms based on minimum Frobenius-norm updated quadratic models. Specifically, it includes the corresponding problems, model-based trust-region optimization algorithms and sampling schemes, minimum Frobenius-norm updated quadratic models for transformed objective functions, trust-region subproblems, and optimality-preserving transformations. In addition, we discuss model properties corresponding to positive monotone transformations and affine transformations, as well as the corresponding fully linear models and convergence analysis, and we present numerical results and summaries for corresponding test problems and real-world problems.

Chapter 4 introduces subspace methods and parallel methods. For large-scale problems, we propose a new derivative-free subspace trust-region method: the 2D-MoSub algorithm, and we provide details including the 2D-MoSub algorithm, the poisedness and quality of the interpolation set, some properties of 2D-MoSub, and numerical results. At the same time, we propose a new (parallelizable) derivative-free optimization algorithm that combines line search and trust-region techniques, including its background and motivation, the specific process of combining the SUSD direction with trust-region interpolation, theoretical analysis of the iterative direction of the SUSD-TR algorithm, exploratory steps and structure steps, as well as numerical results.

The final chapter provides a summary of the dissertation and discusses future research directions.

Chapter 2 Improvements to Quadratic Interpolation Models in Derivative-Free Trust-Region Algorithms

2.1 Least H^2 Norm Update of Quadratic Interpolation Models in Derivative-Free Trust-Region Algorithms

The methods and models proposed in this section are improvements over Powell's derivative-free optimization algorithms and models [92, 94, 170]. The main idea of Powell's method is to obtain a quadratic model function via underdetermined interpolation at each iteration and to update the previous model. Specifically, at the k-th iteration, the unique quadratic model Q_k is obtained by solving

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \ \left\|\nabla^2 Q - \nabla^2 Q_{k-1}\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \boldsymbol{y}_i \in \mathcal{X}_k. \end{aligned} \tag{2-1}$$

We denote by m the number of interpolation points in $\mathcal{X}_k$. The method aims to obtain a new iterate by minimizing the quadratic model Q_k within the trust region. Conn and Toint [171], Conn, Scheinberg, and Toint [160, 172], and Wild [157] proposed choosing the objective in (2-1) as $\left\|\nabla^2 Q\right\|_F^2 + \|\nabla Q\|_2^2$ or $\left\|\nabla^2 Q\right\|_F^2$. Bandeira, Scheinberg, and Vicente [153] discussed obtaining underdetermined quadratic interpolation models by minimizing $\left\|\text{vec}\left(\nabla^2 Q\right)\right\|_1$ for problems with sparse structure. Here the symbol vec denotes vectorization of a matrix, converting a matrix into a vector. Specifically, for a matrix $\boldsymbol{C} \in \mathfrak{R}^{m\times n}$, vec($\boldsymbol{C}$) denotes the vector obtained by stacking the columns of $\boldsymbol{C}$.

As shown by Conn, Scheinberg, and Vicente [20], in order to ensure an interpolation model with fully linear properties (see Definition 6.1 in the monograph by Conn, Scheinberg, and Vicente [20]), at least $n+1$ interpolation points are needed, which is expensive in derivative-free applications. In practice, we find that the $n+1$ interpolation conditions, i.e., $n+1$ equality constraints, can be relaxed to "making the average interpolation error small overall within a certain region." We use the H^2 norm to measure and control the interpolation error. Our proposed least H^2 norm updated quadratic model function can reduce the lower bound on the number of interpolation points required to construct the model, while also controlling interpolation errors locally. This is the first time the H^2 norm is used to construct underdetermined quadratic models in derivative-free methods or trust-region methods. Previously, Zhang [46, 154] discussed the related use of the H^1 seminorm, and more details will be shown at the beginning of Section 2.1.2.

The remainder of this part is organized as follows. Section 2.1.1 introduces some basic results on the H^2 norm for quadratic functions. Section 2.1.2 discusses the motivation for using the least H^2 norm updated quadratic model functions, the projection

theory under the H^2 norm, and interpolation error bounds. Section 2.1.3 presents the least H^2 norm updated quadratic model function and details the KKT conditions of the optimization problem required to obtain the quadratic model. In addition, we provide implementation details, including the updating formula for the inverse of the KKT matrix, as well as a model-improvement step that maximizes the denominator in the update formula. Section 2.1.4 presents numerical results. Finally, we give a summary and some possible future work.

2.1.1 H^2 Norm and Derivative-Free Trust-Region Algorithms

We first give the relevant notation and then proceed with more discussion. Here we define $\Omega = \mathcal{B}_r(\boldsymbol{x}_0) = \left\{\boldsymbol{x} \in \mathfrak{R}^n : \left\|\boldsymbol{x} - \boldsymbol{x}_0\right\|_2 \leq r\right\}$, $\boldsymbol{H} = \nabla^2 Q \in \mathfrak{R}^{n\times n}$, $\boldsymbol{g} = \nabla Q(\boldsymbol{x}_0) \in \mathfrak{R}^n$, $c \in \mathfrak{R}$. Note that, unless otherwise stated, $\|\cdot\|$ denotes the ℓ_2 norm for vectors. Below we give the definitions of different kinds of (semi)norms.

Definition 2.1. *Assume u is a function on $\Omega \subseteq \mathfrak{R}^n$ and $1 \leq p < \infty$. If u is twice differentiable on Ω and for any natural number a with $a \leq 2$, we have $\frac{\partial^a u}{\partial \boldsymbol{x}^a} \in L^2(\Omega)$, then*

$$\|u\|_{H^0(\Omega)} = \left(\int_\Omega |u(\boldsymbol{x})|^2 \, d\boldsymbol{x}\right)^{\frac{1}{2}},$$

$$|u|_{H^1(\Omega)} = \left(\int_\Omega \|\nabla u(\boldsymbol{x})\|_2^2 \, d\boldsymbol{x}\right)^{\frac{1}{2}},$$

$$|u|_{H^2(\Omega)} = \left(\int_\Omega \left\|\nabla^2 u(\boldsymbol{x})\right\|_F^2 \, d\boldsymbol{x}\right)^{\frac{1}{2}}.$$

In addition, the H^2 norm of u is defined as

$$\|u\|_{H^2(\Omega)} = \left(\|u\|_{H^0(\Omega)}^2 + |u|_{H^1(\Omega)}^2 + |u|_{H^2(\Omega)}^2\right)^{\frac{1}{2}}.$$

Note that we use $|\cdot|$ to denote seminorms and $\|\cdot\|$ to denote norms (with corresponding subscripts). Based on Definition 2.1, a simple calculation yields the following theorem on the H^2 norm of quadratic functions.

Remark 2.1. The point $\boldsymbol{x}_0$ denotes the center of the region where the H^2 norm is computed. $\boldsymbol{x}_0$ is sometimes called the base point [94].

Theorem 2.2. *Given the quadratic function*

$$Q(\boldsymbol{x}) = \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{H}\left(\boldsymbol{x} - \boldsymbol{x}_0\right) + \left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{g} + c,$$

we have

$$\begin{aligned}\|Q\|_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))}^2 =& \mathcal{V}_n r^n \left[\left(\frac{r^4}{2(n+4)(n+2)} + \frac{r^2}{n+2} + 1\right)\|\boldsymbol{H}\|_F^2 + \left(\frac{r^2}{n+2} + 1\right)\|\boldsymbol{g}\|_2^2\right. \\ &\left. + \frac{r^4}{4(n+4)(n+2)}(\mathrm{Tr}(\boldsymbol{H}))^2 + \frac{r^2}{n+2} c\,\mathrm{Tr}(\boldsymbol{H}) + c^2\right],\end{aligned} \tag{2-2}$$

where $\mathcal{V}_n$ denotes the volume of the n-dimensional ℓ_2 unit ball $\mathcal{B}_1(\boldsymbol{x}_0)$, and $\mathrm{Tr}(\cdot)$ *denotes the trace of a matrix.*

Proof. By direct calculation (see Zhang [46] for details)

$$\begin{aligned}\|Q(\boldsymbol{x})\|^2_{H^0(\mathcal{B}_r(\boldsymbol{x}_0))} =& \mathcal{V}_n r^n \left(\frac{2r^4}{4(n+2)(n+4)} \|\boldsymbol{H}\|_F^2 + \frac{r^2}{n+2} \|\boldsymbol{g}\|_2^2 \right. \\ &\left. + \frac{r^4}{4(n+2)(n+4)} (\mathrm{Tr}(\boldsymbol{H}))^2 + \frac{r^2}{n+2} c\,\mathrm{Tr}(\boldsymbol{H}) + c^2 \right),\end{aligned}$$

and

$$|Q|^2_{H^1(\mathcal{B}_r(\boldsymbol{x}_0))} = \mathcal{V}_n r^n \left(\frac{r^2}{n+2} \|\boldsymbol{H}\|_F^2 + \|\boldsymbol{g}\|_2^2 \right).$$

Moreover, we have $|Q|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} = \mathcal{V}_n r^n \|\boldsymbol{H}\|_F^2$. Therefore, (2-2) follows. □

Considering that the interpolation model functions given in this section are applicable to general model-based derivative-free trust-region algorithms based on interpolation models, before giving more details we first present the general framework of a model-based derivative-free trust-region algorithm[1], Algorithm 3. Note that this section does not propose a new framework, but focuses on a new quadratic model (which can be used in a general trust-region framework). Here, we still keep the wording "accept the model" rather than "accept the interpolation set" for consistency. More introductions to derivative-free trust-region methods can be found in the survey by Larson, Menickelly, and Wild [128], and in the monographs by Conn, Scheinberg, and Vicente [20] and by Audet and Hare [21].

Algorithm 3 Framework of a Model-Based Derivative-Free Trust-Region Algorithm

Input: black-box objective function f and initial point $\boldsymbol{x}_{\text{int}}$.

Output: minimizer $\boldsymbol{x}^*$ and least function value.

Initialize and obtain the interpolation set $\mathcal{X}_0$, the initial quadratic model $Q_0(\boldsymbol{x})$ (whose gradient at the current point is denoted by $\boldsymbol{g}_0$), and the parameters $\Delta_0, \gamma, \varepsilon_c, \mu, \hat{\eta}_1, \hat{\eta}_2$. Let $k = 0$.

Step 1 (**Acceptance test**): If $\|\boldsymbol{g}_k\| > \varepsilon_c$, then accept $\boldsymbol{g}_k$, Δ_k, and the model Q_k. If $\|\boldsymbol{g}_k\| \le \varepsilon_c$, then call the model-improvement step and test whether the current model on the trust region is accepted. If the model Q_k cannot be accepted or $\Delta_k > \mu \|\boldsymbol{g}_k\|$, then construct an acceptable model using the model-improvement step and accordingly adjust the radius Δ_k.

Step 2 (**Trial step**): Solve

$$\begin{aligned}&\min_{\boldsymbol{d}} Q_k(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \le \Delta_k,\end{aligned}$$

[1]The testing code framework in this section is based on Algorithm 10.1 in the monograph by Conn, Scheinberg, and Vicente [20]. This section focuses on our new quadratic model and its computation.

where $\boldsymbol{x}_{\text{opt}}$ is the point with the smallest function value among all interpolation points at the current iteration, and obtain $\boldsymbol{d}_k$.

Step 3 (**Acceptance of the trial point**): Compute $f(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}_k)$ and define

$$\rho_k = \frac{f(\boldsymbol{x}_{\text{opt}}) - f(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}_k)}{Q_k(\boldsymbol{x}_{\text{opt}}) - Q_k(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}_k)}.$$

If $\rho_k \geq \hat{\eta}_1$, or $\rho_k > \hat{\eta}_2$ and the model is accepted, then let $\boldsymbol{x}_{k+1} = \boldsymbol{x}_{\text{opt}} + \boldsymbol{d}_k$, update the model and the sample/interpolation set, and obtain Q_{k+1} (the gradient at the current point $\boldsymbol{g}_{k+1}$) and $\mathcal{X}_{k+1} = \mathcal{X}_k \cup \{\boldsymbol{y}_{\text{new}}\} \backslash \{\boldsymbol{y}_t\}$, where $\boldsymbol{y}_{\text{new}} = \boldsymbol{x}_{k+1}$ is the new iterate/interpolation point, and the farthest point $\boldsymbol{y}_t$ is discarded from the interpolation set in this step. Otherwise, keep the model and the iterate unchanged, and set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$.

Step 4 (**Model-improvement step**): If $\rho_k < \hat{\eta}_1$ and the model is not accepted, then improve the model. (The test version iteratively updates the inverse KKT matrix after checking acceptance based on interpolation poisedness, following the description at the end of Section 2.1.3. More details are given in Chapter 6 of the monograph by Conn, Scheinberg, and Vicente [20].) Define Q_{k+1} and $\mathcal{X}_{k+1}$ as the (possibly improved) new model and sample/interpolation set.

Step 5 (**Update of the trust-region radius**): Update Δ_{k+1} according to ρ_k and Δ_k. For example, if $\rho_k < \hat{\eta}_1$ and the model is accepted, set $\Delta_{k+1} = \frac{1}{\gamma}\Delta_k$; if $\rho_k \geq \hat{\eta}_1$, set $\Delta_{k+1} = \gamma\Delta_k$; in other cases, set $\Delta_{k+1} = \Delta_k$.

Let $k = k + 1$, then return to **Step 1**, until $\Delta_k < \varepsilon_c$ and $\|\boldsymbol{g}_k\| < \varepsilon_c$.

2.1.2 Motivation and Properties of Constructing Least Norm Quadratic Models Using H^2 Norm

This section introduces the motivation for obtaining approximation models of the objective function using the H^2 norm. We focus on the relationship between norm measurements at points and norm measurements in an average or global sense over a region. In addition, Powell and others proposed obtaining least Frobenius norm updated quadratic model functions by solving (2-1), demonstrating the advantages of least norm updated quadratic models. Note that, for Powell's model, there is a lower bound on the number of interpolation points or equations [94]. Moreover, most existing models are obtained solely by interpolation at multiple points, without considering the average of the objective function or its derivatives over a (trust) region. Before introducing how to obtain our new model, we first present the convexity of the objective function (2-3).

Theorem 2.3. *Given $C_1, C_2, C_3 > 0$, the function*

$$C_1 \|Q(\boldsymbol{x})\|^2_{H^0(\Omega)} + C_2 |Q(\boldsymbol{x})|^2_{H^1(\Omega)} + C_3 |Q(\boldsymbol{x})|^2_{H^2(\Omega)} \tag{2-3}$$

is strictly convex as a function of Q.

Proof. We have

$$\begin{aligned}
&C_1 \|Q(\boldsymbol{x})\|^2_{H^0(\Omega)} + C_2 |Q(\boldsymbol{x})|^2_{H^1(\Omega)} + C_3 |Q(\boldsymbol{x})|^2_{H^2(\Omega)} \\
=& C_1 \int_\Omega |Q(\boldsymbol{x})|^2 \, d\boldsymbol{x} + C_2 \int_\Omega \|\nabla Q(\boldsymbol{x})\|_2^2 \, d\boldsymbol{x} + C_3 \int_\Omega \left\|\nabla^2 Q(\boldsymbol{x})\right\|_F^2 d\boldsymbol{x},
\end{aligned}$$

and we need to prove the inequality

$$\begin{aligned}
&\Big[\mu \left(C_1 \left\|Q_a(\boldsymbol{x})\right\|^2_{H^0(\Omega)} + C_2 \left|Q_a(\boldsymbol{x})\right|^2_{H^1(\Omega)} + C_3 \left|Q_a(\boldsymbol{x})\right|^2_{H^2(\Omega)}\right) \\
&+ (1-\mu) \left(C_1 \left\|Q_b(\boldsymbol{x})\right\|^2_{H^0(\Omega)} + C_2 \left|Q_b(\boldsymbol{x})\right|^2_{H^1(\Omega)} + C_3 \left|Q_b(\boldsymbol{x})\right|^2_{H^2(\Omega)}\right)\Big] \\
&- \Big[C_1 \left\|\mu Q_a(\boldsymbol{x}) + (1-\mu) Q_b(\boldsymbol{x})\right\|^2_{H^0(\Omega)} + C_2 \left|\mu Q_a(\boldsymbol{x}) + (1-\mu) Q_b(\boldsymbol{x})\right|^2_{H^1(\Omega)} \\
&+ C_3 \left|\mu Q_a(\boldsymbol{x}) + (1-\mu) Q_b(\boldsymbol{x})\right|^2_{H^2(\Omega)}\Big] > 0
\end{aligned} \tag{2-4}$$

for $Q_a, Q_b \in \mathcal{Q}$, $0 < \mu < 1$, and $C_1, C_2, C_3 > 0$.

In fact, the left-hand side of inequality (2-4) can be rewritten as

$$\begin{aligned}
&\left(\mu - \mu^2\right) \left(C_1 \int_\Omega \left|Q_a(\boldsymbol{x}) - Q_b(\boldsymbol{x})\right|^2 d\boldsymbol{x} + C_2 \int_\Omega \left\|\nabla Q_a(\boldsymbol{x}) - \nabla Q_b(\boldsymbol{x})\right\|_2^2 d\boldsymbol{x}\right. \\
&\left. + C_3 \int_\Omega \left\|\nabla^2 Q_a(\boldsymbol{x}) - \nabla^2 Q_b(\boldsymbol{x})\right\|_F^2 d\boldsymbol{x}\right) > 0.
\end{aligned}$$

Therefore, we have proved that (2-3) is strictly convex as a function of Q. □

By Theorem 2.3, the optimization problem

$$\begin{aligned}
&\min_{Q \in \mathcal{Q}} \ \left\|Q - Q_{k-1}\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&\text{s. t. } Q(\boldsymbol{y}) = f(\boldsymbol{y}), \ \boldsymbol{y} \in \mathcal{X}_k
\end{aligned} \tag{2-5}$$

has a unique quadratic model solution $Q(\boldsymbol{x})$.

In addition, we can obtain the following result.

Remark 2.2. For the optimization problem of obtaining the least weighted H^2 norm updated model

$$\begin{aligned}
&\min_{Q \in \mathcal{Q}} C_1 \left\|Q - Q_{k-1}\right\|^2_{H^0(\mathcal{B}_r(\boldsymbol{x}_0))} + C_2 \left|Q - Q_{k-1}\right|^2_{H^1(\mathcal{B}_r(\boldsymbol{x}_0))} + C_3 \left|Q - Q_{k-1}\right|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&\text{s. t. } Q(\boldsymbol{y}) = f(\boldsymbol{y}), \ \boldsymbol{y} \in \mathcal{X}_k,
\end{aligned} \tag{2-6}$$

the solution is unique, where $C_1, C_2, C_3 > 0$.

It is easy to see that the H^1 seminorm of a quadratic model function on $\mathcal{B}_r(\boldsymbol{x}_0)$ corresponds to the average of the corresponding norm of the first derivative over the region $\mathcal{B}_r(\boldsymbol{x}_0)$, while the H^2 seminorm on $\mathcal{B}_r(\boldsymbol{x}_0)$ corresponds to the average of the

corresponding norm of the second derivative over $\mathcal{B}_r(\boldsymbol{x}_0)$. In fact, minimizing the interpolation error of the quadratic model function values is also important. Minimizing the L^2 norm can play part of the role of relaxing the interpolation conditions $Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i)$ by reducing the interpolation error in function values, thereby relaxing the lower bound on the number of interpolation points. According to the projection property to be introduced next, we indeed have reason to include the H^0 norm, i.e., the L^2 norm, in the objective function of (2-6). The (weighted) sum of the H^0 norm, the H^1 seminorm, and the H^2 seminorm embodies our goal of simultaneously minimizing the average model error in function values, the average error of first derivatives, and the average error of second derivatives. This can reduce the lower bound on the number of interpolation points, i.e., m can be smaller than $n+1$. The numerical results in Section 2.1.4 also support our choice of using the H^2 norm to obtain the model function.

Below we present the projection property of the least H^2 norm updated quadratic model and prove that it has an interpolation error bound locally.

Theorem 2.4. *Let Q_k be the solution to problem (2-5). If f is a quadratic function, then*

$$\left\|Q_k - f\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} = \left\|Q_{k-1} - f\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} - \left\|Q_k - Q_{k-1}\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))}. \tag{2-7}$$

Proof. At the k-th iteration, for any $\xi \in \mathfrak{R}$, let $Q_\xi = Q_k + \xi\left(Q_k - f\right)$. Then Q_ξ is an interpolation function that satisfies the interpolation conditions in (2-5). Therefore, by the optimality of Q_k, $\varphi(\xi) = \left\|Q_\xi - Q_{k-1}\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))}$ attains its minimum at $\xi = 0$. Moreover, we have

$$\begin{aligned}
\varphi(\xi) &= \left\|Q_k + \xi\left(Q_k - f\right) - Q_{k-1}\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&= \xi^2\left\|Q_k - f\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} + \left\|Q_k - Q_{k-1}\right\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&+ 2\xi\Big\{\int_{\mathcal{B}_r(\boldsymbol{x}_0)} \left(Q_k(\boldsymbol{x}) - Q_{k-1}(\boldsymbol{x})\right)\cdot\left(Q_k(\boldsymbol{x}) - f(\boldsymbol{x})\right)d\boldsymbol{x} \\
&+ \int_{\mathcal{B}_r(\boldsymbol{x}_0)} \left(\nabla Q_k(\boldsymbol{x}) - \nabla Q_{k-1}(\boldsymbol{x})\right)^\top\left(\nabla Q_k(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\right)d\boldsymbol{x} \\
&+ \int_{\mathcal{B}_r(\boldsymbol{x}_0)} (1,\cdots,1)\left(\nabla^2 Q_k(\boldsymbol{x}) - \nabla^2 Q_{k-1}(\boldsymbol{x})\right)\circ\left(\nabla^2 Q_k(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{x})\right)(1,\cdots,1)^\top d\boldsymbol{x}\Big\},
\end{aligned} \tag{2-8}$$

where the symbol $\circ$ denotes the Hadamard product. Therefore the terms in the last bracket of (2-8) are equal to 0. Considering $\varphi(-1)$ completes the proof.

□

From (2-7), we obtain the relation in the H^2 norm sense

$$\left\|Q_k(\boldsymbol{x}) - f(\boldsymbol{x})\right\|_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \le \left\|Q_{k-1}(\boldsymbol{x}) - f(\boldsymbol{x})\right\|_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))}.$$

This implies, to some extent, that the model function Q_k has more accurate function values and gradients with respect to the approximated objective function f than Q_{k-1} (except in the case $Q_k = Q_{k-1}$). In fact, we can directly obtain the following corollary from Theorem 2.4.

Corollary 2.5. *Let Q_k be the solution to problem (2-6). If f is a quadratic function, then*

$$\begin{aligned}
&C_1 \left\|Q_k - f\right\|^2_{H^0(\mathcal{B}_r(\boldsymbol{x}_0))} + C_2 \left|Q_k - f\right|^2_{H^1(\mathcal{B}_r(\boldsymbol{x}_0))} + C_3 \left|Q_k - f\right|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&= C_1 \left\|Q_{k-1} - f\right\|^2_{H^0(\mathcal{B}_r(\boldsymbol{x}_0))} + C_2 \left|Q_{k-1} - f\right|^2_{H^1(\mathcal{B}_r(\boldsymbol{x}_0))} + C_3 \left|Q_{k-1} - f\right|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\
&- C_1 \left\|Q_k - Q_{k-1}\right\|^2_{H^0(\mathcal{B}_r(\boldsymbol{x}_0))} - C_2 \left|Q_k - Q_{k-1}\right|^2_{H^1(\mathcal{B}_r(\boldsymbol{x}_0))} - C_3 \left|Q_k - Q_{k-1}\right|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))},
\end{aligned}$$

where $C_1, C_2, C_3 > 0$.

Proof. The proof is similar to that of Theorem 2.4. □

The approximation of the model function to the objective function is crucial for our model-based optimization algorithm. Next, we provide an error analysis for interpolation models. First, we give the following two lemmas.

Lemma 2.6 (Interpolation inequality)**.** *Assume $1 \le c_1 \le c_2 \le c_3 \le \infty$, and $\frac{1}{c_2} = \frac{\theta}{c_1} + \frac{(1-\theta)}{c_3}$. Assume $u \in L^{c_1}(\Omega) \cap L^{c_3}(\Omega)$. Then $u \in L^{c_2}(\Omega)$, and $\|u\|_{L^{c_2}(\Omega)} \le \|u\|^{\theta}_{L^{c_1}(\Omega)} \|u\|^{1-\theta}_{L^{c_3}(\Omega)}$.*

Proof. See Evans'monograph [173] for details. □

Definition 2.7 (Sobolev spaces)**.** *The Sobolev space $\mathcal{W}^{k,p}(\Omega)$ contains all locally integrable functions $u : \Omega \to \Re$ such that for all multi-indices α with $|\alpha| \le k$, the weak derivatives $D^\alpha u$ exist[2] and belong to $L^p(\Omega)$. When $p = 2$, we usually write $\mathcal{H}^k(\Omega) = \mathcal{W}^{k,2}(\Omega)$.*

Lemma 2.8. *Assume $u \in \mathcal{H}^1(\Omega)$ and $\left|\partial_i u\right| \le M_1$, $i = 1, \cdots, n$. For $\forall\, \boldsymbol{y} \in \Omega := \mathcal{B}_r(\boldsymbol{x}_0)$, we have*

$$|u(\boldsymbol{y})| \le \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} \|u\|_{L^2(\Omega)} + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_1^{1-\theta} |u|^{\theta}_{H^1(\Omega)}, \tag{2-9}$$

where $\frac{1}{p} + \frac{1}{q} = 1$, $\theta = \frac{2}{p} \le 1$, $p > n$, and $\mathcal{V}_n$ denotes the volume of the n-dimensional ℓ_2 unit ball $\mathcal{B}_1(\boldsymbol{x}_0)$.

[2] More details are given in Evans'monograph [173].

Proof. We extend the function u outside the region Ω so that for $\forall\ \boldsymbol{y} \notin \mathcal{B}_r(\boldsymbol{x}_0)$, $u(\boldsymbol{y}) = 0$. For $\forall\ \boldsymbol{y} \in \mathcal{B}_r\left(\boldsymbol{x}_0\right)$, we have

$$\begin{aligned} |u(\boldsymbol{y})| &\leq \left| u(\boldsymbol{y}) - \frac{1}{\left|\mathcal{B}_r(\boldsymbol{y})\right|} \int_{\mathcal{B}_r(\boldsymbol{y})} u(\boldsymbol{x}) d\boldsymbol{x} \right| + \frac{1}{\left|\mathcal{B}_r(\boldsymbol{y})\right|} \left| \int_{\mathcal{B}_r(\boldsymbol{y})} u(\boldsymbol{x}) d\boldsymbol{x} \right| \\ &\leq \frac{1}{\left|\mathcal{B}_r(\boldsymbol{y})\right|} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} |u(\boldsymbol{y}) - u(\boldsymbol{x})|\, d\boldsymbol{x} + \left| \int_{\mathcal{B}_r(\boldsymbol{y})} u(\boldsymbol{x}) d\boldsymbol{x} \right| \right), \end{aligned} \tag{2-10}$$

where $\left|\mathcal{B}_r(\boldsymbol{y})\right|$ denotes the volume of the ball $\mathcal{B}_r(\boldsymbol{y})$. Moreover, both terms on the right-hand side of (2-10) have upper bounds. Based on Hölder's inequality, we have

$$\begin{aligned} \left| \int_{\mathcal{B}_r(\boldsymbol{y})} u(\boldsymbol{x}) d\boldsymbol{x} \right| &\leq \left(\int_{\mathcal{B}_r(\boldsymbol{y})} (u(\boldsymbol{x}))^2 d\boldsymbol{x} \right)^{\frac{1}{2}} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} 1^2 d\boldsymbol{x} \right)^{\frac{1}{2}} \\ &\leq \left|\mathcal{B}_r(\boldsymbol{y})\right|^{\frac{1}{2}} \|u\|_{L^2(\Omega)} = \mathcal{V}_n^{\frac{1}{2}} r^{\frac{n}{2}} \|u\|_{L^2(\Omega)}. \end{aligned}$$

According to the proof of Morrey's inequality in Evans'monograph [173], we have

$$\begin{aligned} \int_{\mathcal{B}_r(\boldsymbol{y})} |u(\boldsymbol{y}) - u(\boldsymbol{x})|\, d\boldsymbol{x} &\leq \frac{r^n}{n} \int_{\mathcal{B}_r(\boldsymbol{y})} \frac{\|\nabla u(\boldsymbol{x})\|_2}{\|\boldsymbol{y} - \boldsymbol{x}\|_2^{n-1}} d\boldsymbol{x} \\ &\leq \frac{r^n}{n} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} \|\nabla u(\boldsymbol{x})\|_2^p\, d\boldsymbol{x} \right)^{\frac{1}{p}} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} \frac{1}{\|\boldsymbol{y} - \boldsymbol{x}\|_2^{(n-1)q}} d\boldsymbol{x} \right)^{\frac{1}{q}}, \end{aligned}$$

where $\frac{1}{p} + \frac{1}{q} = 1$ and $(n-1)(q-1) \in (0, 1)$. We obtain

$$\begin{aligned} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} \frac{1}{\|\boldsymbol{y} - \boldsymbol{x}\|_2^{(n-1)q}} d\boldsymbol{x} \right)^{\frac{1}{q}} &= \left(\int_{\mathcal{B}_r(\boldsymbol{0})} \frac{1}{\boldsymbol{z}^{(n-1)q}} d\boldsymbol{z} \right)^{\frac{1}{q}} \\ &= \mathcal{V}_n^{\frac{1}{q}} n^{\frac{1}{q}} (n + q - nq)^{-\frac{1}{q}} r^{\frac{n}{q}+1-n}. \end{aligned}$$

In addition, Lemma 2.6 helps us obtain

$$\begin{aligned} \left(\int_{\mathcal{B}_r(\boldsymbol{y})} \|\nabla u(\boldsymbol{x})\|_2^p\, d\boldsymbol{x} \right)^{\frac{1}{p}} &= \left\| \left(\sum_{i=1}^n \left|\partial_i u(\boldsymbol{x})\right|^2 \right)^{\frac{1}{2}} \right\|_{L^p(\mathcal{B}_r(\boldsymbol{y}))} \\ &\leq \left\| \left(\sum_{i=1}^n \left|\partial_i u(\boldsymbol{x})\right|^2 \right)^{\frac{1}{2}} \right\|_{L^p(\Omega)} \\ &\leq \left\| \left(\sum_{i=1}^n \left|\partial_i u(\boldsymbol{x})\right|^2 \right)^{\frac{1}{2}} \right\|_{L^2(\Omega)}^{\theta} \left\| \left(\sum_{i=1}^n \left|\partial_i u(\boldsymbol{x})\right|^2 \right)^{\frac{1}{2}} \right\|_{L^\infty(\Omega)}^{1-\theta} \\ &\leq n^{\frac{1-\theta}{2}} |u|_{H^1(\Omega)}^{\theta} M_1^{1-\theta}, \end{aligned}$$

where $\theta = \frac{2}{p} \leq 1$ and $p > n$. Hence,

$$\begin{aligned}\int_{\mathcal{B}_r(\boldsymbol{y})} |u(\boldsymbol{y}) - u(\boldsymbol{x})|\, d\boldsymbol{x} &\leq \frac{r^n}{n} n^{\frac{1-\theta}{2}} |u|^{\theta}_{H^1(\Omega)} M_1^{1-\theta} \mathcal{V}_n^{\frac{1}{q}} n^{\frac{1}{q}} (n+q-nq)^{-\frac{1}{q}} r^{\frac{n}{q}+1-n} \\ &= \mathcal{V}_n^{\frac{1}{q}} r^{\frac{n}{q}+1} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_1^{1-\theta} |u|^{\theta}_{H^1(\Omega)}.\end{aligned}$$

Therefore,

$$\begin{aligned}|u(\boldsymbol{y})| &\leq \frac{1}{\mathcal{V}_n r^n} \left[r^{\frac{n}{q}+1} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} |u|^{\theta}_{H^1(\Omega)} M_1^{1-\theta} \mathcal{V}_n^{\frac{1}{q}} + \mathcal{V}_n^{\frac{1}{2}} r^{\frac{n}{2}} \|u\|_{L^2(\Omega)} \right] \\ &= \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} \|u\|_{L^2(\Omega)} + \mathcal{V}_n^{\frac{1}{q}-1} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_1^{1-\theta} |u|^{\theta}_{H^1(\Omega)}.\end{aligned}$$

Hence (2-9) holds. □

The following theorem illustrates the relationship between the H^2 norm of a function in a given region and the absolute value of the function as well as the norm of its gradient at a point.

Theorem 2.9. *Let $u \in \mathcal{H}^2(\Omega)$, where $\Omega = \mathcal{B}_r(\boldsymbol{x}_0)$. Suppose there exist M_1, M_2 such that $\left|\partial_i u\right| \leq M_1$, $\left|\partial^2_{ij} u\right| \leq M_2$, $i, j = 1, \cdots, n$, then for $\forall\ \boldsymbol{x} \in \Omega$, we have*

$$|u(\boldsymbol{x})| \leq \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} \|u\|_{L^2(\Omega)} + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_1^{1-\theta} |u|^{\theta}_{H^1(\Omega)}, \quad (2\text{-}11)$$

$$\|\nabla u(\boldsymbol{x})\|_2 \leq n^{\frac{1}{2}} \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} |u|_{H^1(\Omega)} + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_2^{1-\theta} |u|^{\theta}_{H^2(\Omega)}, \quad (2\text{-}12)$$

where $\frac{1}{p} + \frac{1}{q} = 1$, $\theta = \frac{2}{p} \leq 1$, $p > n$, $\mathcal{B}_r(\boldsymbol{x}_0) = \left\{ \boldsymbol{x} \in \mathfrak{R}^n : \left\| \boldsymbol{x} - \boldsymbol{x}_0 \right\|_2 \leq r \right\}$, $\mathcal{V}_n$ denotes the volume of the n-dimensional ℓ_2 unit ball $\mathcal{B}_1(\boldsymbol{x}_0)$.

Proof. We can directly obtain (2-11) from Lemma 2.8. For $\forall\ \boldsymbol{x} \in \Omega := \mathcal{B}_r(\boldsymbol{x}_0)$, we have

$$|\partial_i u(\boldsymbol{x})| \leq \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} |u|_{H^1(\Omega)} + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_2^{1-\theta} |u|^{\theta}_{H^2(\Omega)},$$

where $\frac{1}{p} + \frac{1}{q} = 1$, $\theta = \frac{2}{p} \leq 1$, and $p > n$. Therefore, we obtain

$$\begin{aligned}\|\nabla u\|_2 &\leq n^{\frac{1}{2}} \max_{i=1,\cdots,n} \left\| \partial_i u \right\|_{L^\infty(\Omega)} \\ &\leq n^{\frac{1}{2}} \left[\mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} |u|_{H^1(\Omega)} + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_2^{1-\theta} |u|^{\theta}_{H^2(\Omega)} \right],\end{aligned}$$

which gives (2-12). □

According to Theorem 2.9, the following corollary can be naturally derived. We can observe that reducing the H^2 norm of a function u also reduces the absolute value of u and the norm of its gradient vector.

Corollary 2.10. *Given an objective function $f \in \mathcal{H}^2(\Omega$ and its quadratic model function Q, suppose $\left|\partial_i(Q-f)\right| \leq M_1$, $\left|\partial^2_{ij}(Q-f)\right| \leq M_2$, $i,j = 1,\cdots,n$. Then for $\forall\, \boldsymbol{x} \in \Omega$, we have*

$$
\begin{aligned}
|Q(\boldsymbol{x}) - f(\boldsymbol{x})| &\leq \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} \|Q-f\|_{L^2(\Omega)} \\
&\quad + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{1+\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_1^{1-\theta} |Q-f|^{\theta}_{H^1(\Omega)}, \\
\|\nabla Q(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|_2 &\leq n^{\frac{1}{2}} \mathcal{V}_n^{-\frac{1}{2}} r^{-\frac{n}{2}} |Q-f|_{H^1(\Omega)} \\
&\quad + \mathcal{V}_n^{-\frac{1}{p}} r^{\frac{n}{q}+1-n} n^{\frac{1}{q}-\frac{\theta}{2}} (n+q-nq)^{-\frac{1}{q}} M_2^{1-\theta} |Q-f|^{\theta}_{H^2(\Omega)},
\end{aligned}
$$

where $\frac{1}{p}+\frac{1}{q}=1$, $\theta = \frac{2}{p} \leq 1$, $p > n$, and $\mathcal{V}_n$ denotes the volume of the n-dimensional ℓ_2 unit ball $\mathcal{B}_1(\boldsymbol{x}_0)$.

Proof. Replacing u by $Q-f$ in Theorem 2.9 gives the proof. □

The error analysis in Corollary 2.10 theoretically examines the approximation performance of the least H^2 norm updated quadratic model. Therefore, we conclude that obtaining the model function by solving subproblem (2-5) or subproblem (2-6) can relax the minimum requirement of interpolation conditions, i.e., it allows us to use fewer interpolation points while maintaining good approximation properties.

2.1.3 Least H^2 Norm Updating Quadratic Model

In this section, we present the computational approach to obtain the least H^2 norm updating quadratic model based on the KKT conditions. Theorem 2.2 and its proof help us derive, at iteration k, the coefficients of the quadratic model function by solving the problem

$$
\begin{aligned}
&\min_{c,\boldsymbol{g},\boldsymbol{H}} \ \eta_1 \|\boldsymbol{H}\|_F^2 + \eta_2 \|\boldsymbol{g}\|_2^2 + \eta_3 (\mathrm{Tr}(\boldsymbol{H}))^2 + \eta_4 \,\mathrm{Tr}(\boldsymbol{H}) c + \eta_5 c^2 \\
&\text{s. t. } c + \boldsymbol{g}^\top \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right) + \frac{1}{2}\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top \boldsymbol{H} \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right) = f(\boldsymbol{y}_i) - Q_{k-1}(\boldsymbol{y}_i), i = 1,\cdots,m
\end{aligned}
\tag{2-13}
$$

where $\boldsymbol{H}^\top = \boldsymbol{H}$, and the solution of (2-13) gives the coefficients of the model difference $Q_k - Q_{k-1}$. The choice of the radius r used in computing the H^2 norm in the experiments will be given in Section 2.1.4. For simplicity, we use the points $\boldsymbol{y}_1,\cdots,\boldsymbol{y}_m$ to denote the interpolation points at the k-th iteration. We directly consider a weighted

objective function with weight coefficients C_1, C_2, and C_3. Note that the coefficients $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ satisfy

$$\begin{cases} \eta_1 = C_1 \dfrac{r^4}{2(n+4)(n+2)} + C_2 \dfrac{r^2}{n+2} + C_3, \\ \eta_2 = C_1 \dfrac{r^2}{n+2} + C_2, \\ \eta_3 = C_1 \dfrac{r^4}{4(n+4)(n+2)}, \\ \eta_4 = C_1 \dfrac{r^2}{n+2}, \\ \eta_5 = C_1. \end{cases} \tag{2-14}$$

We know that the Lagrangian function corresponding to problem (2-13) is

$$\begin{aligned} &\mathcal{L}(c, \boldsymbol{g}, \boldsymbol{H}) = \eta_1 \|\boldsymbol{H}\|_F^2 + \eta_2 \|\boldsymbol{g}\|_2^2 + \eta_3 (\mathrm{Tr}(\boldsymbol{H}))^2 + \eta_4 \,\mathrm{Tr}(\boldsymbol{H}) c + \eta_5 c^2 \\ &- \sum_{i=1}^{m} \lambda_i \left[c + \boldsymbol{g}^\top \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right) + \frac{1}{2} \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right)^\top \boldsymbol{H} \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right) - f(\boldsymbol{y}_i) + Q_{k-1}(\boldsymbol{y}_i) \right]. \end{aligned} \tag{2-15}$$

We use T to denote $\mathrm{Tr}(\boldsymbol{H})$. The KKT conditions for problem (2-13) include

$$\begin{aligned} 0 &= \frac{\partial \mathcal{L}(c, \boldsymbol{g}, \boldsymbol{H})}{\partial c} = 2\eta_5 c + \eta_4 T - \sum_{i=1}^{m} \lambda_i, \\ \boldsymbol{0}_n &= \frac{\partial \mathcal{L}(c, \boldsymbol{g}, \boldsymbol{H})}{\partial \boldsymbol{g}} = 2\eta_2 \boldsymbol{g} - \sum_{i=1}^{m} \lambda_i \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right), \end{aligned}$$

where $\boldsymbol{0}_n = (0, \cdots, 0)^\top \in \Re^n$. Other equations in the KKT conditions are given below. Taking derivatives of $\mathcal{L}(c, \boldsymbol{g}, \boldsymbol{H})$ with respect to the elements of $\boldsymbol{H}$, we obtain

$$2\eta_1 \boldsymbol{H} - \frac{1}{2} \sum_{l=1}^{m} \lambda_l \left(\boldsymbol{y}_l - \boldsymbol{x}_0 \right) \left(\boldsymbol{y}_l - \boldsymbol{x}_0 \right)^\top + 2\eta_3 \,\mathrm{Diag}\{T, \cdots, T\} + \eta_4 c \boldsymbol{I} = \boldsymbol{0}_{nn}.$$

Thus,

$$2\eta_1 \boldsymbol{H} = \frac{1}{2} \sum_{i=1}^{m} \lambda_i \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right) \left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right)^\top - \left(2\eta_3 T + \eta_4 c \right) \boldsymbol{I}. \tag{2-16}$$

By multiplying both sides of (2-16) on the left and right by $\left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right)^\top$ and $\left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right)$, we obtain

$$\begin{aligned} &2\eta_1 \left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right)^\top \boldsymbol{H} \left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right) \\ &= \frac{1}{2} \sum_{i=1}^{m} \lambda_i \left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right) \right]^2 - \left(2\eta_3 T + \eta_4 c \right) \left\| \boldsymbol{y}_j - \boldsymbol{x}_0 \right\|_2^2, \ 1 \le j \le m. \end{aligned}$$

Furthermore, by multiplying $\boldsymbol{H}$ on the left and right by $\boldsymbol{e}_j^\top$ and $\boldsymbol{e}_j$, where the vector $\boldsymbol{e}_j$ is defined as the j-th column of the identity matrix $\boldsymbol{I}$, we obtain

$$2\eta_1 \boldsymbol{e}_j^\top \boldsymbol{H} \boldsymbol{e}_j = \frac{1}{2}\sum_{i=1}^{m} \lambda_i \left[\boldsymbol{e}_j^\top \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)\right]^2 - \left(2\eta_3 T + \eta_4 c\right) \boldsymbol{e}_j^\top \boldsymbol{e}_j,\ 1 \le j \le n. \tag{2-17}$$

By summing (2-17) over $j = 1, \ldots, n$, we obtain

$$2\eta_1 T = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} \lambda_l \left[\boldsymbol{e}_j^\top \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)\right]^2 - n\left(2\eta_3 T + \eta_4 c\right),$$

which gives

$$0 = \frac{1}{2}\sum_{i=1}^{m} \lambda_i \left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 - (2n\eta_3 + 2\eta_1)T - n\eta_4 c.$$

Thus, we obtain the expression of T:

$$T = \frac{1}{2\left(2n\eta_3 + 2\eta_1\right)}\sum_{i=1}^{m} \lambda_i \left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 - \frac{n\eta_4}{2n\eta_3 + 2\eta_1} c. \tag{2-18}$$

Combining with the constraints in (2-13), we obtain the system of equations in terms of $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_m)^\top, c, \boldsymbol{g}$:

$$\begin{aligned}
0 &= 2\eta_5 c + \frac{\eta_4}{4n\eta_3 + 4\eta_1}\sum_{i=1}^{m} \lambda_i \left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 - \frac{n\eta_4^2}{2n\eta_3 + 2\eta_1} c - \sum_{i=1}^{m} \lambda_i, \\
\boldsymbol{0}_n &= 2\eta_2 \boldsymbol{g} - \sum_{i=1}^{m} \lambda_i \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right), \\
f(\boldsymbol{y}_j) - Q_{k-1}(\boldsymbol{y}_j) &= \frac{1}{8\eta_1}\sum_{i=1}^{m} \lambda_i \left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right]^2 \\
&\quad - \frac{\eta_3}{8\eta_1\left(n\eta_3 + \eta_1\right)}\sum_{i=1}^{m} \lambda_i \left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 \left\|\boldsymbol{y}_j - \boldsymbol{x}_0\right\|_2^2 \\
&\quad - \frac{\eta_4}{4n\eta_3 + 4\eta_1} c \left\|\boldsymbol{y}_j - \boldsymbol{x}_0\right\|_2^2 + c + \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)^\top \boldsymbol{g},\ j = 1, \cdots, m.
\end{aligned}$$

Since at the k-th iteration, $\boldsymbol{y}_t$ is replaced by $\boldsymbol{y}_{\text{new}}$, and $Q_k(\boldsymbol{y}_i) - Q_{k-1}(\boldsymbol{y}_i) = f(\boldsymbol{y}_i) - Q_{k-1}(\boldsymbol{y}_i)$, given all $\boldsymbol{y}_i$ in the current interpolation set, we obtain the system

$$\overbrace{\begin{pmatrix} \boldsymbol{A} & \boldsymbol{J} & \boldsymbol{X} \\ \boldsymbol{J}^\top & \frac{n\eta_4^2}{2n\eta_3+2\eta_1} - 2\eta_5 & \boldsymbol{0}_n^\top \\ \boldsymbol{X}^\top & \boldsymbol{0}_n & -2\eta_2 \boldsymbol{I} \end{pmatrix}}^{m+1+n} \begin{pmatrix} \boldsymbol{\lambda} \\ c \\ \boldsymbol{g} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ f(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}) \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{2-19}$$

where $\boldsymbol{y}_{\text{new}}$ denotes the new interpolation point, and $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_m)^\top$. In addition, the elements of $\boldsymbol{A}$ are given by

$$\boldsymbol{A}_{ij} = \frac{1}{8\eta_1}\left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right]^2 - \frac{\eta_3}{8\eta_1\left(n\eta_3 + \eta_1\right)}\left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 \left\|\boldsymbol{y}_j - \boldsymbol{x}_0\right\|_2^2,$$

for $1 \le i, j \le m$. Furthermore, $\boldsymbol{X} = \left(\boldsymbol{y}_1 - \boldsymbol{x}_0, \boldsymbol{y}_2 - \boldsymbol{x}_0, \cdots, \boldsymbol{y}_m - \boldsymbol{x}_0\right)^\top$, and

$$\boldsymbol{J} = \left(1 - \frac{\eta_4}{4n\eta_3 + 4\eta_1}\left\|\boldsymbol{y}_1 - \boldsymbol{x}_0\right\|_2^2, \cdots, 1 - \frac{\eta_4}{4n\eta_3 + 4\eta_1}\left\|\boldsymbol{y}_m - \boldsymbol{x}_0\right\|_2^2\right)^\top.$$

We call the matrix on the left-hand side of (2-19) the KKT matrix $\boldsymbol{W}$.

Based on the solution of (2-19) for $\boldsymbol{\lambda}, c, \boldsymbol{g}$, we can obtain the quadratic model function $Q(\boldsymbol{x})$. In fact, the least Frobenius norm updating quadratic model is a special case of the least H^2 norm updating quadratic model, as shown below.

Remark 2.3. If $C_1 = C_2 = 0, C_3 = 1$, then $\eta_1 = 1, \eta_2 = \eta_3 = \eta_4 = \eta_5 = 0$, and the KKT matrix is

$$\boldsymbol{W} = \begin{pmatrix} \bar{\boldsymbol{A}} & \boldsymbol{E} & \boldsymbol{X} \\ \boldsymbol{E}^\top & 0 & \boldsymbol{0}_n^\top \\ \boldsymbol{X}^\top & \boldsymbol{0}_n & \boldsymbol{0}_{nn} \end{pmatrix}, \tag{2-20}$$

where $\boldsymbol{E} \in \Re^m$ is $(1, \cdots, 1)^\top$, and $\bar{\boldsymbol{A}}_{ij} = \frac{1}{8}\left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right]^2$, $1 \le i, j \le m$.

In this case, the Hessian matrix corresponding to the interpolation points $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m$ is

$$\boldsymbol{H} = \frac{1}{4}\sum_{i=1}^{m} \lambda_i \left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top. \tag{2-21}$$

The $(m+n+1)\times(m+n+1)$ matrix in (2-20) is exactly the KKT matrix corresponding to the least Frobenius norm updating quadratic model [94]. Note that the coefficient $\frac{1}{4}$ in (2-21) depends on the coefficients in the Lagrangian function (2-15), but this does not affect the result.

To reduce computational complexity, we will discuss and use the updating formula of the inverse of the KKT matrix in what follows. Before discussing the inverse of the KKT matrix, we first introduce the following theorem, which gives the condition for the KKT matrix to be invertible.

Theorem 2.11. *The $(m+n+1)\times(m+n+1)$ matrix $\boldsymbol{W}$ is invertible if and only if the $(m+1)\times(m+1)$ matrix*

$$\begin{pmatrix} \boldsymbol{A} + \frac{1}{2\eta_2}\boldsymbol{X}\boldsymbol{X}^\top & \boldsymbol{J} \\ \boldsymbol{J}^\top & \frac{n\eta_4^2}{2n\eta_3 + 2\eta_1} - 2\eta_5 \end{pmatrix}$$

is invertible.

Proof. We have

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{J} & \boldsymbol{X} \\ \boldsymbol{J}^\top & \frac{n\eta_4^2}{2n\eta_3+2\eta_1} - 2\eta_5 & \boldsymbol{0}_n^\top \\ \boldsymbol{X}^\top & \boldsymbol{0}_n & -2\eta_2 \boldsymbol{I} \end{pmatrix} \rightarrow \begin{pmatrix} \boldsymbol{A} + \frac{1}{2\eta_2}\boldsymbol{X}\boldsymbol{X}^\top & \boldsymbol{J} & \boldsymbol{X} \\ \boldsymbol{J}^\top & \frac{n\eta_4^2}{2n\eta_3+2\eta_1} - 2\eta_5 & \boldsymbol{0}_n^\top \\ \boldsymbol{0}_{nm} & \boldsymbol{0}_n & -2\eta_2 \boldsymbol{I} \end{pmatrix},$$

where the arrow denotes elementary transformations. The conclusion then follows. □

Theorem 2.11 gives the necessary and sufficient condition for the KKT matrix to be invertible. In what follows, we refer to the inverse of the KKT matrix simply as the KKT inverse.

Note that directly solving the KKT system (2-19) at each iteration to obtain the parameters $\boldsymbol{\lambda}, c, \boldsymbol{g}$ of the quadratic model function is not numerically efficient, with computational complexity $\mathcal{O}((m+n)^3)$. We attempt to use an updating formula for the KKT inverse with lower computational complexity. Similar to the discussion given by Powell [94, 174], a natural question is what happens to the KKT matrix when the interpolation set is updated. In fact, when the interpolation set is updated, we find that $\boldsymbol{W}$ changes only in its t-th column and t-th row, because only $\boldsymbol{y}_t$ is replaced by $\boldsymbol{y}_{\text{new}}$. We borrow the updating formula for the KKT inverse given by Powell [174].

We define a vector $\boldsymbol{\omega} \in \mathfrak{R}^{m+n+1}$ whose components ω_i are given by

$$\begin{cases} \frac{1}{8\eta_1}\left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_{\text{new}} - \boldsymbol{x}_0\right)\right]^2 - \frac{\eta_3}{8\eta_1\left(n\eta_3+\eta_1\right)} \left\|\boldsymbol{y}_i - \boldsymbol{x}_0\right\|_2^2 \left\|\boldsymbol{y}_{\text{new}} - \boldsymbol{x}_0\right\|_2^2, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } 1 \le i \le m, \\ 1 - \frac{\eta_4}{4n\eta_3+4\eta_1}\left\|\boldsymbol{y}_{\text{new}} - \boldsymbol{x}_0\right\|_2^2, \text{if } i = m+1, \\ \left(\boldsymbol{y}_{\text{new}} - \boldsymbol{x}_0\right)_{i-m-1}, \text{if } m+2 \le i \le m+n+1. \end{cases}$$

If an invertible KKT matrix $\boldsymbol{W}$ has its t-th column and t-th row replaced respectively by the vector $\boldsymbol{\omega}$ and $\boldsymbol{\omega}^\top$, the new matrix is denoted as $\boldsymbol{W}_{\text{new}}$. Let $\boldsymbol{V}_{\text{new}} := \boldsymbol{W}_{\text{new}}^{-1}$, $\boldsymbol{V} := \boldsymbol{W}^{-1}$, then the new KKT inverse is

$$\begin{aligned} \boldsymbol{V}_{\text{new}} = \boldsymbol{V} + \sigma^{-1}\Big\{ & \alpha\left(\boldsymbol{e}_t - \boldsymbol{V}\boldsymbol{\omega}\right)\left(\boldsymbol{e}_t - \boldsymbol{V}\boldsymbol{\omega}\right)^\top - \beta \boldsymbol{V}\boldsymbol{e}_t\boldsymbol{e}_t^\top\boldsymbol{V} \\ & + \tau\left[\boldsymbol{V}\boldsymbol{e}_t\left(\boldsymbol{e}_t - \boldsymbol{V}\boldsymbol{\omega}\right)^\top + \left(\boldsymbol{e}_t - \boldsymbol{V}\boldsymbol{\omega}\right)\boldsymbol{e}_t^\top\boldsymbol{V}\right]\Big\}, \end{aligned} \tag{2-22}$$

where

$$\begin{cases} \alpha = \boldsymbol{e}_t^\top \boldsymbol{V}\boldsymbol{e}_t, \\ \beta = \frac{1}{8\eta_1}\left\|\boldsymbol{y}_{\text{new}} - \boldsymbol{x}_0\right\|_2^4 - \boldsymbol{\omega}^\top\boldsymbol{V}\boldsymbol{\omega}, \\ \tau = \boldsymbol{e}_t^\top\boldsymbol{V}\boldsymbol{\omega}, \\ \sigma = \alpha\beta + \tau^2. \end{cases} \tag{2-23}$$

We obtain the new KKT inverse $\boldsymbol{V}_{\text{new}}$ using the updating formula (2-22), and then compute

$$\begin{pmatrix} \boldsymbol{\lambda} \\ c \\ \boldsymbol{g} \end{pmatrix} = \boldsymbol{V}_{\text{new}} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ f(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}) \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

to obtain $(\boldsymbol{\lambda}, c, \boldsymbol{g})^{\top}$. In this case, the update of $\boldsymbol{V}$ can be computed within $\mathcal{O}((m+n)^2)$ operations.

After presenting the updating formula, we further consider additional details. We attempt to improve the robustness of the updating formula by adopting alternative ways of selecting new iteration points within the algorithm that uses the least H^2 norm updating quadratic model (via updating the KKT inverse using (2-22)).

Note that the denominator of the KKT inverse updating formula (2-22) is $\sigma = \alpha\beta + \tau^2$, where α, β, and τ are given in (2-23). To avoid numerical instability caused by the absolute value of σ being too small, we use a model improvement step as Step 4 in Algorithm 3, i.e., obtain the iteration point $\boldsymbol{y}_{\text{new}} = \boldsymbol{x}_{\text{opt}} + \boldsymbol{d}$, where $\boldsymbol{d} \in \Re^n$ is obtained by approximately solving

$$\begin{aligned} &\max_{\boldsymbol{d}} \ \left|\alpha\beta + \tau^2\right| \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \leq \Delta_k. \end{aligned} \tag{2-24}$$

(2-24) is a quartic problem in $\boldsymbol{d}$, since $\boldsymbol{y}_{\text{new}} = \boldsymbol{x}_{\text{opt}} + \boldsymbol{d}$. In fact, when the updating formula (2-22) may be unstable due to ill-conditioning of the interpolation set, the model improvement step in the algorithm based on the least H^2 norm updating quadratic model is chosen as the solution of the quartic problem above within the trust region. Note that the current implementation follows the main idea of the BIGDEN subroutine in Powell's NEWUOA (Section 6 of [94]) to obtain relatively large values of the objective function in (2-24) (we do not need to solve it very precisely). The current implementation attempts to maximize a quadratic approximation (second-order expansion) of the objective function in (2-24), then iteratively finds trial points, and stops once it finds a point where the value of (2-24) is 1.1 times the value at $\boldsymbol{d} = \boldsymbol{0}$. It is worth noting that since the subproblem itself is not derivative-free, other practical approaches may also be tried to solve such subproblems, so no further details are given here.

In the current implementation, if $\rho_k < \hat{\eta}_1$ and the distance between the farthest interpolation point $\boldsymbol{y}_{\text{far}}$ from $\boldsymbol{x}_{\text{opt}}$ satisfies $\|\boldsymbol{y}_{\text{far}} - \boldsymbol{x}_{\text{opt}}\|_2 > 2\Delta_k$, the algorithm rejects the model and calls the model improvement step, similar to the practice in Powell's NEWUOA (related to the well-poisedness of interpolation). In addition, when

$\|\boldsymbol{x}_k - \boldsymbol{x}_0\|_2 > 10\Delta_k$, the base point $\boldsymbol{x}_0$ is changed to the current $\boldsymbol{x}_{\text{opt}}$, i.e., the center of the next trust region. For more details, Powell [94] provides detailed discussions on this point. The updating formula (2-22) reduces the overall computational complexity during iterations. More details about the choice of $\boldsymbol{x}_0$ are introduced in the work of Zhang [154]. In addition, more details on the geometry and well-poisedness of interpolation sets can be found in the work of Conn, Scheinberg, and Vicente [175].

2.1.4 Numerical Results

To demonstrate the advantages of our quadratic model updated by the least H^2 norm, we present numerical results for solving the unconstrained derivative-free optimization problem (1-1). The numerical experiments consist of three parts: observations and comparisons of interpolation errors and updates, a simple simulation, and comparisons via Performance Profiles and Data Profiles obtained from solving a set of test problems. Based on the framework provided by Algorithm 3, we implemented a derivative-free trust-region algorithm in Python for numerical tests. In these tests, the quadratic model with the least H^2 norm is obtained via the updating formula (2-19), and we use the formula (2-22) to update the inverse of the KKT matrix. The model-improvement step in the algorithm is obtained by approximately solving the subproblem (2-24). To directly and fairly compare different model functions in this section, we keep the algorithmic framework identical and replace the corresponding formulas with those of other models one by one. In the numerical experiments here, the weights C_1, C_2, C_3 are all set to $\frac{1}{3}$. In addition, in our implementation, the radius r at step k is set to $\max\{10\Delta_k, \max_{\boldsymbol{y}\in\mathcal{X}_k} \|\boldsymbol{y} - \boldsymbol{x}_{\text{opt}}\|_2\}$ (this is the same setting as Zhang [154][3]). The numerical results indicate that obtaining the quadratic model via the least H^2 norm update rather than the least Frobenius norm update is advantageous.

We first make numerical observations on interpolation error and stability when the quadratic model is updated by minimizing the H^2 norm between two consecutive models. To illustrate the advantage of using the H^2 norm to obtain the model function, we use the following example to numerically compare interpolation based on the least H^2 norm update with that based on the least Frobenius norm update.

Example 2.1. We know that the subproblem corresponding to the updating formula (2-5) can be rewritten as

$$\begin{aligned} &\min_{D\in\mathcal{Q}} \ \|D\|^2_{H^2(\mathcal{B}_r(\boldsymbol{x}_0))} \\ &\text{s. t. } D(\boldsymbol{y}_{\text{new}}) = f(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}),\ \boldsymbol{y}_{\text{new}} \in \mathcal{X}_k, \\ &\qquad D(\boldsymbol{y}_i) = 0,\ \boldsymbol{y}_i \in \mathcal{X}_k \backslash \{\boldsymbol{y}_{\text{new}}\} \end{aligned} \tag{2-25}$$

[3] There are other ways to choose r. The present choice is simple and sufficient for the numerical experiments.

to obtain D_k, where $D_k = Q_k - Q_{k-1}$. Therefore, in this simple 2D example, we assume that, at the k-th iteration, the function $f - Q_{k-1}$ in problem (2-25) satisfies

$$f(\boldsymbol{x}) - Q_{k-1}(\boldsymbol{x}) = \begin{cases} 1, \text{ if } \boldsymbol{x} = \boldsymbol{y}_{\text{new}}, \\ 0, \text{ otherwise.} \end{cases} \tag{2-26}$$

Remark 2.4. This example is closely related to Lagrange basis functions. Note that, in the corresponding k-th iteration, the old point $\boldsymbol{y}_t$ is replaced by $\boldsymbol{y}_{\text{new}}$. The function $Q_k = Q_{k-1} + D_k$ is exactly the k-th model; the initial model is $Q_0(\boldsymbol{x}) = 0$. Before entering the iterations, we set $f((0,0)^\top) = 1$ and $f(\boldsymbol{x}) = 0$ for $\forall\ \boldsymbol{x} \neq (0,0)^\top$, and then in subsequent steps f satisfies (2-26). We use this example to observe the basic behavior of the models. Powell [176] discusses the advantages of obtaining models using Lagrange bases.

We use 3 interpolation points at each step. The initial interpolation points for this simple example are

$$\boldsymbol{y}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ \boldsymbol{y}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \boldsymbol{y}_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

which is a common choice. The total number of iterations is 3, and the trust region here is set to $\mathcal{B}_1(\boldsymbol{x}_{\text{opt}})$, where $\boldsymbol{x}_{\text{opt}}$ is the point with the smallest function value among the current interpolation points. Here, we focus on and compare the interpolation behavior of the least Frobenius norm update model and the least H^2 norm update model in the initial stage. We fix the trust-region radius so that the interpolation errors in the first three iterations provide a fair and intuitive comparison; note that, for comparison, we also consider computing interpolation errors on a grid in the region as a simple reference.

Figure 2-1 shows the numerical results. In each subfigure of Figure 2-1, we plot two curves representing Powell's model and our model, based on the least Frobenius norm update and the least H^2 norm update, respectively. Subfigures 2-1a and 2-1b in Figure 2-1 display the relationships between the number of iterations and the maximum interpolation error at all iterates as well as the average interpolation error at all iterates. Subfigures 2-1c and 2-1d in Figure 2-1 show the relationships between the number of iterations and the maximum interpolation error on grid points as well as the average interpolation error on grid points. Here, the interpolation error at iterates and the interpolation error on grid points are defined respectively by

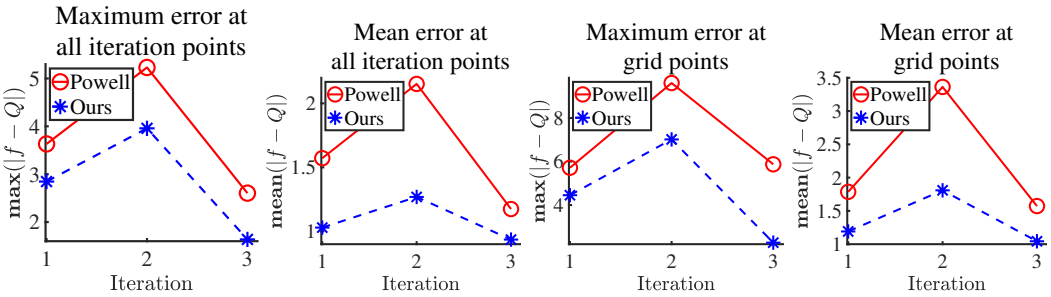
$$\text{Err}_{\text{itr}}(\boldsymbol{z}) = \left| f(\boldsymbol{z}) - Q_k(\boldsymbol{z}) \right|$$

and

$$\text{Err}_{\text{grid}}(\boldsymbol{z}_{pq}) = \left| f(\boldsymbol{z}_{pq}) - Q_k(\boldsymbol{z}_{pq}) \right|,$$

where $\boldsymbol{z}$ denotes a historical iterate, and $\boldsymbol{z}_{pq} = (\frac{p}{100}, \frac{q}{100})^\top$, $p, q \in [-100, 100] \cap \mathbb{Z}$.

The differences in interpolation error in Figure 2-1 illustrate the advantage of our quadratic model with the least H^2 norm update. During the iterations, our model yields smaller interpolation errors at the old discarded interpolation points and at grid points in the region $\{\boldsymbol{x} = (x_1, x_2)^\top : x_1, x_2 \in [-1, 1]\}$ than the model updated by the least Frobenius norm. In other words, in this example we observe that the least H^2 norm update is numerically more stable.



(a) Maximum error at historical iterates **(b) Average error at historical iterates** **(c) Maximum error on grid points** **(d) Average error on grid points**

Figure 2-1 Interpolation error comparison of different interpolation quadratic models

It is worth noting that, in our design here, the target function scales the function value in the interpolation constraint at $\boldsymbol{y}_{\text{new}}$ to 1, and the target function itself is discontinuous. Considering that the model function is continuous and that the above setup helps us make a clear observation under relatively fair and simple conditions, we do not require the interpolation error here to always be very small. Of course, numerically, smaller errors are preferable.

The following example further shows the advantage of the least H^2 norm update quadratic model when iteratively solving a simple and classic test problem.

Example 2.2. The objective function we test is the 2-dimensional Rosenbrock function [72]

$$f(\boldsymbol{x}) = f(x_1, x_2) = \left(1 - x_1\right)^2 + 100\left(x_2 - x_1^2\right)^2,$$

where x_1 and x_2 denote the first and second components of the variable $\boldsymbol{x}$. The initial interpolation points used in the experiment are the origin and three uniformly distributed points on the unit circle, namely,

$$\boldsymbol{y}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ \boldsymbol{y}_2 = \begin{pmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{pmatrix}, \ \boldsymbol{y}_3 = \begin{pmatrix} -\frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{pmatrix}, \ \boldsymbol{y}_4 = \begin{pmatrix} 0 \\ -1 \end{pmatrix},$$

which is a simple and common setting when using 4 interpolation points in $\mathfrak{R}^2$. The purpose of this example is to compare the two models from the perspective of minimizing a classical function. According to Powell [94], we choose the number of interpolation points to be $n + 2$, so that when minimizing an n-dimensional objective function,

at least one constraint is provided on the Hessian matrix of the least Frobenius norm update model; here $n = 2$.

We apply derivative-free trust-region methods based on the least Frobenius norm update quadratic model and on the least H^2 norm update quadratic model, respectively, to iteratively minimize the 2D Rosenbrock function. The initial interpolation points are as above, and the initial trust-region radius is 1. Moreover, the tolerances for the trust-region radius and the model gradient norm are 10^{-8}, and $\mu = 0.1$. The parameters for updating the trust-region radius are $\gamma = 2, \hat{\eta}_1 = \frac{1}{2}, \hat{\eta}_2 = \frac{1}{4}$. Figure 2-2 shows the iteration results, and details of the numbers of function evaluations, final function values, model gradient norms, and the best points are given in Table 2-1. NF denotes the number of function evaluations. It can be seen that, for this example, the algorithm using our model exhibits faster numerical convergence than the algorithm using the least Frobenius norm update model, which relies to some extent on the high approximation accuracy of our model. This experiment shows that, for minimizing the 2D Rosenbrock function, the least H^2 norm update quadratic model has advantages over the least Frobenius norm update quadratic model.

Table 2-1 Numbers of function evaluations, final function values, model gradient norms, and the best points for Example 2.2

Model	NF	Final f value	Model grad. norm	Best point
Powell	67	3.8630×10^{-9}	0.0015	$(1.00005607, 1.00011483)^{\top}$
Least H^2 norm update	49	7.8825×10^{-12}	7.8587×10^{-6}	$(1.00000271, 1.00000535)^{\top}$

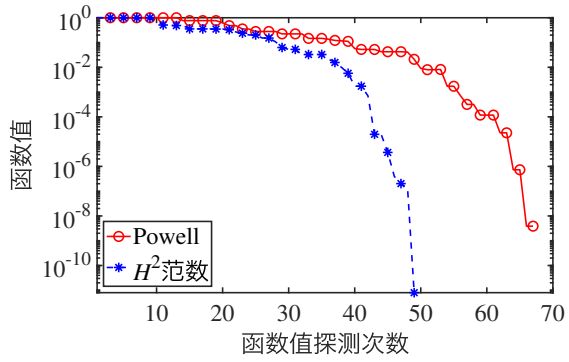


Figure 2-2 Convergence plot of minimizing 2-dimensional Rosenbrock function based on Powell's least Frobenius norm updating model and our least H^2 norm updating model

In addition, we conducted further numerical experiments for the algorithm based on our quadratic model updated by the least H^2 norm in order to investigate the effect of "using different numbers of interpolation points," i.e., we set m to be from 1 to $\frac{1}{2}(n+1)(n+2)$. Table 2-2 reports the numbers of function evaluations when minimizing the 2D Rosenbrock function using the least H^2 norm update quadratic model with different numbers of interpolation points. The other settings are the same as before.

Table 2-2 Minimizing Rosenbrock function with different number of interpolation points

Initial interpolation points						NF
$(0,0)^\top$	-	-	-	-	-	56
$(0,0)^\top$	$(1,0)^\top$	-	-	-	-	58
$(0,0)^\top$	$(1,0)^\top$	$(0,1)^\top$	-	-	-	60
$(0,0)^\top$	$(\frac{\sqrt{3}}{2},\frac{1}{2})^\top$	$(-\frac{\sqrt{3}}{2},\frac{1}{2})^\top$	$(0,-1)^\top$	-	-	49
$(0,0)^\top$	$(1,0)^\top$	$(0,1)^\top$	$(-1,0)^\top$	$(0,-1)^\top$	-	61
$(0,0)^\top$	$(1,0)^\top$	$(0,1)^\top$	$(-1,0)^\top$	$(0,-1)^\top$	$(\frac{\sqrt{2}}{2},-\frac{\sqrt{2}}{2})^\top$	63

The main considerations regarding the optimal number of interpolation points per iteration are as follows: fewer interpolation points entail lower computational cost and may, under the guarantee of the projection property in Theorem 2.4, provide a better update. This is because, when solving problem (2-5) to obtain Q_k, having fewer interpolation constraints can make $\|Q_k - Q_{k-1}\|_{H^2(\Omega)}$ smaller. Note that the comparison here is made in the sense of different numbers of interpolation conditions. In addition, the number of interpolation points can also be chosen dynamically according to the accuracy required during the optimization process.

There is still room for improvement in our method. For example, changing the relevant parameters in the algorithm (such as the coefficients C_1, C_2, C_3 and the number of interpolation points) may lead to different outcomes. Below we present an example in which the least Frobenius norm update quadratic model performs better than the least H^2 norm update quadratic model, indicating that the least Frobenius norm update model can be numerically preferable in some situations.

Example 2.3. In this example, we attempt to minimize the test function "DQRTIC" [177], whose expression is

$$f(\boldsymbol{x}) = f(x_1, x_2) = \left(x_1 - 1\right)^4 + \left(x_2 - 2\right)^4.$$

The global minimum of this example is 0. We fix the trust-region radius at 1. The initial interpolation points $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{y}_3, \boldsymbol{y}_4, \boldsymbol{y}_5$ (we use 5 interpolation points at each iteration) are

$$\boldsymbol{y}_1 = \begin{pmatrix} -20 \\ 1 \end{pmatrix}, \ \boldsymbol{y}_2 = \begin{pmatrix} -19 \\ 1 \end{pmatrix}, \ \boldsymbol{y}_3 = \begin{pmatrix} -20 \\ 2 \end{pmatrix}, \ \boldsymbol{y}_4 = \begin{pmatrix} -21 \\ 1 \end{pmatrix}, \ \boldsymbol{y}_5 = \begin{pmatrix} -20 \\ 0 \end{pmatrix}.$$

The best function values over the first 16 function evaluations are shown in Figure 2-3. In this example, Powell's model performs better than ours, especially between the 10-th and 16-th evaluations, achieving function values 2.08×10^4 and 8.38×10^4 respectively at the 16-th evaluation.

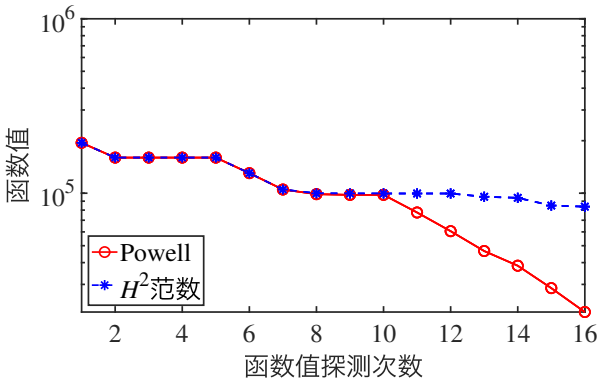


Figure 2-3 Minimizing 2-dimensional DQRTIC function based on Powell's least Frobenius norm updating model and our least H^2 norm updating model

To further examine the numerical performance of our algorithm based on the quadratic model updated by the least H^2 norm, we solve some classic test problems and present the results using Performance Profiles and Data Profiles. The test problems corresponding to the Performance Profiles in Figure 2-4 and the Data Profiles in Figure 2-5 are listed in Table 2-3. They are all selected from classic and commonly used unconstrained optimization test function sets, and the objective functions in the tested optimization problems are smooth. For the Performance and Data Profiles, the stopping criterion regarding the number of function evaluations for all tested algorithms is set to at most $100n$ evaluations, where n is the dimension of the corresponding problem. Here, $\boldsymbol{x}_{\text{int}}$ denotes the starting point, and $\boldsymbol{x}^*$ denotes the best known point (obtained in the numerical comparisons).

Table 2-3 50 test problems for Figure 2-4 and Figure 2-5

Problem	Dimension	$f(\boldsymbol{x}_{\text{int}})$	$f(\boldsymbol{x}^*)$
ARGLINA [177, 178]	8	4.00×10^{1}	8.00
ARGLINB [177, 178]	10	8.66×10^{6}	4.63
ARGTRIG [177]	8	8.45×10^{-3}	5.01×10^{-14}
BDQRTIC [177]	100	2.17×10^{4}	3.79×10^{2}
BDVALUE [177, 178]	100	1.23×10^{-6}	9.33×10^{-7}
BRYBND [177, 178]	180	6.48×10^{3}	1.44×10^{-9}
CHAINWOO [177, 179]	140	5.16×10^{5}	2.98×10^{2}
CHEBQUAD [177, 178]	120	1.75×10^{-2}	6.56×10^{-3}
CHNROSNB [177, 180]	80	1.61×10^{3}	3.52×10^{-11}
CHPOWELLS [178, 181]	20	1.10×10^{3}	5.52×10^{-10}
COSINE [177]	90	7.81×10^{1}	-8.90×10^{1}
CUBE [177]	50	3.02×10^{4}	3.51×10^{-3}
CURLY10 [177]	10	-5.06×10^{-5}	-1.00×10^{3}

Table 2-3 (continued)

CURLY20 [177]	20	-1.01×10^{-4}	-2.01×10^{3}
CURLY30 [177]	30	-1.52×10^{-4}	-3.01×10^{3}
DIXMAANE [177]	60	4.45×10^{2}	1.00
DIXMAANF [177]	65	4.70×10^{2}	1.00
DIXMAANG [177]	70	8.81×10^{2}	1.00
DIXMAANH [177]	75	1.82×10^{3}	1.00
DIXMAANI [177]	80	5.26×10^{2}	1.00
DIXMAANJ [177]	85	5.65×10^{2}	1.00
DIXMAANK [177]	90	1.08×10^{3}	1.00
DIXMAANL [177]	95	2.19×10^{3}	1.00
DIXMAANM [177]	100	3.14×10^{2}	1.00
DIXMAANN [177]	105	3.33×10^{2}	1.00
DIXMAANO [177]	120	5.96×10^{2}	1.00
DIXMAANP [177]	130	1.14×10^{3}	1.00
DQRTIC [177]	110	3.01×10^{9}	4.76×10^{-6}
ERRINROS [177]	170	5.39×10^{5}	1.34×10^{2}
EXPSUM [182]	175	1.80×10^{6}	8.03×10^{3}
EXTROSNB [177, 180]	180	7.16×10^{4}	8.33×10^{-4}
FLETCHCR [177]	165	1.64×10^{4}	4.05×10^{-2}
FREUROTH [177, 178]	100	9.96×10^{4}	1.08×10^{4}
GENROSE [177]	130	5.14×10^{2}	1.18×10^{2}
INTEGREQ [177, 178]	110	6.30×10^{-1}	3.84×10^{-12}
MOREBV [177, 178]	8	1.37×10^{-3}	6.50×10^{-14}
NCB20 [177]	175	5.47×10^{3}	2.81×10^{2}
NONDQUAR [177]	160	1.66×10^{2}	2.88×10^{-4}
POWELLSG [177, 178]	180	9.68×10^{3}	1.89×10^{-3}
POWER [177]	135	8.29×10^{5}	3.78×10^{-20}
ROSENBROCK [177, 178]	10	3.64×10^{3}	4.69×10^{-7}
SBRYBND [177, 178]	50	7.68×10^{2}	1.98×10^{1}
SCOSINE [177]	180	1.03×10^{1}	-5.45×10^{1}
SPARSINE [177]	160	5.33×10^{4}	1.47×10^{-5}
SPMSRTLS [177]	180	1.30×10^{2}	9.84×10^{-11}
SROSENBR [177, 178]	8	9.68×10^{1}	4.58×10^{-2}
TOINTGSS [177]	100	8.92×10^{2}	9.71
TQUARTIC [177]	20	8.10×10^{-1}	1.12×10^{-12}
WOODS [177, 178]	24	1.15×10^{5}	2.45×10^{1}
VARDIM [177, 178]	180	1.41×10^{16}	4.15

Based on the numerical results of the above interpolation error and simple examples, the least H^2 norm updating quadratic model function performs better for Example 2.1 and Example 2.2. The following numerical results will show that, for the tested problem set, the algorithm using our model converges faster and performs better numerically than the algorithm using the least Frobenius norm updating quadratic model.

For each problem in this experiment, all algorithms start from the same input point $\boldsymbol{x}_{\text{int}}$, and the accuracy τ is set to 10^{-1}, 10^{-3}, and 10^{-5}. The algorithm framework is shown in Algorithm 3, where "Powell" denotes the least Frobenius norm updating quadratic model of Powell. Here, for the algorithm using Powell's least Frobenius norm updating quadratic model, the number of interpolation points per iteration is $m = 2n+1$. "H^2 $(m = 2n+1)$" and "H^2 $(m = \lceil \frac{n}{2} \rceil + 1)$" both use the least H^2 norm updating quadratic model and share the same framework as "Powell". For the three algorithms in Figure 2-4, the tolerances of the trust-region radius and gradient norm are set to 10^{-8}. They share the same initial trust-region radius. In Algorithm 3, the parameters are $\gamma = 2, \hat{\eta}_1 = \frac{1}{2}, \hat{\eta}_2 = \frac{1}{4}, \mu = 0.1$. For a fair comparison with other quadratic models, the methods "Powell" and "H^2 $(m = 2n+1)$" use $2n+1$ interpolation points per iteration and share the same initial interpolation points $\boldsymbol{x}_{\text{int}}, \boldsymbol{x}_{\text{int}} \pm \boldsymbol{e}_i, i = 1, \cdots, n$. In addition, the method "H^2 $(m = \lceil \frac{n}{2} \rceil + 1)$" uses $\lceil \frac{n}{2} \rceil + 1$ interpolation points per iteration, with initial interpolation points $\boldsymbol{x}_{\text{int}}, \boldsymbol{x}_{\text{int}} + \boldsymbol{e}_j, j = 1, \cdots, \lceil \frac{n}{2} \rceil + 1$.

In fact, choosing different m for different problems yields different numerical performance. Considering that the least H^2 norm updating quadratic model has already reduced the lower bound of the number of interpolation points per step, this is worth further study in the future. The performance of "H^2 $(m = \lceil \frac{n}{2} \rceil + 1)$" can demonstrate the numerical advantage of our method and model when using fewer interpolation points per iteration.

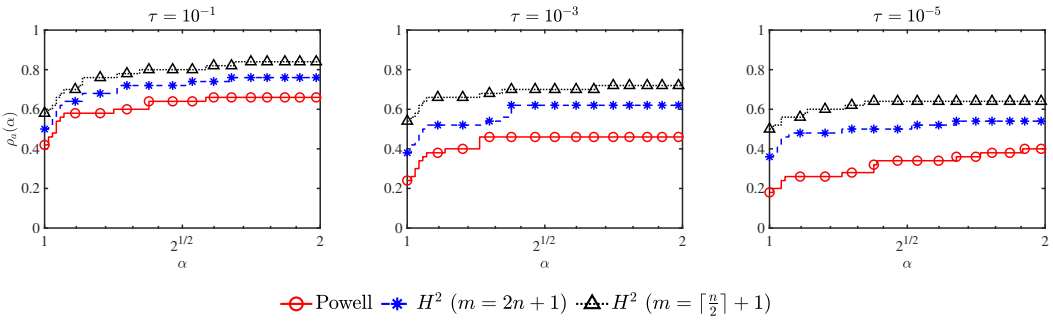


Figure 2-4 Performance Profile of solving test problems with derivative-free trust-region algorithms based on different quadratic models

In Figure 2-4, the results of $\rho_a(1)$ show that the algorithm based on the least H^2 norm updating quadratic model with $m = \lceil \frac{n}{2} \rceil + 1$ can solve the most problems with the fewest function evaluations among the three cases shown. The two algorithms using the

least H^2 norm updating quadratic model perform better than the algorithm using the least Frobenius norm updating quadratic model on this test set.

To further numerically observe the overall performance of our algorithm based on the least H^2 norm updating quadratic model, we present below the numerical comparison results between our model-based algorithm and other algorithms using Data Profile. The derivative-free trust-region algorithms tested with Powell's model (least Frobenius norm updating quadratic model) and the least Frobenius norm quadratic model [160] are respectively the Python interface of NEWUOA in PDFO [142] [94] and a Python implementation of the DFO algorithm[4]. In addition, the Nelder-Mead simplex algorithm and the BFGS method using first-order finite-difference derivative approximations are obtained from the scipy.optimize library[5].

For each problem in the experiment, all algorithms start from the input point $\boldsymbol{x}_{\text{int}}$ of the problem, and the accuracy τ is set to 10^{-1}, 10^{-3}, and 10^{-5}. We keep the settings of the methods "H^2 $(m = 2n + 1)$" and "H^2 $(m = \lceil \frac{n}{2} \rceil + 1)$" the same as those in Section 2.1.4. "H^1 $(m = 2n + 1)$" shares the same algorithm framework and settings with our model, but it uses the least H^1 seminorm quadratic model, namely the combination of the classical $\left\|\nabla^2 Q\right\|_F^2$ and $\|\nabla Q\|_2^2$. For the trust-region algorithms "NEWUOA" and "DFO-py" in Figure 2-5, the tolerances of the trust-region radius and gradient norm are set to 10^{-8}. They share the same initial trust-region radius. In addition, in our numerical experiments, the trust-region methods "NEWUOA" and "DFO-py" use $2n+1$ interpolation points per iteration, and share the same initial interpolation points with "H^2 $(m = 2n + 1)$". For "Nelder-Mead-py", the initial simplex is an $n + 1$ dimensional simplex with vertices $\boldsymbol{x}_{\text{int}}, \boldsymbol{x}_{\text{int}} + \boldsymbol{e}_i, i = 1, \cdots, n$, and the absolute error tolerance of function values between iterations is set to 10^{-8}. For "BFGS", the relative step size for numerical gradient approximation is automatically selected by setting it to "None", and the corresponding gradient norm must be less than 10^{-8} before successful termination.

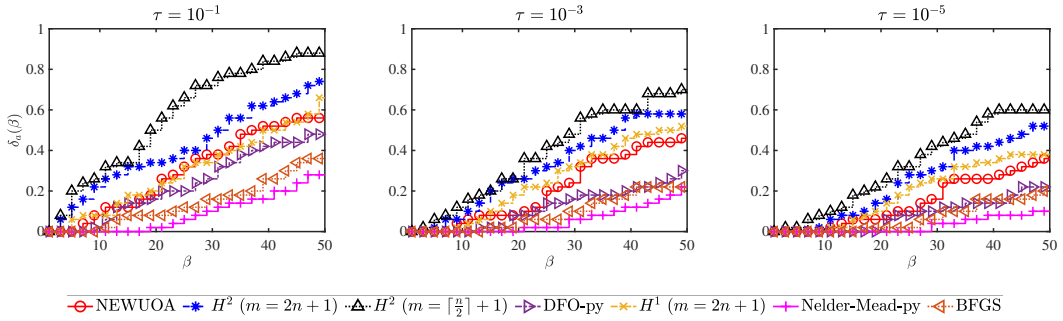


Figure 2-5 Data Profile of solving test problems with different algorithms

We can observe from Figure 2-5 that the trust-region algorithms based on the least H^2 norm updating quadratic model functions perform better than the other algorithms.

[4] https://coral.ise.lehigh.edu/katyas/software

[5] https://docs.scipy.org/doc/scipy

More specifically, when β exceeds about 40, in the case of $\tau = 10^{-1}$, they can solve more than 60% of the problems. The method using the model with $m = \lceil \frac{n}{2} \rceil + 1$ interpolation conditions per iteration performs better than the other methods. The above results demonstrate the advantage of the algorithm using our model.

2.1.5 Summary

For derivative-free trust-region optimization methods, using underdetermined quadratic interpolation models can reduce the number of interpolation points and function evaluations to some extent. To obtain a unique quadratic model function, a common approach is to determine the coefficients of the quadratic function at each iteration by solving an optimization problem with interpolation conditions as constraints. This section attempts to obtain the quadratic model function by minimizing the H^2 norm of the difference between the new and old quadratic model functions during the iteration. We find that this can reasonably reduce the lower bound of the number of interpolation points or equations. We presented the projection property and error bounds, and derived the corresponding updating formulas for computing the model function coefficients based on the KKT conditions of the corresponding optimization problem. Based on solving the KKT system (2-19) and formula (2-22), we obtained the updating formula for the inverse KKT matrix, providing more choices for underdetermined least norm updating quadratic models. Numerical results from different perspectives indicate the good performance of our model-based algorithm.

Regarding future work, we can further study and develop more convergence properties of derivative-free trust-region algorithms based on the least H^2 norm updating quadratic model. We can also design adaptive weight coefficients for problems with different structures, and obtain the quadratic model function at the k-th iteration by minimizing

$$C_1^{(k)} \left\| Q - Q_{k-1} \right\|_{H^0(\Omega)}^2 + C_2^{(k)} \left| Q - Q_{k-1} \right|_{H^1(\Omega)}^2 + C_3^{(k)} \left| Q - Q_{k-1} \right|_{H^2(\Omega)}^2$$

where k corresponds to the k-th iteration. Another potential future work is to find better choices for the number of interpolation points used in constructing the model at each iteration, since we have now reduced the lower bound of this number. As shown in Section 2.1.4 (especially Example 2.3), our model still has limitations, so finding deeper relationships between performance and coefficients or the number of interpolation points will be valuable. Other different types of derivative-free optimization interpolation models can also be further studied, including underdetermined model functions suitable for large-scale sparse derivative-free optimization problems, as well as models suitable for solving optimization problems with nonlinear or linear constraints.

2.2 Least Weighted H^2 Norm Updating Quadratic Interpolation Model

This section further discusses the least weighted H^2 norm updating quadratic interpolation model based on the previous section, and applies it to model-based derivative-free trust-region methods. The least weighted H^2 norm updating quadratic interpolation model here refers to the solution obtained by solving subproblem (2-6). It is worth noting that in model-based derivative-free trust-region methods, as the number of iterations increases, the trust-region radius will converge to 0. This section will focus on this situation. At the same time, this situation also applies to numerical computation in high-precision environments.

Specifically, this section focuses on the least weighted H^2 norm updating quadratic model and its corresponding KKT matrix, defines new distance and error measures called KKT matrix distance and KKT matrix error, respectively; gives the definition of the barycenter of the coefficient region, and provides the analytical barycenter of the weight coefficient region of the least weighted H^2 norm updating quadratic model; and finally provides numerical support for the best choice of weight coefficient region.

The remainder of this section is organized as follows. Section 2.2.1 gives more details of the least weighted H^2 norm updating quadratic model and the KKT matrix, showing the corresponding results when the radius r used in computing the H^2 norm converges to 0. In addition, Section 2.2.2 proposes new distance and error measures, called KKT matrix distance and KKT matrix error, to illustrate the differences in the KKT matrix caused by different weight coefficients. This section also gives the definition of the barycenter of the coefficient region. Section 2.2.3 gives the barycenter of the weight coefficient region of the least weighted H^2 norm updating quadratic model when the trust-region radius is small. Section 2.2.4 presents the numerical performance comparison results for different weight coefficients.

2.2.1 Least Weighted H^2 Norm Updating Quadratic Model and KKT Matrix

According to the previous discussion, the coefficient r is usually proportional to the trust-region radius and the maximum distance between the interpolation points and the current trust-region center in the iteration, where the latter is proportional to the trust-region radius. Therefore, $r \to 0$ corresponds to the case of a small trust-region radius.

Note that we rewrite $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ in (2-14) as function forms:

$$\eta_1(C_1, C_2, C_3, n, r),\ \eta_2(C_1, C_2, n, r),\ \eta_3(C_1, n, r),\ \eta_4(C_1, n, r),\ \eta_5(C_1).$$

Some results of the parameters of the KKT matrix $\boldsymbol{W}$ in the limit case of a small trust-region radius are given in the following proposition.

Proposition 2.12. *$\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ satisfy*

$$\begin{cases} \lim\limits_{r\to 0} \eta_1(C_1, C_2, C_3, n, r) = C_3, \\ \lim\limits_{r\to 0} \eta_2(C_1, C_2, n, r) = C_2, \\ \lim\limits_{r\to 0} \eta_3(C_1, n, r) = 0, \\ \lim\limits_{r\to 0} \eta_4(C_1, n, r) = 0, \\ \lim\limits_{r\to 0} \eta_5(C_1) = C_1, \end{cases} \tag{2-27}$$

then

$$\begin{cases} \lim\limits_{r\to 0} \dfrac{1}{8\eta_1} = \dfrac{1}{8C_3}, \\ \lim\limits_{r\to 0} -\dfrac{\eta_3}{8\eta_1\left(n\eta_3 + \eta_1\right)} = 0, \\ \lim\limits_{r\to 0} -\dfrac{\eta_4}{4n\eta_3 + 4\eta_1} = 0, \\ \lim\limits_{r\to 0} -2\eta_2 = -2C_2, \\ \lim\limits_{r\to 0} \dfrac{n\eta_4^2}{2n\eta_3 + 2\eta_1} - 2\eta_5 = -2C_1. \end{cases}$$

Proof. The result can be obtained by direct calculation. □

2.2.2 KKT Matrix Error and the Barycenter of the Coefficient Region

We know that computing the corresponding parameters $\boldsymbol{\lambda}, c, \boldsymbol{g}$ based on given C_1 and C_2 yields the least weighted H^2 norm updating quadratic model. Since $\boldsymbol{\lambda}, c, \boldsymbol{g}$ depend directly on the KKT matrix $\boldsymbol{W}$ in the KKT system (2-19), we attempt to characterize the distance between two models using a KKT matrix distance; see Definition 2.16. Specifically, as shown in the KKT system (2-19), the vector $(0, \cdots, 0, f(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}), 0, \cdots, 0)^\top$ remains unchanged for different weight coefficients, while the only difference lies in the KKT matrix $\boldsymbol{W}$ on the left-hand side of (2-19). This KKT matrix $\boldsymbol{W}$ directly determines the parameters of our quadratic model and thereby the least weighted H^2 norm updating quadratic model associated with it. Therefore, the KKT matrix can sufficiently represent the differences among models obtained by minimizing the weighted H^2 norm with different weight coefficients. In other words, the KKT matrix is, in a certain sense, key to measuring the quality of the corresponding quadratic model.

This subsection seeks a "balance point" for the above weight coefficients. Our aim is to identify a central KKT matrix by giving the barycenter of the weight-coefficient region, in order to find optimal coefficients in this sense.

Remark 2.5. Without loss of generality, we assume $C_1 + C_2 + C_3 = 1$. This does not affect our discussion of the weight coefficients.

The analysis below is conducted under the following assumption.

Assumption 2.13. $\boldsymbol{W}$ and $\boldsymbol{W}^*$ are the KKT matrices corresponding to the model functions determined by (2-19) with weight coefficients C_1, C_2 and C_1^*, C_2^*, respectively.

We have the following theorem.

Theorem 2.14. *Assume that $\boldsymbol{W}$ and $\boldsymbol{W}^*$ satisfy Assumption 2.13. Then $\left\|\boldsymbol{W}-\boldsymbol{W}^*\right\|_F^2$ equals*

$$\begin{aligned}
\left\|\boldsymbol{W}-\boldsymbol{W}^*\right\|_F^2 &= \left\{\sum_{i=1}^{m}\sum_{j=1}^{m}\left[\left(\boldsymbol{y}_i-\boldsymbol{x}_0\right)^{\top}\left(\boldsymbol{y}_j-\boldsymbol{x}_0\right)\right]^4\right\}\left(\frac{1}{8\eta_1}-\frac{1}{8\eta_1^*}\right)^2 \\
&+\left(\sum_{i=1}^{m}\sum_{j=1}^{m}\left\|\boldsymbol{y}_i-\boldsymbol{x}_0\right\|_2^4\left\|\boldsymbol{y}_j-\boldsymbol{x}_0\right\|_2^4\right)\left(\frac{\eta_3}{8\eta_1\left(n\eta_3+\eta_1\right)}-\frac{\eta_3^*}{8\eta_1^*\left(n\eta_3^*+\eta_1^*\right)}\right)^2 \\
&+\left(\sum_{i=1}^{m}\left\|\boldsymbol{y}_i-\boldsymbol{x}_0\right\|_2^4\right)\left[-\frac{\eta_4}{4n\eta_3+4\eta_1}-\left(-\frac{\eta_4^*}{4n\eta_3^*+4\eta_1^*}\right)\right]^2 \\
&+n\left[2\left(\eta_2^*-\eta_2\right)\right]^2+\left[\frac{n\eta_4^2}{2n\eta_3+2\eta_1}-2\eta_5-\left(\frac{n(\eta_4^*)^2}{2n\eta_3^*+2\eta_1^*}-2\eta_5^*\right)\right]^2,
\end{aligned}$$

where $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ are defined by (2-14), and $\eta_1^*, \eta_2^*, \eta_3^*, \eta_4^*, \eta_5^*$ are

$$\begin{cases}
\eta_1^* = C_1^*\dfrac{r^4}{2(n+4)(n+2)}+C_2^*\dfrac{r^2}{n+2}+C_3^*, \\
\eta_2^* = C_1^*\dfrac{r^2}{n+2}+C_2^*, \\
\eta_3^* = C_1^*\dfrac{r^4}{4(n+4)(n+2)}, \\
\eta_4^* = C_1^*\dfrac{r^2}{n+2}, \\
\eta_5^* = C_1^*.
\end{cases}$$

Proof. Computing $\boldsymbol{A}-\boldsymbol{A}^*$, $\boldsymbol{J}-\boldsymbol{J}^*$, and $\boldsymbol{X}-\boldsymbol{X}^*$, and then summing the squares yields the result. □

Moreover, we can obtain the following corollary for $\|\boldsymbol{W}-\boldsymbol{W}^*\|_F^2$.

Corollary 2.15. *Assume that $\boldsymbol{W}$ and $\boldsymbol{W}^*$ satisfy Assumption 2.13. Then $\left\|\boldsymbol{W}-\boldsymbol{W}^*\right\|_F^2$ is a function of $C_1, C_2, C_1^*, C_2^*, n, r$, and we have*

$$\begin{aligned}
&\left\|\boldsymbol{W}-\boldsymbol{W}^*\right\|_F^2 := \mathcal{D}(C_1, C_2, C_1^*, C_2^*, n, r) \\
&= \mathcal{R}_1(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m)\mathcal{P}_1(C_1, C_2, C_1^*, C_2^*, n, r)+\mathcal{R}_2(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m)\mathcal{P}_2(C_1, C_2, C_1^*, C_2^*, n, r) \\
&+\mathcal{R}_3(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m)\mathcal{P}_3(C_1, C_2, C_1^*, C_2^*, n, r)+\mathcal{P}_4(C_1, C_2, C_1^*, C_2^*, n, r),
\end{aligned} \tag{2-28}$$

where

$$\begin{cases} \mathcal{R}_1(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left[\left(\boldsymbol{y}_i - \boldsymbol{x}_0 \right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0 \right) \right]^4, \\ \mathcal{R}_2(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left\| \boldsymbol{y}_i - \boldsymbol{x}_0 \right\|_2^4 \left\| \boldsymbol{y}_j - \boldsymbol{x}_0 \right\|_2^4, \\ \mathcal{R}_3(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m) = \sum_{i=1}^{m} \left\| \boldsymbol{y}_i - \boldsymbol{x}_0 \right\|_2^4 \end{cases}$$

depend only on the interpolation points $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m$ given the current base point $\boldsymbol{x}_0$ of the iteration, and $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4$ are functions of $C_1, C_2, C_1^*, C_2^*, n, r$, whose explicit expressions will be given later.

Proof. Substituting the expressions of $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$ and $\eta_1^*, \eta_2^*, \eta_3^*, \eta_4^*, \eta_5^*$ yields (2-28). □

To further discuss the central KKT matrix, we first introduce the definitions of the KKT matrix distance and KKT matrix error.

Definition 2.16 (KKT matrix distance)**.** *We define the KKT matrix distance between two KKT matrices $\boldsymbol{W}$ and $\boldsymbol{W}^*$ as $\|\boldsymbol{W} - \boldsymbol{W}^*\|_F$.*

Theorem 2.17. *The KKT matrix distance in Definition 2.16 is a well-defined distance on the set of KKT matrices.*

Proof. We have the following facts.

- The KKT matrix distance is nonnegative, and $\left\|\boldsymbol{W} - \boldsymbol{W}^*\right\|_F = 0$ if and only if $\boldsymbol{W} = \boldsymbol{W}^*$.
- Symmetry holds: $\left\|\boldsymbol{W} - \boldsymbol{W}^*\right\|_F = \|\boldsymbol{W}^* - \boldsymbol{W}\|_F$.
- The triangle inequality

$$\left\|\boldsymbol{W} - \bar{\boldsymbol{W}}^*\right\|_F = \left\|\left(\boldsymbol{W} - \boldsymbol{W}^*\right) + \left(\boldsymbol{W}^* - \bar{\boldsymbol{W}}^*\right)\right\|_F \leq \left\|\boldsymbol{W} - \boldsymbol{W}^*\right\|_F + \left\|\boldsymbol{W}^* - \bar{\boldsymbol{W}}^*\right\|_F.$$

Therefore, the KKT matrix distance is a well-defined distance. □

Definition 2.18 (KKT matrix error)**.** *Assume that $\boldsymbol{W}$ and $\boldsymbol{W}^*$ satisfy Assumption 2.13. We define the KKT matrix error between two sets of weight coefficients (C_1, C_2) and (C_1^*, C_2^*) as $\sqrt{\mathcal{D}(C_1, C_2, C_1^*, C_2^*, n, r)}$, where $\mathcal{D}(C_1, C_2, C_1^*, C_2^*, n, r)$ is defined in (2-28).*

We will use the KKT matrix error to help find appropriate weight coefficients C_1, C_2 and $C_3 = 1 - C_1 - C_2$, where $(C_1, C_2)^\top$ lies in the region $\mathcal{C}$. Figure 2-6 shows the coefficient region $\mathcal{C}$. To avoid a too small denominator η_1 in the KKT matrix of (2-19) when $r \to 0$, we assume in our analysis that C_3 has a lower bound ε with $0 < \varepsilon < 1$.

To further introduce the average KKT matrix distance, we give the following definition.

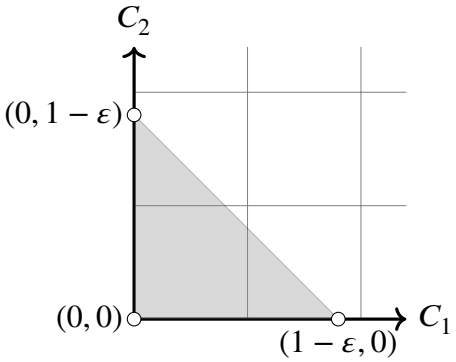


Figure 2-6 Coefficient region C

Definition 2.19 (Average squared KKT matrix error)**.** *Consider (2-28). Given a pair of coefficients C_1 and C_2, we define the average squared KKT matrix error as*

$$\begin{aligned}\mathrm{Error}_{\mathrm{ave}}(C_1, C_2, n, r, \varepsilon) &:= \frac{\int_0^{1-\varepsilon-C_1^*}\int_0^{1-\varepsilon} \mathcal{D}(C_1, C_2, C_1^*, C_2^*, n, r) dC_1^* dC_2^*}{\int_0^{1-\varepsilon-C_1}\int_0^{1-\varepsilon} 1 dC_1^* dC_2^*} \\ &= \frac{2\int_0^{1-\varepsilon-C_1^*}\int_0^{1-\varepsilon} \mathcal{D}(C_1, C_2, C_1^*, C_2^*, n, r) dC_1^* dC_2^*}{(1-\varepsilon)^2}.\end{aligned}$$

Definition 2.20 (Barycenter of the coefficient region)**.** *The barycenter of the weight-coefficient region C for the least weighted H^2 norm updating quadratic model is the solution of*

$$\min_{(C_1, C_2)^\top \in C} \mathrm{Error}_{\mathrm{ave}}(C_1, C_2, n, r, \varepsilon). \tag{2-29}$$

The barycenter of the weight-coefficient region C has the smallest average squared KKT matrix error. Note that the measure here is the KKT matrix error (rather than Euclidean distance).

2.2.3 Barycenter of the Weight Coefficient Region of Least Weighted H^2 Norm Updating Quadratic Models

This subsection provides an analytic result for the barycenter of the weight-coefficient region of the least weighted H^2 norm updating quadratic model when the trust-region radius is small.

Theorem 2.21. *Given the lower bound ε of C_3, if $r \to 0$, then*

$$\lim_{r\to 0} \mathrm{Error}_{\mathrm{ave}}(C_1, C_2, n, r, \varepsilon) = \mathcal{R}_1(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m)\mathrm{Error}^{(1)}_{\mathrm{ave}}(C_1, C_2, \varepsilon) + \mathrm{Error}^{(2)}_{\mathrm{ave}}(C_1, C_2, n, \varepsilon),$$

where $\mathrm{Error}^{(1)}_{\mathrm{ave}}(C_1, C_2, \varepsilon)$ *is*

$$\frac{\frac{(\varepsilon-1)\left(\varepsilon(4C_1+4C_2-5)-2(C_1+C_2-1)^2+\varepsilon^2\right)}{2\varepsilon} + (C_1+C_2-3)(C_1+C_2-1)\ln(\varepsilon)}{32(1-\varepsilon)^2(C_1+C_2-1)^2},$$

and $\mathrm{Error}_{\mathrm{ave}}^{(2)}(C_1, C_2, n, \varepsilon)$ *is*

$$\frac{1}{6}\left(24{C_1}^2 + 16C_1(\varepsilon - 1) + 6{C_2}^2 n + \varepsilon(4C_2 n - 2n - 8) - 4C_2 n + \varepsilon^2(n+4) + n + 4\right).$$

Proof. The result follows by directly computing the integrals and the limit. □

Next, we obtain the following result.

Theorem 2.22. *If* $\mathcal{R}_1(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m) \to 0$*, then* $C_1 = \frac{1-\varepsilon}{3}, C_2 = \frac{1-\varepsilon}{3}$ *corresponds to the weight coefficients that provide the central KKT matrix.*

Proof. We have

$$\lim_{\mathcal{R}_1(\boldsymbol{y}_1,\cdots,\boldsymbol{y}_m)\to 0} \mathrm{Error}_{\mathrm{ave}}(C_1, C_2, n, r, \varepsilon) = \mathrm{Error}_{\mathrm{ave}}^{(2)}(C_1, C_2, n, r, \varepsilon) \tag{2-30}$$

and

$$\left(\frac{1-\varepsilon}{3}, \frac{1-\varepsilon}{3}\right)^{\top} = \arg \min_{(C_1,C_2)^{\top}\in C} \mathrm{Error}_{\mathrm{ave}}^{(2)}(C_1, C_2, n, \varepsilon). \tag{2-31}$$

Therefore, the conclusion holds. □

We now give a numerical example.

Example 2.4. For $\varepsilon = 0.01$ and $n = 100$, we list the values of $\mathrm{Error}_{\mathrm{ave}}^{(1)}$ and $\mathrm{Error}_{\mathrm{ave}}^{(2)}$ corresponding to sampled weight coefficients when the trust-region radius is small.

Table 2-4 Values of $\mathrm{Error}_{\mathrm{ave}}^{(1)}$ and $\mathrm{Error}_{\mathrm{ave}}^{(2)}$ for sampled weight coefficients, $\varepsilon = 0.01$, $n = 100$

$(C_1, C_2)^{\top}$	$(\frac{1-\varepsilon}{3}, \frac{1-\varepsilon}{3})^{\top}$	$(\frac{1}{2}-\varepsilon, \frac{1}{2})^{\top}$	$(0, \frac{1}{2})^{\top}$	$(1-\varepsilon, 0)^{\top}$	$(0, 1-\varepsilon)^{\top}$	$(0,0)^{\top}$
$\mathrm{Error}_{\mathrm{ave}}^{(1)}$	5.663	8.655	8.988	18.3	49.66	16.99
$\mathrm{Error}_{\mathrm{ave}}^{(2)}$	2.467	136.2	2.611	136.2	136.2	2.795

Table 2-4 numerically indicates that $(\frac{1-\varepsilon}{3}, \frac{1-\varepsilon}{3})^{\top}$ achieves the smallest $\mathrm{Error}_{\mathrm{ave}}$ among the six pairs of weight coefficients.

Note that, ideally, the lower bound ε of C_3 tends to 0. As a limiting result of Theorem 2.22, the optimal weight coefficients are $C_1 = \frac{1}{3}, C_2 = \frac{1}{3}, C_3 = \frac{1}{3}$.

2.2.4 Numerical Results

We present the following numerical example by using the different models described above within Algorithm 3.

Example 2.5. In this numerical example, we use a derivative-free algorithm based on the least weighted H^2 norm updating quadratic model (constructed with different weight coefficients) to minimize the 2D Rosenbrock function

$$f(\boldsymbol{x}) = \left(1 - x_1\right)^2 + 100\left(x_2 - x_1^2\right)^2, \tag{2-32}$$

which is a smooth nonconvex function (as mentioned earlier), and $\boldsymbol{x} = (x_1, x_2)^\top$. The global minimizer of the test problem is 0, attained at $(1, 1)^\top$.

In the experiment, the initial input point is set as $\boldsymbol{x}_{\text{int}} = (1.04, 1.1)^\top$. In each iteration, we use 5 points for interpolation. The maximum number of function evaluations is uniformly set to 16, and the initial trust-region radius is $\Delta_0 = 10^{-4}$. In addition, the tolerances for the trust-region radius, the function value, and the gradient norm are all set to 10^{-8}. The parameters in Algorithm 3 are $\gamma = 2, \hat{\eta}_1 = \frac{1}{4}, \hat{\eta}_2 = \frac{3}{4}$, and $\mu = 0.1$. The initial interpolation points are $\boldsymbol{x}_{\text{int}}$, $\boldsymbol{x}_{\text{int}} \pm (\Delta_0, 0)^\top$, and $\boldsymbol{x}_{\text{int}} \pm (0, \Delta_0)^\top$. Table 2-5 lists six different classical (semi-)norms; this numerical example reports the corresponding results when using them to construct the least weighted H^2 norm quadratic model.

Table 2-5 Different (semi-)norms with corresponding coefficients in the coefficient set

Weight coefficients			Corresponding (semi-)norm	ID
$C_1 = \frac{1}{3}$	$C_2 = \frac{1}{3}$	$C_3 = \frac{1}{3}$	H^2 norm	(a)
$C_1 = \frac{1}{2}$	$C_2 = \frac{1}{2}$	$C_3 = 0$	H^1 norm	(b)
$C_1 = 0$	$C_2 = \frac{1}{2}$	$C_3 = \frac{1}{2}$	H^1 seminorm + H^2 seminorm	(c)
$C_1 = 1$	$C_2 = 0$	$C_3 = 0$	H^0 norm (L^2 norm)	(d)
$C_1 = 0$	$C_2 = 1$	$C_3 = 0$	H^1 seminorm	(e)
$C_1 = 0$	$C_2 = 0$	$C_3 = 1$	H^2 seminorm	(f)

Table 2-6 presents the results of this numerical experiment, including the number of function evaluations for each method, the obtained minimizer, and the best function value. In addition, Figure 2-7 shows the iteration plot for minimizing the Rosenbrock

Table 2-6 Results of the numerical experiment of Example 2.5

ID	Function value	Solution
(a)	0.0031	$(1.0495, 1.1040)^\top$
(b)	0.0169	$(1.0427, 1.0996)^\top$
(c)	0.0203	$(1.0418, 1.0990)^\top$
(d)	0.0078	$(1.0455, 1.1008)^\top$
(e)	0.0169	$(1.0427, 1.0996)^\top$
(f)	0.0147	$(1.0426, 1.0984)^\top$

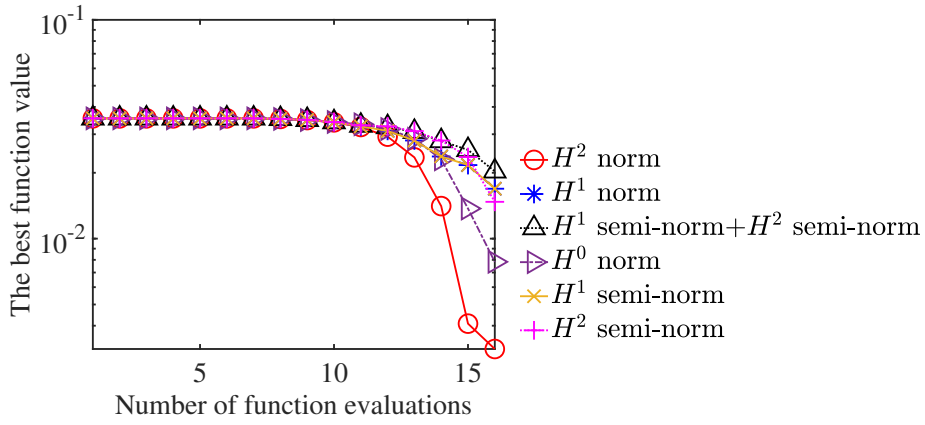


Figure 2-7 Minimizing Rosenbrock function

function (the current best function value versus iteration), where we display the first 16 evaluations. It can be seen that, in this example and under a (initially) small trust-region radius, the algorithm using the least H^2 norm updating quadratic model has an advantage.

Table 2-7 Test problems for Figure 2-8

ARGLINA	ARGLINA4	ARGLINB	ARGLINC	ARGTRIG
ARWHEAD	BDQRTIC	BDQRTICP	BDVALUE	BROWNAL
BROYDN3D	BROYDN7D	BRYBND	CHAINWOO	CHNROSNB
CHPOWELLB	CHROSEN	CRAGGLVY	CUBE	DQRTIC
EDENSCH	ENGVAL1	ERRINROS	EXPSUM	EXTROSNB
EXTTET	FIROSE	FLETCBV2	FLETCBV3	FLETCHCR
FREUROTH	GENBROWN	GENROSE	INDEF	INTEGREQ
LIARWHD	LILIFUN3	LILIFUN4	MOREBV	MOREBVL
NONDIA	PENALTY1	PENALTY2	PENALTY3	PENALTY3P
ROSENBROCK	SBRYBND	SBRYBNDL	SEROSE	SINQUAD
SROSENBR	STMOD	TOINTTRIG	TQUARTIC	TRIGSABS
TRIGSSQS	TRIROSE1	TRIROSE2	VARDIM	WOODS

Therefore, this illustrates the advantage of using the weight coefficients corresponding to the central KKT matrix in the least weighted H^2 norm updating quadratic model function.

To demonstrate more numerical performance of the algorithm based on the least weighted H^2 norm updating quadratic model function, we attempt to solve some classical test problems and use the Performance Profile to present the numerical results. Table 2-7 shows the test problems corresponding to the Performance Profile, whose dimensions range from 2 to 200. They are selected from classical and commonly used

unconstrained optimization test function sets [92, 174, 177, 180, 181, 183–186].

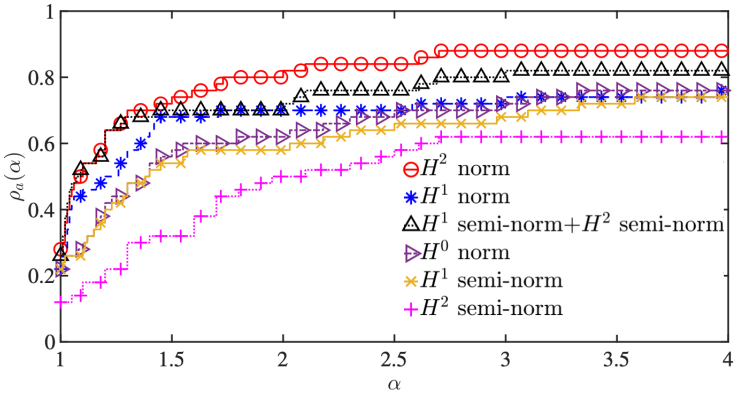


Figure 2-8 Solving test problems with different algorithms: Performance Profile

In each iteration, $2n + 1$ interpolation points are used. In addition, each algorithm starts from the given initial point $\boldsymbol{x}_{\text{int}}$ of the problem set, and the accuracy τ in the Performance Profile is set to 1%. The initial interpolation set is $\{\boldsymbol{x}_{\text{int}}, \boldsymbol{x}_{\text{int}} \pm \boldsymbol{e}_j, i = 1, \cdots, n\}$. The tolerances for the trust-region radius and the model gradient norm are both set to 10^{-8}. Their common initial trust-region radius is 1. The parameters in Algorithm 3 are $\gamma = 2, \hat{\eta}_1 = \frac{1}{4}, \hat{\eta}_2 = \frac{3}{4}$, and $\mu = 0.1$.

We can observe from Figure 2-8: among the listed derivative-free algorithms based on the least weighted H^2 norm updating quadratic model, when α is greater than approximately 1.5, the trust-region algorithm based on the least H^2 norm updating quadratic model function with weight coefficients $C_1 = C_2 = C_3 = \frac{1}{3}$ can solve more than 70% of the problems, performing better than the other algorithms. This demonstrates the advantage of this set of weight coefficients.

2.2.5 Conclusion

This section mainly discussed how to evaluate a set of weight coefficients of the weighted H^2 norm and how to find their optimal choice. We considered the weight coefficients appearing in the objective function of the interpolation model subproblem, where minimizing these objective functions yields the corresponding quadratic model function. We defined the KKT matrix distance, the KKT matrix error, and the barycenter of the coefficient region. Then we computed the barycenter of the coefficient region $\mathcal{C}$ of the least weighted H^2 norm updating quadratic model as the trust-region radius tends to 0. We provided related numerical experiments on minimizing the Rosenbrock function. We also used the Performance Profile to present the numerical performance comparison of algorithms corresponding to different models. For future work, we may consider comparing the weight coefficients of the least weighted H^2 norm updating quadratic model from other perspectives and further exploring more properties of underdetermined interpolation models in derivative-free optimization.

Table 2-8 The subproblem for the proposed under-determined quadratic model Q_k

Subproblem	$\min\limits_{Q\in\mathcal{Q}} \|\nabla^2 Q - \nabla^2 Q_{k-1}\|_F^2 + \alpha_k\|\nabla Q(\boldsymbol{x}_k)\|_2^2 + \beta_k\|(\boldsymbol{I}-\boldsymbol{P}_k)\nabla Q(\boldsymbol{x}_k)\|_2^2$ s. t. $Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k$
Parameters	$\rho_{k-1} = \frac{f(\boldsymbol{x}_k)-f(\boldsymbol{x}_{k-1})}{Q_{k-1}(\boldsymbol{x}_k)-Q_{k-1}(\boldsymbol{x}_{k-1})}$, $(\boldsymbol{x}_k \neq \boldsymbol{x}_{k-1})$ $\mathbb{1}$: indicator function ($\mathbb{1}([\text{true}]) = 1, \mathbb{1}([\text{false}]) = 0$) $\eta_0 \geq 0$: pre-given algorithm hyperparameter $\alpha_k = \mathbb{1}\{0 < \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 < \Delta_{k-1}\}\mathbb{1}\{\rho_{k-1} > \eta_0\}$ $\beta_k = \mathbb{1}\{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 = \Delta_{k-1}\}\mathbb{1}\{\rho_{k-1} > \eta_0\}$ $\boldsymbol{I}$: identity matrix; Δ_{k-1}: trust-region radius at iteration $k-1$ $\boldsymbol{P}_k = \frac{(\boldsymbol{x}_k-\boldsymbol{x}_{k-1})(\boldsymbol{x}_k-\boldsymbol{x}_{k-1})^\top}{\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2^2}$ projects a vector in $\mathfrak{R}^n$ onto span$\{\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\}$
Model	$Q(\boldsymbol{x}) := \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_k)^\top\nabla^2 Q(\boldsymbol{x}-\boldsymbol{x}_k) + \nabla Q(\boldsymbol{x}_k)^\top(\boldsymbol{x}-\boldsymbol{x}_k) + c$ c: constant term of the quadratic Q; $\mathcal{Q}$: set of quadratic functions
Interpolation	$\mathcal{X}_k = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$: interpolation point set at step k $\mathcal{X}_k = \mathcal{X}_{k-1} \cup \{\boldsymbol{x}_k\}\backslash\{\boldsymbol{x}_t^{(k)}\}$ (in a successful step) ; $\boldsymbol{x}_t^{(k)}$: discarded point
$\boldsymbol{x}_k$	$\boldsymbol{x}_k \in \{\arg\min\limits_{\boldsymbol{x}} Q_{k-1}(\boldsymbol{x}),\ \text{s. t. } \boldsymbol{x} \in \mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1})\}$ (in a successful step) $\mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1}) = \{\boldsymbol{x}, \|\boldsymbol{x}-\boldsymbol{x}_{k-1}\|_2 \leq \Delta_{k-1}\}$

2.3 Derivative-Free Methods Using New Under-Determined Quadratic Interpolation Models

2.3.1 Background and Motivation

Conn and Toint [171] proposed a least norm type under-determined quadratic interpolation model for model-based derivative-free trust-region algorithms, which is the soution of the subproblem about a quadratic function Q

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \|\nabla^2 Q\|_F^2 + \left\|\nabla Q(\boldsymbol{x}_k)\right\|_2^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k. \end{aligned} \tag{2-33}$$

Conn and Toint showed the numerical advantages of this model. Following their work, more under-determined quadratic interpolation models were proposed [93, 97, 153, 154]. At present, a common way to iteratively obtain an under-determined quadratic model Q_k is to solve a corresponding least norm (change/update) constrained optimization problem. For example, the model proposed by Powell [93] is obtained by solving the subproblem

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \left\|\nabla^2 Q - \nabla^2 Q_{k-1}\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k \end{aligned} \tag{2-34}$$

(as mentioned before). If $\nabla^2 Q_{k-1}$ in (2-34) is replaced by the zero matrix, then it becomes the least norm quadratic model proposed by Conn, Scheinberg, and Toint [156], which also has good numerical performance and is therefore widely used in practice.

As mentioned above, constructing the model function and obtaining the next iterate by solving the trust-region subproblem of the model are two main steps of model-based derivative-free trust-region methods. Most existing work on constructing models for derivative-free methods does not take into account the previous trust-region subproblem and its numerical solution (in most cases, the current iterate). This section considers constructing a quadratic model according to the position of the iterate given by the previous model within the previous trust region and the success of the iteration. The main motivation is to let the previous model and the numerical solution of its corresponding trust-region subproblem provide as much useful information as possible for the current model. Given that the ultimate goal of constructing models for derivative-free optimization is to make the next iterate given by solving each corresponding trust-region subproblem close to the true minimizer in the same trust region, we can use the position of the current iterate and the success information of the iteration to guide the construction of the model function.

This section attempts to provide a new perspective for understanding and analyzing the Conn-Toint model, and to propose a new model (a rough illustration is given in Table 2-8, and see Figure 2-9, where $\boldsymbol{x} \in \mathfrak{R}^n$, $\boldsymbol{P}_k \in \mathfrak{R}^{n\times n}$, and vectors are written as column vectors), with the formula of the model given based on the KKT conditions of the corresponding subproblem.

Remark 2.6. If for a pre-given algorithm hyperparameter $\eta_0 \geq 0$ we have $\rho_{k-1} > \eta_0$, then the algorithm obtains a successful step; otherwise (including the case $\boldsymbol{x}_k = \boldsymbol{x}_{k-1}$), we call it an unsuccessful step.

This work is the first to consider constructing under-determined interpolation models by exploiting the trust-region iteration properties. In short, our innovation is to construct the quadratic model of the current step by considering the position of the iterate generated in the previous step by the quadratic model and the success of the iteration.

The remaining part of this section is organized as follows. Section 2.3.2 discusses how to use the trust-region iteration properties to theoretically analyze and improve the Conn-Toint model. Section 2.3.3 analyzes the strict convexity of the subproblem of our model and gives a computation formula for obtaining our new model based on KKT conditions. Section 2.3.4 presents numerical results. Finally, we give a conclusion.

2.3.2 A Model Considering the Previous Trust-Region Iteration Properties

For derivative-free trust-region methods based on quadratic interpolation models, it is very important to ensure that the minimizer of the quadratic model within the trust

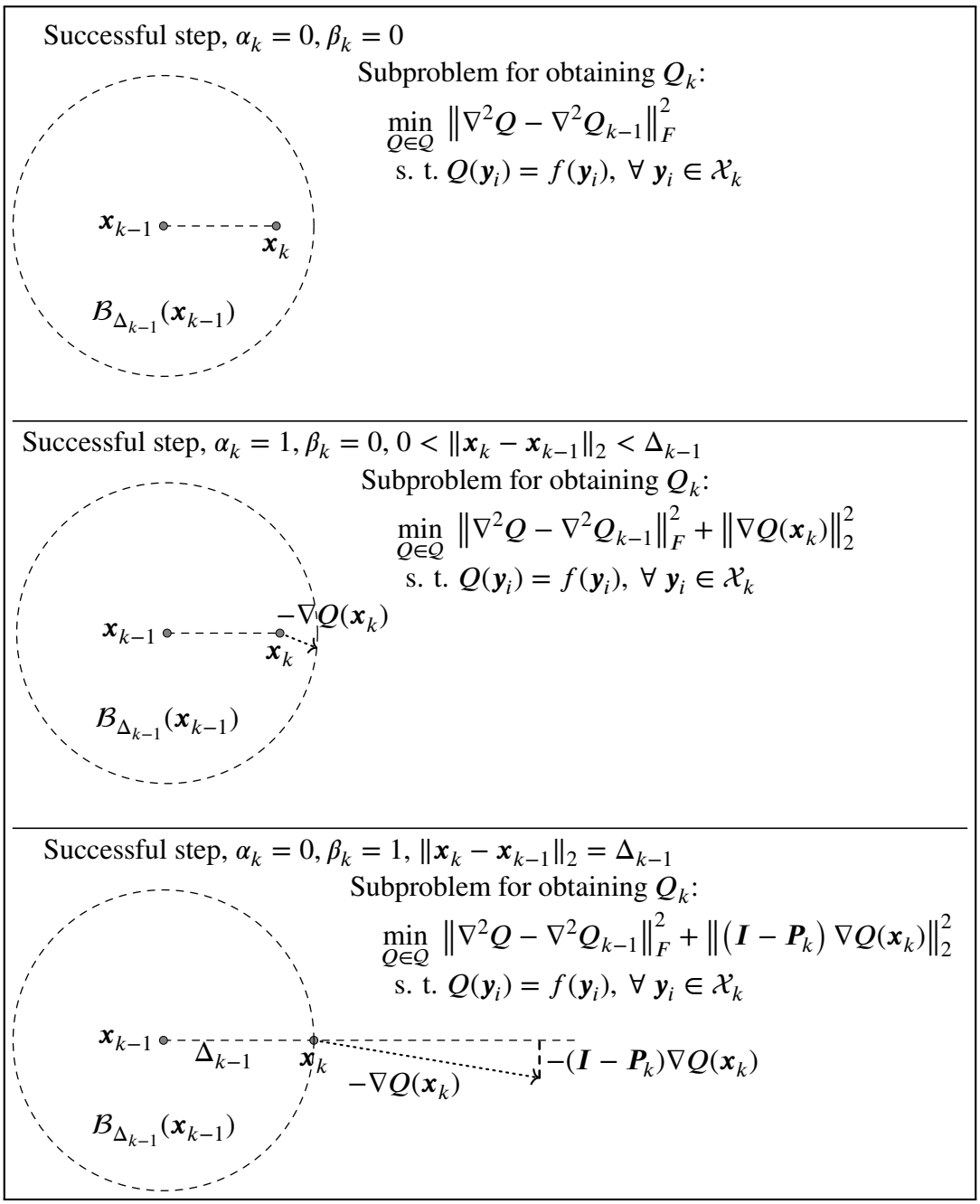


Figure 2-9 Illustration of the subproblem

region is sufficiently close to the minimizer of the objective function f in the same trust region. This is because the role of the quadratic interpolation model function in model-based derivative-free trust-region methods is to provide a new iterate with a smaller function value, obtained by minimizing the current model function within the current trust region. In fact, model-based trust-region methods attempt to iteratively use the minimizer of a good model function within the trust region to replace the minimizer of the black-box objective function f in the same trust region.

Therefore, we analyze the Conn-Toint model through trust-region iterations and the optimality of the newly obtained iterate. This helps us better understand the Conn-Toint model and derive an improved model. Let us further observe the following lemma, which considers the drawbacks and risks of traditional function value interpolation (1-3) in the one-dimensional case, given by Robinson [187] in 1979.

Lemma 2.23 (Risk of minimizer in quadratic interpolation)**.** *Let $x^* \in \mathfrak{R}$ and $\varepsilon > 0$. Suppose f is a continuous unimodal function from the interval $\left[x^* - \hat{\varepsilon}, x^* + \hat{\varepsilon}\right]$ to $\mathfrak{R}$, with minimizer x^*. Then unless f coincides with some quadratic function on $[x^* - \hat{\varepsilon}, x^* + \hat{\varepsilon}]$, there exist points $x_0 < x_1 < x_2$ within $\left[x^* - \hat{\varepsilon}, x^* + \hat{\varepsilon}\right]$, with $x_1 \neq x^*$, such that the unique minimizer of the quadratic function Q interpolating f at these three points is x_1.*

The above lemma shows that a quadratic interpolation model function (for more details see Definition 6.2 in the book by Conn, Scheinberg, and Vicente [20]) may fail to provide the correct minimizer. Moreover, by choosing interpolation points, one can even make the minimizer of the quadratic interpolation model fall at a completely wrong location. This reveals that function value constraints (1-3) may not directly characterize the optimality or minimizer of the approximated function.

Zhang [154] established a connection between the Conn – Toint model and the quadratic model with the least H^1 seminorm. Since then, there has been little direct analysis of the original approximation model first proposed by Conn and Toint. To the best of our knowledge, analyses of under-determined interpolation models for model-based derivative-free optimization rarely involve the properties of trust-region iterations. Here we attempt to analyze the Conn–Toint model through the lens of trust-region iteration properties. We will propose a new model by selectively regarding the under-determined quadratic model as a quadratic model or a linear model on successful iterations of the algorithm, as described earlier; this can be viewed as a combination of model and iteration properties.

Remark 2.7 (Optimality of the model in derivative-free quasi-Newton methods). Greenstadt [89] proposed a derivative-free quasi-Newton method that considers the optimality of the model at the iterate. However, in derivative-free trust-region optimization algorithms, the model's optimality at the iterate has not been directly used to construct the

model. This motivates us, when attempting to obtain an under-determined quadratic interpolation model, to consider the model's optimality and descent property at the known iterate.

In fact, at the k-th step when solving with the algorithm, if the step is successful, then the iterate and interpolation point $\boldsymbol{x}_k$ is special, because it is a point obtained by numerically solving a trust-region subproblem rather than a generic sample point used only for interpolation. We know that each newly added iterate/interpolation point $\boldsymbol{x}_k$ is the minimizer of the $(k-1)$-st model Q_{k-1} within the $(k-1)$-st trust region. Traditional trust-region methods design and use a subroutine to solve the trust-region subproblem

$$\begin{aligned} &\min_{\boldsymbol{d}} Q_{k-1}(\boldsymbol{x}_{k-1}+\boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \leq \Delta_{k-1} \end{aligned} \tag{2-35}$$

to obtain the new iterate $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \boldsymbol{d}_{k-1}$, where $\boldsymbol{d}_{k-1}$ is a solution of (2-35). In practice, we have reason to trust and use the information provided by the model from the previous step. Powell's least norm updating follows a similar idea. In other words, the premise of improving the previous model is to believe that the previous model (especially when it provides a successful step) is good.

We know that the term $\|\nabla^2 Q\|_F^2$ in the objective function appears in (2-33) in order to fill the remaining degrees of freedom of the under-determined quadratic model that are not fixed by the function value constraints (1-3). However, in the subproblem corresponding to the Conn–Toint model, the term $\left\|\nabla Q(\boldsymbol{x}_k)\right\|_2^2$ may seem unnecessary for constructing the model. In fact, from the perspective of trust-region iteration and model optimality, it is beneficial to the model; we give the reason below.

Regarding the solution of the trust-region subproblem, we recall the following classical result[6].

Proposition 2.24 (Solution of the trust-region subproblem)**.** *For a quadratic function Q, a vector $\boldsymbol{z} \in \mathfrak{R}^n$ satisfies*

$$\boldsymbol{z} \in \left\{ \arg\min_{\boldsymbol{x}} \; Q(\boldsymbol{x}),\ \text{s. t. } \boldsymbol{x} \in \mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1}) \right\}, \tag{2-36}$$

if and only if $\|\boldsymbol{z}-\boldsymbol{x}_{k-1}\|_2 \leq \Delta_{k-1}$ and there exists $\omega \geq 0$ such that $\boldsymbol{z}$ satisfies

$$\left(\nabla^2 Q + \omega \boldsymbol{I}\right)\left(\boldsymbol{z}-\boldsymbol{x}_{k-1}\right) + \nabla Q(\boldsymbol{x}_{k-1}) = \boldsymbol{0}_n, \tag{2-37}$$

and

$$\begin{aligned} \omega\left(\Delta_{k-1} - \|\boldsymbol{z}-\boldsymbol{x}_{k-1}\|_2\right) &= 0, \\ \nabla^2 Q + \omega \boldsymbol{I} &\succeq \boldsymbol{0}_{nn}, \end{aligned} \tag{2-38}$$

where $\boldsymbol{A} \succeq \boldsymbol{0}$ means that $\boldsymbol{A}$ is positive semidefinite.

[6]More details can be found in classical numerical optimization textbooks, e.g., Theorem 4.1 in the book by Nocedal and Wright [3].

Moreover,

$$\nabla Q(\boldsymbol{x}_k) = \nabla^2 Q\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right) + \nabla Q(\boldsymbol{x}_{k-1}).$$

Therefore, the subproblem (2-33) of the Conn – Toint model can be rewritten as the subproblem

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \left\|\nabla^2 Q\right\|_F^2 + \left\|\nabla^2 Q\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right) + \nabla Q(\boldsymbol{x}_{k-1})\right\|_2^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k, \end{aligned} \tag{2-39}$$

where the second term in the objective forces $\boldsymbol{x}_k$ to be close to a stable point of the quadratic model function Q_k when $\boldsymbol{x}_k$ lies in the interior of $\mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1})$ (i.e., $\left\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right\|_2 < \Delta_{k-1}$), because in this case $\omega = 0$. In this case we anticipate $\nabla^2 Q_k$ to be positive semidefinite; in fact, solving the subproblem (2-39) tends to achieve this goal, since $\boldsymbol{x}_k$ has a small or minimal function value within $\mathcal{X}_k$ and $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 < \Delta_{k-1}$.

We write subproblem (2-33) as (2-39) to make it consistent with (2-41). Analyzing a successful $\boldsymbol{x}_k$ with $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 < \Delta_{k-1}$ helps us establish new insight into the Conn–Toint model. Below we consider the case $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 = \Delta_{k-1}$, which helps us derive our model.

Based on the above discussion, we seek to propose a new model. According to the above, if $0 < \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 < \Delta_{k-1}$ and $\boldsymbol{x}_k$ is a successful step, i.e.,

$$\rho_{k-1} = \frac{f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k-1})}{Q_{k-1}(\boldsymbol{x}_k) - Q_{k-1}(\boldsymbol{x}_{k-1})} > \eta_0 \geq 0,$$

then it is reasonable to regard the second- and first-order information of the underdetermined quadratic model Q_{k-1} as guidance for obtaining $\boldsymbol{x}_k$. As discussed earlier, if $\nabla^2 Q_k$ is positive definite, then the regularization term $\left\|\nabla Q(\boldsymbol{x}_k)\right\|_2^2$ will make the minimizer of Q_k within $\mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1})$ close to $\boldsymbol{x}_k$. In this case, Q_k should inherit the second-order property of the quadratic model Q_{k-1}.

However, in the case where $\boldsymbol{x}_k$ is a successful step and $\left\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right\|_2 = \Delta_{k-1}$, according to the KKT conditions, if the algorithm still uses the Conn–Toint model, then the term $\|\nabla Q(\boldsymbol{x}_k)\|_2^2$, as we discussed before, implies treating $\boldsymbol{x}_k$ as a good approximate stable point of the objective function f; this actually misuses, in some sense, the information provided by Q_{k-1} and $\boldsymbol{x}_k$, because in this case $\boldsymbol{x}_k$ may not be close to a stable point of f.

In this case, according to Proposition 2.24, if the algorithm still wants the k-th model to follow the quadratic nature of the $(k-1)$-st model and tries to make the minimizer of Q_k within $\mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1})$ close to $\boldsymbol{x}_k$, then the regularization term can be the squared ℓ_2 norm of the left-hand side of (2-37), namely

$$\left\|\left(\nabla^2 Q + \omega \boldsymbol{I}\right)\left(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\right) + \nabla Q(\boldsymbol{x}_{k-1})\right\|_2^2 = \left\|\omega(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) + \nabla Q(\boldsymbol{x}_k)\right\|_2^2, \tag{2-40}$$

where $\omega \geq 0$ and we wish ω to satisfy $\nabla^2 Q_k + \omega \boldsymbol{I} \succeq \boldsymbol{0}$, with Q_k obtained by solving the subproblem

$$\begin{aligned} &\min_{Q \in \mathcal{Q}} \ \left\|\nabla^2 Q\right\|_F^2 + \left\|\omega(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) + \nabla Q(\boldsymbol{x}_k)\right\|_2^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i), \ \forall \ \boldsymbol{y}_i \in \mathcal{X}_k \end{aligned} \tag{2-41}$$

to obtain. However, the uncertainty of the parameter ω makes the above idea difficult to implement directly, for the following reasons. Note that our goal is to obtain the model Q_k given ω, which differs from the classical setting of obtaining the minimizer of a known quadratic function within a trust region.

On the one hand, if we set $\omega = 0$ in the objective of (2-41) to provide a quadratic interpolation model, then it is exactly the Conn – Toint subproblem (2-33), and thus yields the corresponding model function; but we have discussed that this is inappropriate when $\boldsymbol{x}_k$ is a successful step with $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 = \Delta_{k-1}$.

On the other hand, if we consider a not-necessarily-zero ω, it is hard to choose a suitable ω to ensure that $\nabla^2 Q_k + \omega \boldsymbol{I}$ is positive semidefinite, and different ω will yield different model functions Q_k after solving (2-41).

Moreover, for our under-determined quadratic interpolation model, the number of interpolation points used to construct each model is fewer than $\frac{1}{2}(n+1)(n+2)$. Therefore, a unique quadratic model cannot be determined by interpolation alone.

According to the above analysis, the uncertainty of ω leads to non-uniqueness of the quadratic model function, which makes adding the second term in the objective of (2-41) to capture the quadratic information provided by Q_{k-1} not necessarily accurate, even in the case where the iterate $\boldsymbol{x}_k$ is a successful iteration. The above situation motivates us to propose a new model: depending on the relationship between $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2$ and Δ_{k-1}, we selectively regard the previous model as a linear model or a quadratic model[7].

For the case $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 = \Delta_{k-1}$ with a successful step, we consider it reasonable to assume that "the direction $\boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ still provides an approximate gradient descent direction at the latest iterate $\boldsymbol{x}_k$". Specifically, the algorithm has obtained, in a successful step, a new iterate that sufficiently reduces the function value. In this case, the obtained $(k-1)$-st quadratic model is considered to provide a relatively accurate gradient descent direction. Moreover, the algorithm (in such a successful step) should have obtained only a good first-order approximation.

Therefore, we consider it reasonable and practical to make the new model Q_k as consistent as possible with the descent property or information of Q_{k-1}. In other words, besides minimizing $\|\nabla Q_k(\boldsymbol{x}_k)\|_2$ while satisfying the function value constraints (1-3), the direction of $\boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ should be close to the direction of $-\nabla Q_k(\boldsymbol{x}_k)$. In addition, if $Q_k(\boldsymbol{x})$ also attains its minimum at $\boldsymbol{x}_k$ within the trust region $\mathcal{B}_{\Delta_{k-1}}(\boldsymbol{x}_{k-1})$, then

[7] We selectively regard the information and optimality at $\boldsymbol{x}_k$ provided by Q_{k-1} as being given by a reliable linear or quadratic model according to whether $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2 = \Delta_{k-1}$ holds.

$\left(\boldsymbol{I}-\boldsymbol{P}_k\right)\nabla Q(\boldsymbol{x}_k)=\boldsymbol{0}_n$, where $\boldsymbol{P}_k=(\boldsymbol{x}_k-\boldsymbol{x}_{k-1})(\boldsymbol{x}_k-\boldsymbol{x}_{k-1})^\top/\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2^2$ projects vectors in $\mathfrak{R}^n$ onto span $\left\{\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\right\}$, $\boldsymbol{0}_n\in\mathfrak{R}^n$ is the zero vector. Based on this, we [188] propose obtaining the model Q_k by solving the subproblem

$$\begin{aligned}&\min_{Q\in\mathcal{Q}}\ \left\|\nabla^2 Q-\nabla^2 Q_{k-1}\right\|_F^2+\alpha_k\left\|\nabla Q(\boldsymbol{x}_k)\right\|_2^2+\beta_k\left\|\left(\boldsymbol{I}-\boldsymbol{P}_k\right)\nabla Q(\boldsymbol{x}_k)\right\|_2^2\\&\text{s. t. } Q(\boldsymbol{y}_i)=f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i\in\mathcal{X}_k\end{aligned}\tag{2-42}$$

where the coefficients α_k and β_k are defined in Table 2-8, i.e.,

$$\alpha_k=\begin{cases}1, \text{ if } 0<\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2<\Delta_{k-1} \text{ and } \rho_{k-1}>\eta_0,\\0, \text{ otherwise,}\end{cases}$$

$$\beta_k=\begin{cases}1, \text{ if } \|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2=\Delta_{k-1} \text{ and } \rho_{k-1}>\eta_0,\\0, \text{ otherwise,}\end{cases}$$

$|\mathcal{X}_k|<\frac{1}{2}(n+1)(n+2)$ and

$$\rho_{k-1}=\frac{f(\boldsymbol{x}_k)-f(\boldsymbol{x}_{k-1})}{Q_{k-1}(\boldsymbol{x}_k)-Q_{k-1}(\boldsymbol{x}_{k-1})}.$$

Please note that we keep the term $\left\|\nabla^2 Q-\nabla^2 Q_{k-1}\right\|_F^2$ in the objective of (2-42) in order to inherit some merits of Powell's least norm updating, since the historical information of the interpolation model Q_{k-1} is still useful. When $\boldsymbol{x}_k$ corresponds to an unsuccessful step, the subproblem reduces to Powell's Frobenius norm updating underdetermined model, because in this case $\alpha_k=0$ and $\beta_k=0$.

Remark 2.8. When $\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2=\Delta_{k-1}$ and $\omega=\left\|\boldsymbol{P}_k\nabla Q(\boldsymbol{x}_k)\right\|_2/\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2$, minimizing the objective in (2-42) reduces the ℓ_2 norm value of the left-hand side of (2-37), because if $\omega=\left\|\boldsymbol{P}_k\nabla Q(\boldsymbol{x}_k)\right\|_2/\|\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\|_2$, then

$$\left\|\left(\nabla^2 Q+\omega\boldsymbol{I}\right)\left(\boldsymbol{x}_k-\boldsymbol{x}_{k-1}\right)+\nabla Q(\boldsymbol{x}_{k-1})\right\|_2^2=\left\|(\boldsymbol{I}-\boldsymbol{P}_k)\nabla Q(\boldsymbol{x}_k)\right\|_2^2.$$

The above analysis shows that, in this case, minimizing (2-41) is equivalent to minimizing (2-42) to obtain our model Q_k.

We should note that a quadratic model interpolation obtained solely via function value interpolation based on (1-3) or a traditional least norm scheme may already reflect the curvature and shape of a quadratic function in some sense. However, without increasing the number of interpolation points, considering the optimality at $\boldsymbol{x}_k$ at the k-th step can, in a certain sense, yield a better approximation than not considering optimality. The numerical results in Section 2.3.4 demonstrate the advantage of our model.

2.3.3 Convexity of the Subproblem and the Computation Formula of the Model

Next, we analyze the strict convexity of the subproblem (2-42) to show that the subproblem (2-42) has a unique solution.

Theorem 2.25 (Strict convexity of the subproblem objective)**.** *Given* $\alpha_k \geq 0, \beta_k \geq 0, \boldsymbol{x}_k \in \mathfrak{R}^n, \boldsymbol{I} \in \mathfrak{R}^{n\times n}, \boldsymbol{P}_k \in \mathfrak{R}^{n\times n}$*, assume the set* $\mathcal{X}_k$ *is not contained in a subspace of dimension less than n. For all quadratic functions that satisfy the interpolation conditions in (2-42), the objective*

$$\left\|\nabla^2 Q - \nabla^2 Q_{k-1}\right\|_F^2 + \alpha_k \left\|\nabla Q(\boldsymbol{x}_k)\right\|_2^2 + \beta_k \left\|\left(\boldsymbol{I} - \boldsymbol{P}_k\right) \nabla Q(\boldsymbol{x}_k)\right\|_2^2$$

as a function of the quadratic Q *is strictly convex.*

Proof. Let $\mathcal{F}(Q)$ denote the objective. We need to prove that for $0 < \varphi < 1$ and quadratic functions $Q_a \neq Q_b$ satisfying the constraints in (2-42),

$$\mathcal{F}(\varphi Q_a + (1-\varphi)Q_b) < \varphi\mathcal{F}(Q_a) + (1-\varphi)\,\mathcal{F}(Q_b). \tag{2-43}$$

The theorem's assumption implies $\nabla^2 Q_a \neq \nabla^2 Q_b$. Hence,

$$\begin{aligned}
&\mathcal{F}(\varphi Q_a + (1-\varphi)Q_b) - \left(\varphi\mathcal{F}(Q_a) + (1-\varphi)\mathcal{F}(Q_b)\right) \\
&= \left(\varphi^2 - \varphi\right)\left[\left\|\nabla^2 Q_a - \nabla^2 Q_b\right\|_F^2 + \alpha_k \left\|\left(\nabla Q_a(\boldsymbol{x}_k) - \nabla Q_b(\boldsymbol{x}_k)\right)\right\|_2^2\right. \\
&\quad \left. + \beta_k \left\|\left(\boldsymbol{I} - \boldsymbol{P}_k\right)\left(\nabla Q_a(\boldsymbol{x}_k) - \nabla Q_b(\boldsymbol{x}_k)\right)\right\|_2^2\right] < 0.
\end{aligned}$$

Thus (2-43) holds, and strict convexity follows. □

We then obtain the strict convexity of subproblem (2-42) and the uniqueness of its solution, which ensures that our under-determined interpolation model is uniquely determined at every step of the algorithm.

In order to derive a computable formula for obtaining the model, we give the following theorem based on the KKT conditions.

Theorem 2.26 (Computation formula for the quadratic model)**.** *The quadratic function*

$$Q(\boldsymbol{x}) = \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}_k\right)^\top \boldsymbol{H}\left(\boldsymbol{x} - \boldsymbol{x}_k\right) + \boldsymbol{g}^\top\left(\boldsymbol{x} - \boldsymbol{x}_k\right) + c,$$

where

$$\boldsymbol{H} = \nabla^2 Q_{k-1} + \frac{1}{4}\sum_{j=1}^{m} \lambda_j \left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)\left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)^\top,$$

$\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_n)^\top \in \mathfrak{R}^n$*, and* $(\boldsymbol{\lambda}, c, \boldsymbol{g})^\top \in \mathfrak{R}^{m+1+n}$ *is the solution of the KKT system*

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{E} & \boldsymbol{X} \\ \boldsymbol{E}^\top & 0 & \boldsymbol{0}_n^\top \\ \boldsymbol{X}^\top & \boldsymbol{0}_n & \boldsymbol{B} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ c \\ \boldsymbol{g} \end{pmatrix} = \begin{pmatrix} \boldsymbol{r} \\ 0 \\ \boldsymbol{0}_n \end{pmatrix} \tag{2-44}$$

is the solution of subproblem (2-42), where

$$\begin{aligned}
\boldsymbol{r} &= \begin{pmatrix} f(\boldsymbol{y}_1) - \frac{1}{2}\left(\boldsymbol{y}_1 - \boldsymbol{x}_k\right)^\top \nabla^2 Q_{k-1}\left(\boldsymbol{y}_1 - \boldsymbol{x}_k\right) \\ \vdots \\ f(\boldsymbol{y}_m) - \frac{1}{2}\left(\boldsymbol{y}_m - \boldsymbol{x}_k\right)^\top \nabla^2 Q_{k-1}\left(\boldsymbol{y}_m - \boldsymbol{x}_k\right) \end{pmatrix}, \\
\boldsymbol{X} &= \left(\boldsymbol{y}_1 - \boldsymbol{x}_k, \cdots, \boldsymbol{y}_m - \boldsymbol{x}_k\right)^\top, \\
\boldsymbol{B} &= -2\alpha_k \boldsymbol{I} - 2\beta_k (\boldsymbol{I} - \boldsymbol{P}_k)^\top (\boldsymbol{I} - \boldsymbol{P}_k),
\end{aligned}$$

the entries of the matrix $\boldsymbol{A}$ are

$$\boldsymbol{A}_{ij} = \frac{1}{8}\left[\left(\boldsymbol{y}_i - \boldsymbol{x}_k\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)\right]^2, \ 1 \le i, j \le m,$$

and $\boldsymbol{E}$ is the vector whose entries are all ones.

Proof. The Lagrangian function corresponding to subproblem (2-42) is

$$\begin{aligned}
\mathcal{L}(c, \boldsymbol{g}, \boldsymbol{H}) = &\left\|\boldsymbol{H} - \nabla^2 Q_{k-1}\right\|_F^2 + \alpha_k \|\boldsymbol{g}\|_2^2 + \beta_k \left\|\left(\boldsymbol{I} - \boldsymbol{P}_k\right)\boldsymbol{g}\right\|_2^2 \\
&- \sum_{j=1}^{m} \lambda_j \left[\frac{1}{2}\left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)^\top \boldsymbol{G}\left(\boldsymbol{y}_j - \boldsymbol{x}_k\right) + \boldsymbol{g}^\top\left(\boldsymbol{y}_j - \boldsymbol{x}_k\right) + c\right].
\end{aligned}$$

From the KKT conditions, we have

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial c} &= \sum_{j=1}^{m} \lambda_j = 0, \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{g}} &= 2\alpha_k \boldsymbol{g} + 2\beta_k \left(\boldsymbol{I} - \boldsymbol{P}_k\right)^\top \left(\boldsymbol{I} - \boldsymbol{P}_k\right)\boldsymbol{g} - \sum_{j=1}^{m} \lambda_j \left(\boldsymbol{y}_j - \boldsymbol{x}_k\right) = \boldsymbol{0}_n, \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{H}} &= 2\boldsymbol{H} - 2\nabla^2 Q_{k-1} - \frac{1}{2}\sum_{j=1}^{m} \lambda_j \left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)\left(\boldsymbol{y}_j - \boldsymbol{x}_k\right)^\top = \boldsymbol{0}_{nn},
\end{aligned}$$

and

$$f(\boldsymbol{y}_i) = c + \boldsymbol{g}^\top\left(\boldsymbol{y}_i - \boldsymbol{x}_k\right) + \frac{1}{2}\left(\boldsymbol{y}_i - \boldsymbol{x}_k\right)^\top \boldsymbol{H}\left(\boldsymbol{y}_i - \boldsymbol{x}_k\right), \ i = 1, \cdots, m.$$

From the above relations we can derive the KKT system (2-44), proving the theorem. □

For the model subproblem discussed above, we only need to modify the KKT system to obtain the formula for the corresponding quadratic model function, which is easy to implement.

2.3.4 Numerical results

This section presents numerical results, including the outcome of solving a numerical example and the Performance Profile and Data Profile for solving a set of test problems.

The following example illustrates the advantage of the model and method proposed in this section.

Example 2.6 (Initial iterative performance)**.** We present an unconstrained optimization problem as an example. The objective is the 2D Rosenbrock function

$$f(\boldsymbol{x}) = \left(1 - x_1\right)^2 + 100\left(x_2 - x_1^2\right)^2,$$

with minimum value 0, and the initial trust-region radius is $\Delta_0 = 1$.

Step 1. The initial interpolation points $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{y}_3$ are

$$\boldsymbol{y}_1 = \begin{pmatrix} 0 \\ 7 \end{pmatrix}, \ \boldsymbol{y}_2 = \begin{pmatrix} 1 \\ 7 \end{pmatrix}, \ \boldsymbol{y}_3 = \begin{pmatrix} 0 \\ 8 \end{pmatrix}.$$

Step 2. Obtain Q_0 by solving

$$\begin{aligned} &\min_{Q \in \mathcal{Q}} \ \left\|\nabla^2 Q\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i), \ \forall\, i = 1, 2, 3. \end{aligned}$$

Step 3. Obtain $\boldsymbol{d}^*$ by solving

$$\begin{aligned} &\min_{\boldsymbol{d}} \ Q_0(\boldsymbol{y}_2 + \boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \leq \Delta_0 \end{aligned}$$

and set $\boldsymbol{y}_4 = \boldsymbol{y}_2 + \boldsymbol{d}^*$. If this step is successful, replace the interpolation point with the largest function value by $\boldsymbol{y}_4$. Note that we choose $\boldsymbol{y}_2$ as the trust-region center because it has the smallest function value among $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{y}_3$.

Step 4. Construct different Q_1 using the different methods listed in Table 2-9 for comparison ; if the new point's function value is smaller than some point in the interpolation set, add the new point to the interpolation set and discard the point farthest (in Euclidean distance) from the point with the smallest function value.

Step 5. Obtain $\boldsymbol{d}^*$ again by solving

$$\begin{aligned} &\min_{\boldsymbol{d}} \ Q_1(\boldsymbol{x}_{\text{small}} + \boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \leq \Delta_0, \end{aligned}$$

where $\boldsymbol{x}_{\text{small}}$ is the point with the smallest function value among the current sample points, set $\boldsymbol{y}_5 = \boldsymbol{x}_{\text{small}} + \boldsymbol{d}^*$, and denote by $\boldsymbol{x}_{\min}$ the point satisfying

$$f(\boldsymbol{x}_{\min}) = \min\left\{f(\boldsymbol{y}_1), f(\boldsymbol{y}_2), f(\boldsymbol{y}_3), f(\boldsymbol{y}_4), f(\boldsymbol{y}_5)\right\}.$$

Note that all algorithms use the same model Q_0 in the first step; they share the same $\boldsymbol{y}_4 = (1.6552, 6.2446)^\top$ with $f(\boldsymbol{y}_4) = 1.23 \times 10^3$, obtained by minimizing the least-Frobenius norm quadratic model within the trust region. Moreover, they construct different model functions in the second step according to Table 2-9.

Table 2-9 Results of Example 2.6: using different models

Model	Objective of the subproblem	$f(\boldsymbol{x}_{\min})$
Our model (considering optimality)	Objective of (2-42)	2.09
Least Frobenius norm quadratic model	$\|\nabla^2 Q\|_F^2$ [156, 160, 189]	34.1
Powell's quadratic model	$\|\nabla^2 Q - \nabla^2 Q_{k-1}\|_F^2$ [93]	34.1
Least H^2 norm updating quadratic model	$\|Q - Q_{k-1}\|_{H^2}^2$ [97]	5.33
Conn and Toint's quadratic model	$\|\nabla^2 Q\|_F^2 + \|\nabla Q(\boldsymbol{x}_k)\|_2^2$ [171]	74.9
Shared function value constraints: $Q(\boldsymbol{y}_i) = f(\boldsymbol{y}_i),\ \forall\ \boldsymbol{y}_i \in \mathcal{X}_k$		

We observe that the function value of the iterate obtained by the algorithm using the quadratic model proposed in this section is smaller than the corresponding function values obtained by algorithms that use models not considering trust-region iteration.

Given that the interpolation model functions proposed in this section are mainly suitable for model-based derivative-free trust-region algorithms, we provide here a framework of a model-based derivative-free trust-region algorithm used for testing in this section (the framework used in these experiments is not exactly the same as the earlier one), as shown in Algorithm 4. For more on derivative-free trust-region methods, see the survey by Larson, Menickelly, and Wild [128] and the monograph by Conn, Scheinberg, and Vicente [20], among others.

Algorithm 4 Model-based derivative-free trust-region algorithm

(Initialization)

Set parameters $\varepsilon, \varepsilon_{\text{stop}} > 0$, η_0, η_1: $0 \leq \eta_0 \leq \eta_1 < 1$ and γ_0, γ_1: $0 < \gamma_0 < 1 \leq \gamma_1$, $\bar{\gamma}$. Choose an initial point $\boldsymbol{x}_{\text{int}}$ and the value $f(\boldsymbol{x}_{\text{int}})$. Choose an initial trust-region radius $\Delta_0 > 0$ and an upper bound $\Delta_{\text{up}} > \Delta_0$. Choose an initial well-poised interpolation set $\mathcal{X}_0$ [20]. Determine $\boldsymbol{x}_0 \in \mathcal{X}_0$ so that it has the smallest objective value among the current points, i.e., $f(\boldsymbol{x}_0) = \min\limits_{\boldsymbol{y}_i \in \mathcal{X}_0} f(\boldsymbol{y}_i)$.

Step 1. (Construct the model)

Construct an interpolation model $Q_k(\boldsymbol{x})$ using the set $\mathcal{X}_k$.

while $\|\nabla Q_k(\boldsymbol{x}_k)\|_2 < \varepsilon$ **do**

  if Q_k is accurate on $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_k)$ **then**

    Set $\Delta_k = \bar{\gamma}\Delta_k$.

  else

    Update $\mathcal{X}_k$ so that Q_k is accurate on $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_k)$.

  end if

end while

Step 2. (Stopping criterion)

if $\Delta_k < \varepsilon_{\text{stop}}$ and $\|\nabla Q_k(\boldsymbol{x}_k)\|_2 < \varepsilon_{\text{stop}}$ **then**

terminate the algorithm.

end if

Step 3. (Minimize the model within the trust region)

Compute $\boldsymbol{d}_k$ such that

$$Q_k(\boldsymbol{x}_k + \boldsymbol{d}_k) = \min_{\|\boldsymbol{d}\|_2 \le \Delta_k} Q_k(\boldsymbol{x} + \boldsymbol{d}).$$

if $\boldsymbol{d}_k = \boldsymbol{0}$ **then**

Go to **Step 4** and handle it as the case $\rho_k \le \eta_0$;

else

Evaluate $f(\boldsymbol{x}_k + \boldsymbol{d}_k)$, and set

$$\rho_k = \frac{f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k + \boldsymbol{d}_k)}{Q_k(\boldsymbol{x}_k) - Q_k(\boldsymbol{x}_k + \boldsymbol{d}_k)}.$$

end if

Step 4. (Update the interpolation set and trust-region radius)

if $\rho_k < \eta_1$ and Q_k is not accurate on $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_k)$ **then**

Generate new interpolation points in $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_k)$ and add them to $\mathcal{X}_k$ to improve the poisedness of $\mathcal{X}_{k+1}$, then discard one interpolation point.

end if

if $\rho_k \ge \eta_1$ **then**

Enlarge the trust-region radius: set $\Delta_{k+1} = \min\{\Delta_{\text{up}}, \gamma_1 \Delta_k\}$;

Update the interpolation set: set $\mathcal{X}_{k+1} = \mathcal{X}_k \cup \{\boldsymbol{x}_{k+1}\} \backslash \{\arg\max_{\boldsymbol{x}} \ \|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|_2\}$.

else if Q_k is accurate on $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_k)$ **then**

Shrink the trust-region radius: set $\Delta_{k+1} = \gamma_0 \Delta_k$;

else

Set $\Delta_{k+1} = \Delta_k$.

end if

Step 5. (Update the current iterate)

if $\rho_k > \eta_0$ **then**

Choose $\boldsymbol{x}_{k+1}$ to be a point satisfying $f(\boldsymbol{x}_{k+1}) \le f(\boldsymbol{x}_k + \boldsymbol{d}_k)$;

else

Set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$.

end if

Set $k = k + 1$ and go to **Step 1**.

Remark 2.9. An accurate model in the algorithm usually refers to a fully linear model (see more details in Chapter 6 of Conn, Scheinberg, and Vicente [20], especially Definition 6.1), which here corresponds to the case where the coefficient matrix of the KKT system (2-44) is invertible. In Step 1 of the current implementation, if the coefficient

matrix is not invertible, the interpolation points are perturbed to improve the interpolation set and the model (with $\bar{\gamma} = 1$ in testing) in order to obtain an accurate model. In Step 4, the step that improves the poisedness of $\mathcal{X}_k$ is also called the model-improvement step (see Chapter 6 of Conn, Scheinberg, and Vicente [20], especially Algorithm 6.3, for more details). To simplify and focus on model comparison, in the current implementation this step is carried out as follows: when $\rho_k < \eta_1$, remove $\boldsymbol{x}_{\text{far}} = \arg\max_{\boldsymbol{x}\in\mathcal{X}_k} \|\boldsymbol{x} - \boldsymbol{x}_k\|_2$ from $\mathcal{X}_k$ and include $\boldsymbol{x}_k + \boldsymbol{d}_k$, even if the algorithm does not accept $\boldsymbol{x}_k + \boldsymbol{d}_k$ as the next iterate. In fact, we have already obtained the function value $f(\boldsymbol{x}_k + \boldsymbol{d}_k)$, and we should make use of such a known evaluation [160]. In Step 1, it uses the Frobenius norm quadratic model as the initial quadratic model $Q_0(\boldsymbol{x})$ for the initial iteration (the least-Frobenius quadratic model based on $\mathcal{X}_0$), and then constructs the corresponding under-determined model in subsequent iterations (e.g., obtains our proposed model in this section by solving (2-44)).

If $m \geq n + 1$, then at least the same convergence results can be obtained as for the corresponding trust-region algorithms whose models are obtained by solving least norm type subproblems. For such models, it can be shown under appropriate assumptions that any limit point $\boldsymbol{x}^*$ of the derivative-free trust-region algorithm is a stationary point, i.e., $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ [20].

We provide the corresponding Performance Profile and Data Profile to observe the numerical performance of the trust-region algorithm using the new model function that accounts for its optimality within the trust region when solving the unconstrained test problems (1-1).

The method using the model function proposed in this section (Algorithm 4, indexed by $a = 1$) is compared with the same framework but using the least-Frobenius norm quadratic model [156, 160, 189], the least Frobenius norm updating quadratic model (Powell's model) [93], the least H^2 norm updating quadratic model [97], the Conn-Toint model [171], as well as the state-of-the-art derivative-free methods NEWUOA [94], Fminsearch (MATLAB Optimization Toolbox) [190, 191], Fminunc (MATLAB Optimization Toolbox) [190], CMA-ES [131], and NMSMAX [192–194], indexed by $a = 2$ to $a = 10$, respectively.

In Algorithm 4, the parameters for shrinking or enlarging the radius and for deciding whether to accept the obtained point are $\eta_1 = 0.75$ and $\eta_0 = 0$, respectively. In addition, the shrinking and enlarging factors of the trust-region radius are $\gamma_0 = 0.8$ and $\gamma_1 = 1.5$, respectively. The tolerances for the trust-region radius and the gradient norm of the model function are 10^{-6} and 10^{-5}, respectively. The model accuracy parameter is $\varepsilon = 10^{-8}$. They use $m = 2n + 1$ interpolation points at each iteration, and the initial interpolation points are the origin and the points $\pm\frac{1}{2}\Delta_0 \boldsymbol{e}_i, i = 1, \cdots, n$, where $\boldsymbol{e}_i \in \Re^n$ denotes the vector with only its i-th element equal to 1 and the others 0.

In the numerical tests, NEWUOA uses $2n + 1$ interpolation points (the same initial points as the method proposed in this section) to construct the corresponding quadratic model function, with $\rho_{\text{end}} = 10^{-6}$ and ρ_{beg} the same as the initial radius of the first five methods using different models. For Fminsearch, the tolerances for the function value and the iterates are 10^{-6}. Fminunc is set to use a quasi-Newton method with finite-difference gradients of step length 0.1, and the other related tolerances are of the same order as for Fminsearch. The parameters of CMA-ES use the default settings [131], with the relevant stopping criterion for the function value set to 10^{-5}. For NMSMAX, the iteration terminates when the scale of the simplex is less than or equal to 10^{-6}, and the initial simplex is a regular simplex with edges of equal length. All compared algorithms share the same stopping rule on the number of function evaluations, namely, the total number of evaluations does not exceed $100n$, where n is the dimension of the corresponding problem.

We choose the test set $\mathcal{P}$ listed in Table 2-10 (containing 110 problems, including 51 different types, with dimensions ranging from 2 to 800) to test the effectiveness of our algorithm for unconstrained derivative-free problems. Note that the average dimension of our test problems is about 74, with a standard deviation of about 129. They come from classic and commonly used sets of unconstrained optimization test functions, and most of the objective functions f in the test problems are smooth (BROYDN7D and TRIGSABS are piecewise smooth).

Table 2-10 110 test problems for Figure 2-10 and Figure 2-11

Problem	Dimension	$f(\boldsymbol{x}_{\text{int}})$	$f(\boldsymbol{x}^*)$
ARGLINA [177, 178]	2	1.00×10^{1}	2.00
ARGLINB [177, 178]	2	2.14×10^{2}	6.67×10^{1}
BDVALUE [177, 178]	2	2.43×10^{-2}	2.18×10^{-15}
BROYDN3D [177, 178]	2	1.30×10^{1}	6.03×10^{-17}
BROYDN7D [177, 184]	2	7.81	6.59×10^{1}
CHEBQUAD [177, 178]	2	1.98×10^{-1}	9.50×10^{-18}
CHROSEN [94]	2	2.00×10^{1}	2.09×10^{-18}
CURLY10 [177]	2	-1.01×10^{-5}	-2.01×10^{2}
CURLY20 [177]	2	-1.01×10^{-5}	-2.01×10^{2}
CURLY30 [177]	2	-1.01×10^{-5}	-2.01×10^{2}
DIXMAANE [177]	2	7.00	1.00
DIXMAANF [177]	2	7.00	1.00
DIXMAANG [177]	2	7.00	1.00
DIXMAANH [177]	2	7.00	1.00
DIXMAANI [177]	2	6.00	1.00

Table 2-10 (Continued)

DIXMAANJ [177]	2	6.00	1.00
DIXMAANK [177]	2	6.00	1.00
DIXMAANL [177]	2	6.00	1.00
DIXMAANM [177]	2	6.00	1.00
DIXMAANN [177]	2	6.00	1.00
DIXMAANO [177]	2	6.00	1.00
DIXMAANP [177]	2	6.00	1.00
ENGVAL1 [177]	2	5.90×10^{1}	0.00
EXPSUM [182]	2	5.00	0.00
INTEGREQ [177, 178]	2	2.06×10^{-2}	2.96×10^{-18}
MOREBV [177, 178]	2	2.43×10^{-2}	2.18×10^{-15}
MOREBVL [180]	2	7.13	4.10×10^{-17}
NONCVXU2 [177]	2	3.95×10^{1}	4.63
NONCVXUN [177]	2	5.32×10^{1}	4.63
NONDIA [177]	2	1.02×10^{4}	1.00
POWER [177]	2	5.00	0.00
SBRYBND [177, 178]	2	2.67×10^{14}	4.99×10^{-12}
SPARSINE [177]	2	1.24×10^{1}	9.47×10^{-15}
TOINTTRIG [184]	2	1.03×10^{1}	-2.00×10^{1}
TRIGSSQS [94]	2	3.22×10^{3}	1.25×10^{-17}
TRIROSE1 [186]	2	1.55×10^{3}	2.27×10^{-15}
ARGLINA [177, 178]	3	1.50×10^{1}	3.00
ARGLINB [177, 178]	3	3.03×10^{3}	1.15
ARGLINC [177]	3	8.60×10^{1}	2.67
BROYDN3D [177, 178]	3	1.40×10^{1}	1.64×10^{-15}
EXTTET [185]	3	2.91	2.56
NONCVXU2 [177]	3	1.18×10^{2}	6.95
NONCVXUN [177]	3	1.57×10^{2}	6.95
POWER [177]	3	1.40×10^{1}	0.00
SPARSINE [177]	3	2.48×10^{1}	1.23×10^{-13}
TOINTGSS [177]	3	1.10×10^{1}	2.00
FLETCHCR [177]	5	4.00×10^{2}	1.21×10^{-12}
TOINTTRIG [184]	5	-1.37×10^{1}	-2.50×10^{2}
BROYDN3D [177, 178]	4	1.50×10^{1}	1.79×10^{-13}
NONCVXUN [177]	4	2.90×10^{2}	9.27
POWER [177]	4	3.00×10^{1}	0.00
FLETCHCR [177]	6	5.00×10^{2}	7.91×10^{-13}

Table 2-10 (Continued)

POWELLSG [177, 178]	6	2.15×10^{2}	1.39×10^{-15}
SBRYBND [177, 178]	6	2.68×10^{14}	3.60×10^{3}
BROYDN3D [177, 178]	7	1.80×10^{1}	6.83×10^{-13}
EXTTET [185]	7	8.73	7.68
SBRYBND [177, 178]	9	2.73×10^{14}	2.83×10^{3}
GENROSE [177]	15	9.59×10^{1}	1.00
SCOSINEL [180]	16	4.56	-1.19×10^{1}
TRIGSSQS [94]	17	2.47×10^{6}	5.78×10^{1}
GENROSE [177]	18	1.06×10^{2}	1.00
TOINTTRIG [184]	20	-9.09×10^{1}	-4.75×10^{3}
SROSENBR [177, 178]	25	2.90×10^{2}	2.36×10^{1}
TRIGSSQS [94]	26	1.38×10^{7}	1.36×10^{3}
EXTTET [185]	37	5.24×10^{1}	4.61×10^{1}
COSINE [177]	39	3.33×10^{1}	-3.80×10^{1}
PENALTY3 [177]	40	2.72×10^{6}	1.00
TRIGSABS [94]	42	5.56×10^{3}	4.36×10^{1}
TOINTTRIG [184]	43	-9.89×10^{2}	-2.26×10^{4}
SCOSINEL [180]	46	3.72	-2.33×10^{1}
TRIGSSQS [94]	46	2.45×10^{7}	1.13×10^{1}
BROYDN7D [177, 184]	48	1.33×10^{2}	6.01
TRIGSABS [94]	50	8.48×10^{3}	3.32×10^{1}
COSINE [177]	55	4.74×10^{1}	-5.40×10^{1}
TRIGSSQS [94]	61	3.23×10^{7}	3.43×10^{2}
COSINE [177]	63	5.44×10^{1}	-6.19×10^{1}
PENALTY3 [177]	65	1.82×10^{7}	1.55×10^{4}
TRIGSABS [94]	66	2.38×10^{4}	1.11×10^{2}
TRIGSABS [94]	68	1.93×10^{4}	9.33×10^{1}
TOINTGSS [177]	84	7.48×10^{2}	9.65
TOINTGSS [177]	89	7.93×10^{2}	9.67
PENALTY3 [177]	90	6.62×10^{7}	2.90×10^{4}
PENALTY2 [177, 178]	100	9.73×10^{9}	8.93×10^{9}
TOINTGSS [177]	100	8.92×10^{2}	9.71
TOINTGSS [177]	108	9.64×10^{2}	9.73
SROSENBR [177, 178]	115	1.38×10^{3}	3.53
PENALTY2 [177, 178]	118	3.55×10^{11}	2.63×10^{11}
SPARSINE [177]	119	2.95×10^{4}	1.46×10^{6}
SROSENBR [177, 178]	120	1.45×10^{3}	5.79

Table 2-10 (Continued)

SROSENBR [177, 178]	130	1.57×10^{3}	4.29
PENALTY3 [177]	140	3.85×10^{8}	1.47
PENALTY2 [177, 178]	161	1.93×10^{15}	1.04×10^{15}
TQUARTIC [177]	165	8.10×10^{-1}	1.81×10^{3}
SROSENBR [177, 178]	175	2.11×10^{3}	2.31×10^{1}
PENALTY2 [177, 178]	192	9.51×10^{17}	5.52×10^{17}
TQUARTIC [177]	193	8.10×10^{-1}	2.69×10^{3}
ARGLINA [177, 178]	250	1.25×10^{2}	2.50×10^{2}
ARGLINB [177, 178]	250	4.11×10^{16}	1.25×10^{2}
CHNROSNB [177, 180]	250	5.46×10^{3}	1.04×10^{-9}
CHROSEN [94]	250	4.98×10^{3}	4.90×10^{-3}
COSINE [177]	250	2.19×10^{2}	-2.49×10^{2}
CURLY30 [177]	250	-2.91×10^{-4}	-2.50×10^{4}
PENALTY2 [177, 178]	300	2.29×10^{27}	1.78×10^{27}
TOINTTRIG [184]	350	1.04×10^{1}	-1.53×10^{6}
COSINE [177]	220	1.92×10^{2}	-2.19×10^{2}
ROSENBROCK [177, 178]	250	1.01×10^{5}	2.16×10^{2}
GENROSE [177]	320	1.21×10^{3}	2.03×10^{2}
ARGLINA [177, 178]	500	2.50×10^{3}	5.00×10^{2}
ARGLINA [177, 178]	600	3.00×10^{3}	6.00×10^{2}
PENALTY3 [177]	800	4.09×10^{11}	1.11

Figures 2-10 and 2-11 present the Performance Profile and Data Profile of the tested derivative-free optimization methods, where the values of τ are 10^{-1}, 10^{-3}, and 10^{-5}. The method proposed in this section (for simplicity, labeled as "Model (optimality)" in the figures) achieves the best numerical performance on such problems and accuracies.

We can observe the performance differences among the compared methods. For example, in the Performance Profile shown in Figure 2-10, when $\alpha = 1$, our method has the highest values (50%, 52.73%, and 57.27%) for $\tau = 10^{-1}$, $\tau = 10^{-3}$, and $\tau = 10^{-5}$, respectively, which means that it successfully solves the largest number of problems among all the tested methods. The Data Profile in Figure 2-11 also shows that, for all listed accuracies τ, the method proposed in this section solves the highest proportion of problems.

In addition, Table 2-11 shows the corresponding problem-solving ratios, where subscripts $1, 2, \cdots, 10$ represent the ten methods compared in Figures 2-10 and 2-11: "Model (optimality)" (i.e., the method proposed in this section), "Least Frob. Norm", "Powell", "Least H^2 Norm Update", "Conn & Toint", "NEWUOA", "Fminsearch", "Fminunc", "CMA-ES", and "NMSMAX". For example, in the Performance Profile for $\tau = 10^{-5}$,

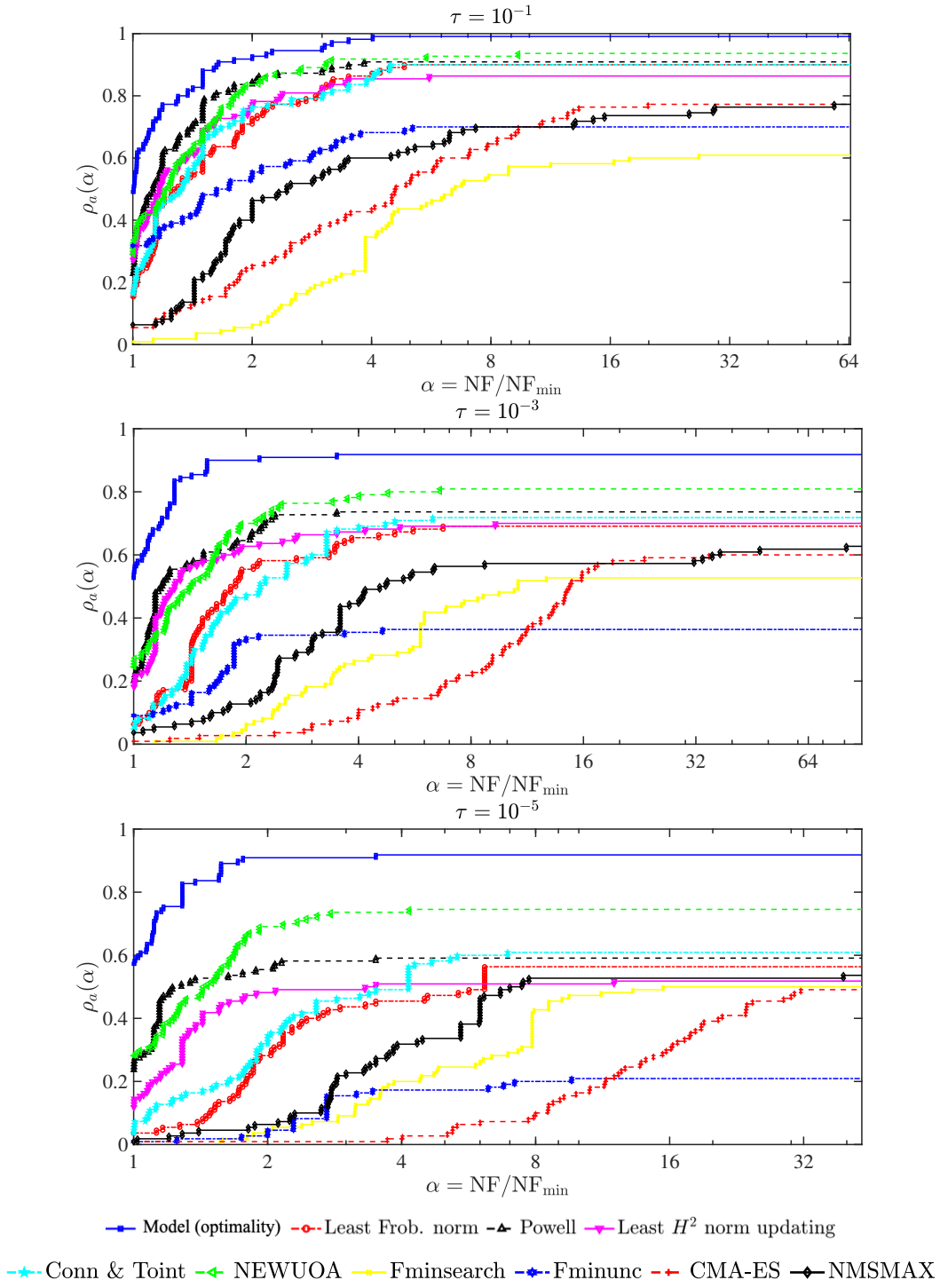


Figure 2-10 Performance Profile of minimizing test problems

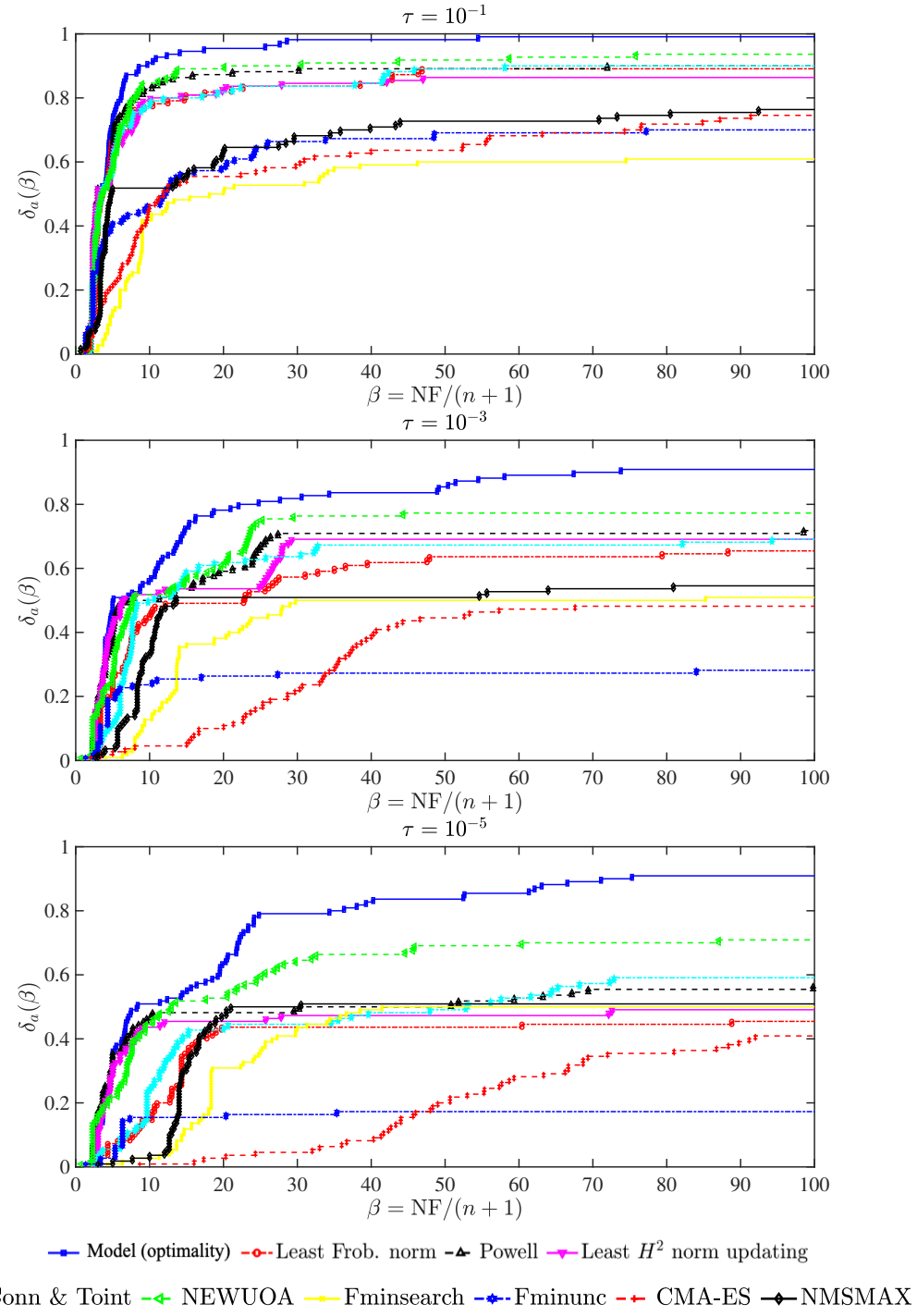


Figure 2-11 Data Profile of minimizing test problems

Table 2-11 The ratio of the solved problems

The ratio of the solved problems for Performance Profiles when $\alpha = 2$					
τ	$\rho_1(2)$	$\rho_2(2)$	$\rho_3(2)$	$\rho_4(2)$	$\rho_5(2)$
10^{-1}	92.73%	72.73%	84.55%	77.27%	76.36%
10^{-3}	90.00%	55.45%	64.55%	62.73%	47.27%
10^{-5}	90.91%	28.18%	55.45%	48.18%	34.55%
τ	$\rho_6(2)$	$\rho_7(2)$	$\rho_8(2)$	$\rho_9(2)$	$\rho_{10}(2)$
10^{-1}	83.64%	6.36%	55.45%	25.45%	46.36%
10^{-3}	70.00%	6.36%	33.64%	2.73%	12.73%
10^{-5}	69.09%	4.55%	4.55%	0.91%	6.36%
The ratio of the solved problems for Data Profiles when $\beta = 30$					
τ	$\delta_1(30)$	$\delta_2(30)$	$\delta_3(30)$	$\delta_4(30)$	$\delta_5(30)$
10^{-1}	98.18%	83.64%	88.18%	84.55%	83.64%
10^{-3}	81.82%	57.27%	70.91%	69.09%	63.64%
10^{-5}	79.09%	43.64%	49.09%	47.27%	44.55%
τ	$\delta_6(30)$	$\delta_7(30)$	$\delta_8(30)$	$\delta_9(30)$	$\delta_{10}(30)$
10^{-1}	90.00%	52.73%	66.36%	59.09%	68.18%
10^{-3}	76.36%	50.00%	27.27%	21.82%	50.91%
10^{-5}	64.55%	43.64%	16.36%	4.55%	50.00%

the method proposed in this section solves 90.91% of the problems when $\alpha = 2$, which is more than any other method. Furthermore, under a maximum of $30(n + 1)$ function evaluations and with accuracy $\tau = 10^{-5}$, the Data Profile indicates that our method solves 79.09% of the problems, approximately 15% more than the second-best algorithm (NEWUOA). The above profiles fully demonstrate the advantages of the model and method proposed in this section.

2.3.5 Summary

In this section, we analyzed and improved the Conn-Toint model using trust-region iterations and proposed a new derivative-free optimization approximation model by incorporating the optimality of the iteration point. This section utilizes trust-region iteration to help construct interpolation models. We identified and exploited more relationships between optimality and interpolation, provided the motivation for the method proposed in this section, analyzed its convexity, and explained how to obtain the coefficients of the proposed quadratic model in an implementation-friendly way. Numerical results demonstrate the advantages of the proposed model and method.

Beyond the classical convergence results for algorithms using general models, new convergence results for methods using the newly proposed model in this section remain to be studied further. It is also valuable to investigate the optimal number of interpolation points for constructing the model proposed in this section. Another potential direction is to explore the corresponding handling in the case of unsuccessful current iterations.

2.4 Sufficient Conditions for Reducing the Distance Between Minimizers of Nonconvex Quadratic Functions in a Trust Region

This section analyzes sufficient conditions under which the distance between minimizers of nonconvex quadratic functions within a trust region decreases after two iterations. We also provide some examples corresponding to the theoretical results.

We know that trust-region methods obtain the next iterate by solving the subproblem

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathfrak{R}^n} \text{Model}(\boldsymbol{x}) \\ &\text{s. t. } \left\|\boldsymbol{x}-\boldsymbol{x}_c\right\|_2 \le \Delta_k \end{aligned}$$

where $\boldsymbol{x}_c \in \mathfrak{R}^n$ is the center of the trust region $\mathcal{B}_{\Delta_k}(\boldsymbol{x}_c) = \{\boldsymbol{z} \in \mathfrak{R}^n, \left\|\boldsymbol{z}-\boldsymbol{x}_c\right\|_2 \le \Delta_k\}$, and $\Delta_k > 0$ is the trust-region radius at iteration k. The function Model is a quadratic model approximating the objective function to be minimized. The notation Model is used here to distinguish the general model function from the functions f and Q used later. Therefore, quadratic models play a crucial role in generating the next iterate. This section considers the distance between the minimizers of two nonconvex quadratic functions within their respective trust regions. This analysis is motivated by

the observation that in trust-region methods using quadratic models, the model's role is essentially to provide a minimizer that approximates that of the true objective function.

We aim to investigate the conditions under which two quadratic models f and Q yield minimizers within their respective trust regions whose distance is reduced after one iteration. This will be presented in Theorems 2.29 and 2.31. These results help guide the iterative correction of quadratic models and inform model selection in both derivative-based and derivative-free trust-region methods [20, 92, 97, 102, 150]. Moreover, we use concrete examples to show the applicability of our results. For instance, we can directly use these conditions to determine whether two different models can lead to a reduced distance between their respective minimizers after iteration.

Please note that the quadratic functions f and Q discussed here refer to the quadratic model functions used in trust-region algorithms. It is especially important to clarify that in this section, f is not the original objective function, even though in some minimization settings it can act as such. In general, we consider both f and Q to be quadratic models used in trust-region subproblems.

In summary, the distance between minimizers of two nonconvex quadratic functions within their respective trust regions can decrease under certain conditions. This section derives sufficient conditions for such behavior.

In what follows, we assume that $\boldsymbol{x}_1 \in \mathfrak{R}^n$ and $\tilde{\boldsymbol{x}}_1 \in \mathfrak{R}^n$ are the minimizers of the nonconvex quadratic functions f and Q, respectively, within trust regions $\mathcal{B}_{\Delta_1}(\boldsymbol{x}_0)$ and $\mathcal{B}_{\tilde{\Delta}_1}(\boldsymbol{x}_0)$. Likewise, $\boldsymbol{x}_2$ and $\tilde{\boldsymbol{x}}_2$ are the minimizers of f and Q within trust regions $\mathcal{B}_{\Delta_2}(\boldsymbol{x}_1)$ and $\mathcal{B}_{\tilde{\Delta}_2}(\tilde{\boldsymbol{x}}_1)$, respectively, where $\boldsymbol{x}_0 \in \mathfrak{R}^n$ is the initial point (or the center of the first trust region), and $\Delta_1, \tilde{\Delta}_1, \Delta_2, \tilde{\Delta}_2 \in \mathfrak{R}^+$ are the corresponding trust-region radii. In other words, there exist real parameters $\omega_1, \tilde{\omega}_1, \omega_2, \tilde{\omega}_2 > 0$ such that

$$\begin{cases} \boldsymbol{x}_1 - \boldsymbol{x}_0 = -\left(\nabla^2 f + \omega_1 \boldsymbol{I}\right)^{-1} \nabla f(\boldsymbol{x}_0), \\ \tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_0 = -\left(\nabla^2 Q + \tilde{\omega}_1 \boldsymbol{I}\right)^{-1} \nabla Q(\boldsymbol{x}_0) \end{cases} \tag{2-45}$$

and

$$\begin{cases} \boldsymbol{x}_2 - \boldsymbol{x}_1 = -\left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \nabla f(\boldsymbol{x}_1), \\ \tilde{\boldsymbol{x}}_2 - \tilde{\boldsymbol{x}}_1 = -\left(\nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I}\right)^{-1} \nabla Q(\tilde{\boldsymbol{x}}_1), \end{cases} \tag{2-46}$$

where $\Delta_1 = \|\boldsymbol{x}_1 - \boldsymbol{x}_0\|_2$, $\tilde{\Delta}_1 = \|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_0\|_2$, $\Delta_2 = \|\boldsymbol{x}_2 - \boldsymbol{x}_1\|_2$, $\tilde{\Delta}_2 = \|\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_1\|_2$, and $\nabla^2 f + \omega_1 \boldsymbol{I} \succeq \boldsymbol{0}, \nabla^2 Q + \tilde{\omega}_1 \boldsymbol{I} \succeq \boldsymbol{0}, \nabla^2 f + \omega_2 \boldsymbol{I} \succeq \boldsymbol{0}, \nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I} \succeq \boldsymbol{0}$.

Assumption 2.27. Assume that f and Q are nonconvex quadratic functions, with $\nabla^2 f + \omega_2 \boldsymbol{I} \succ \boldsymbol{0}, \nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I} \succ \boldsymbol{0}$, and $\tilde{\boldsymbol{x}}_1 \neq \boldsymbol{x}_1$.

Remark 2.10. For simplicity, the same symbols used across different results in this section may denote objects of different dimensions. Also, $\boldsymbol{A} \succ \boldsymbol{0}$ denotes that the matrix $\boldsymbol{A}$ is positive definite.

We now state the question under consideration.

Question. Under Assumption 2.27, for $0 \leq \rho \leq 1$, under what sufficient conditions do the minimizers of the quadratic functions f and Q satisfy

$$\left\|\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_2\right\|_2 \leq \rho \left\|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right\|_2 \ ? \tag{2-47}$$

2.4.1 Distance Analysis of Minimizers of Quadratic Functions within a Trust Region

We now provide results on the distance between minimizers of quadratic functions.

Proposition 2.28. *The difference between minimizers satisfies*

$$\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_2 = \tilde{\omega}_1 \left(\nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I}\right)^{-1} \left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_0\right) - \omega_1 \left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right) + \left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right).$$

Proof. From (2-45) and (2-46), we have

$$\begin{aligned}
\boldsymbol{x}_2 - \boldsymbol{x}_1 &= -\left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \nabla f(\boldsymbol{x}_1) \\
&= -\left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \left(\nabla f(\boldsymbol{x}_0) + \nabla^2 f \cdot \left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right)\right) \\
&= -\left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \left(-\left(\nabla^2 f + \omega_1 \boldsymbol{I}\right)\left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right) + \nabla^2 f \cdot \left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right)\right) \\
&= \omega_1 \left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right)
\end{aligned}$$

and

$$\tilde{\boldsymbol{x}}_2 - \tilde{\boldsymbol{x}}_1 = \tilde{\omega}_1 \left(\nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I}\right)^{-1} \left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_0\right),$$

from which the result follows directly by computation. □

Theorem 2.29 (Sufficient and Necessary Condition in the 1-D Case)**.** *Assume that Assumption 2.27 holds, the dimension $n = 1$, and $\kappa := \frac{\boldsymbol{x}_1 - \boldsymbol{x}_0}{\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1} \in \mathfrak{R}$, then (2-47) holds for $0 \leq \rho \leq 1$ if and only if*

$$\begin{cases} \left(\nabla^2 Q + \tilde{\omega}_2\right) \omega_1 > \left(\nabla^2 f + \omega_2\right) \tilde{\omega}_1, \\ \kappa_1 \leq \kappa \leq \kappa_2, \end{cases} \tag{2-48}$$

or

$$\begin{cases} \left(\nabla^2 Q + \tilde{\omega}_2\right) \omega_1 < \left(\nabla^2 f + \omega_2\right) \tilde{\omega}_1, \\ \kappa_2 \leq \kappa \leq \kappa_1, \end{cases} \tag{2-49}$$

where

$$\begin{cases} \kappa_1 = \dfrac{\left(\nabla^2 f + \omega_2\right)\left[(-\rho + 1)\left(\nabla^2 Q + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f + \omega_2\right)\tilde{\omega}_1}, \\ \kappa_2 = \dfrac{\left(\nabla^2 f + \omega_2\right)\left[(\rho + 1)\left(\nabla^2 Q + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f + \omega_2\right)\tilde{\omega}_1}. \end{cases}$$

Proof. The condition that either (2-48) or (2-49) holds is equivalent to

$$\left|1 + \frac{\tilde{\omega}_1(1+\kappa)}{\nabla^2 Q + \tilde{\omega}_2} - \frac{\omega_1 \kappa}{\nabla^2 f + \omega_2}\right| \left\|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right\|_2 \leq \rho \left\|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right\|_2,$$

which follows directly from basic computation. □

Corollary 2.30. *Assume that Assumption 2.27 holds, the problem dimension is* $n = 1$*, and* $\kappa := \frac{x_1 - x_0}{\tilde{x}_1 - x_1} \in \mathfrak{R}$*. If* $-1 < \kappa < 0$*, i.e.,* $\tilde{x}_1 \leq x_0 < x_1$ *or* $x_1 < x_0 \leq \tilde{x}_1$*, then there exists no* $0 < \rho < 1$ *such that (2-47) holds.*

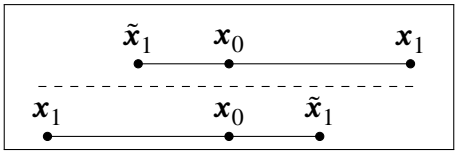


Figure 2-12 Distribution of $x_{k-1}, x_k, \tilde{x}_k$ corresponding to Corollary 2.30

Proof. Given $\omega_1 > 0, \tilde{\omega}_1 > 0, G > 0, H > 0, \rho > 0, -1 \leq \kappa \leq 0$, we have

$$-\rho \leq 1 + \frac{\tilde{\omega}_1(1+\kappa)}{G} - \frac{\omega_1 \kappa}{H} \leq \rho$$

which is equivalent to

$$\rho \geq \frac{GH - G\kappa\omega_1 + H\kappa\tilde{\omega}_1 + H\tilde{\omega}_1}{GH} \geq 1. \tag{2-50}$$

Thus, the conclusion follows from (2-50). □

Remark 2.11. Figure 2-12 illustrates the case described in Corollary 2.30.

Theorem 2.31 (Sufficient Condition for General n-Dimensional Case with Diagonal Hessian)**.** *Assume that Assumption 2.27 holds, and define* $\boldsymbol{\kappa} := \text{Diag}\left\{\kappa^{[1]}, \kappa^{[2]}, \cdots, \kappa^{[n]}\right\} \in \mathfrak{R}^{n\times n}$ *such that* $\boldsymbol{\kappa}\left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right) = \boldsymbol{x}_1 - \boldsymbol{x}_0$*, and assume* $\nabla^2 f$ *and* $\nabla^2 Q$ *are diagonal matrices. If for any* $i \in \{1, 2, \cdots, n\}$*, we have*

$$\begin{cases} \left(\nabla^2 Q^{[i]} + \tilde{\omega}_2\right)\omega_1 > \left(\nabla^2 f^{[i]} + \omega_2\right)\tilde{\omega}_1, \\ \kappa_1^{[i]} \leq \kappa^{[i]} \leq \kappa_2^{[i]} \end{cases}$$

or

$$\begin{cases} \left(\nabla^2 Q^{[i]} + \tilde{\omega}_2\right)\omega_1 < \left(\nabla^2 f^{[i]} + \omega_2\right)\tilde{\omega}_1, \\ \kappa_2^{[i]} \leq \kappa^{[i]} \leq \kappa_1^{[i]} \end{cases}$$

then (2-47) holds, where the superscript $[i]$ *denotes the* i*th diagonal element of the matrices* $\nabla^2 f$ *or* $\nabla^2 Q$*, or the* i*th entry of vectors* κ_1 *and* κ_2*, respectively.*

Proof. We have

$$\begin{aligned} &\left\|\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_2\right\|_2 \\ =&\left\|\left(\boldsymbol{I} + \tilde{\omega}_1\left(\nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I}\right)^{-1}(\boldsymbol{I} + \boldsymbol{\kappa}) - \omega_1\left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1}\boldsymbol{\kappa}\right)\left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right)\right\|_2 \\ \leq&\left\|\rho\left(\tilde{\boldsymbol{x}}_1^{[1]} - \boldsymbol{x}_1^{[1]}, \cdots, \tilde{\boldsymbol{x}}_1^{[n]} - \boldsymbol{x}_1^{[n]}\right)^\top\right\|_2 = \rho\|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\|_2, \end{aligned}$$

where the superscript $[i]$ denotes the i-th component of the corresponding vector. This holds because

$$\left|1+\tilde{\omega}_1\frac{1+\kappa^{[i]}}{\nabla^2 Q^{[i]}+\tilde{\omega}_2}-\omega_1\frac{\kappa^{[i]}}{\nabla^2 f^{[i]}+\omega_2}\right|\leq\rho,\ \forall\, i=1,\cdots,n.$$

Based on the above, the conclusion is proved. □

Corollary 2.32. *Suppose Assumption 2.27 holds and that* $\boldsymbol{\kappa} := \mathrm{Diag}\left\{\kappa^{[1]},\kappa^{[2]},\cdots,\kappa^{[n]}\right\} \in \mathfrak{R}^{n\times n}$ *satisfies* $\boldsymbol{\kappa}\left(\tilde{\boldsymbol{x}}_1-\boldsymbol{x}_1\right)=\boldsymbol{x}_1-\boldsymbol{x}_0$. *If for all i,* $-1<\kappa^{[i]}<0$, *i.e.,* $\tilde{\boldsymbol{x}}_1^{[i]}\leq\boldsymbol{x}_0^{[i]}<\boldsymbol{x}_1^{[i]}$ *or* $\boldsymbol{x}_1^{[i]}<\boldsymbol{x}_0^{[i]}\leq\tilde{\boldsymbol{x}}_1^{[i]}$, *then there does not exist* $0<\rho<1$ *such that (2-47) holds.*

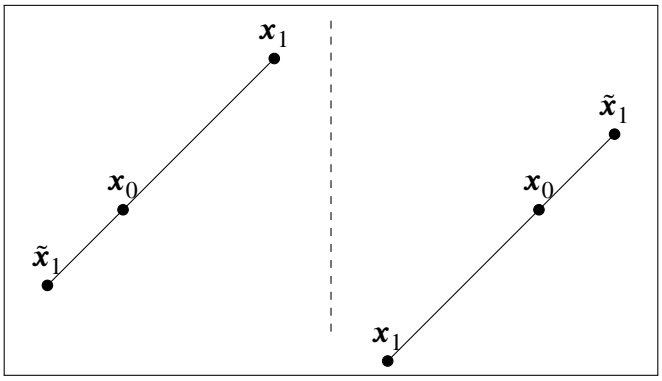


Figure 2-13 Distribution of $\boldsymbol{x}_{k-1},\boldsymbol{x}_k,\tilde{\boldsymbol{x}}_k$ corresponding to Corollary 2.32

Proof. The conclusion follows directly by applying Corollary 2.30 componentwise. □

Remark 2.12. Figure 2-13 illustrates the scenario described in Corollary 2.32.

2.4.2 Example

We present the following example to illustrate the above results.

Example 2.7. In this example, we illustrate the case where the dimension is $n=2$, the quadratic models have diagonal Hessian matrices, and $\boldsymbol{\kappa}$ has different nonzero components.

We consider

$$\begin{cases} f(\boldsymbol{x})=-\frac{1}{2}\boldsymbol{x}^\top\begin{pmatrix}1&0\\0&2\end{pmatrix}\boldsymbol{x}+\left(\frac{1}{7},\frac{5}{3}\right)^\top\boldsymbol{x},\\ Q(\boldsymbol{x})=-\frac{1}{2}\boldsymbol{x}^\top\begin{pmatrix}1&0\\0&1\end{pmatrix}\boldsymbol{x}.\end{cases}$$

Additionally, $\boldsymbol{x}_0 = (1,1)^\top,\ \omega_1 = 3,\ \tilde{\omega}_1 = 3,\ \omega_2 = 4,\ \tilde{\omega}_2 = 5,\ \rho = \frac{1}{2}$. We compute

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 - \left(\nabla^2 f(\boldsymbol{x}_0) + \omega_1 \boldsymbol{I}\right)^{-1} \nabla f = \begin{pmatrix} \frac{10}{7} \\ \frac{4}{3} \end{pmatrix},$$

$$\tilde{\boldsymbol{x}}_1 = \boldsymbol{x}_0 - \left(\nabla^2 Q(\boldsymbol{x}_0) + \tilde{\omega}_1 \boldsymbol{I}\right)^{-1} \nabla Q = \begin{pmatrix} \frac{3}{2} \\ \frac{3}{2} \end{pmatrix}$$

and

$$\boldsymbol{\kappa} = \begin{pmatrix} \frac{\frac{10}{7}-1}{\frac{3}{2}-\frac{10}{7}} & 0 \\ 0 & \frac{\frac{4}{3}-1}{\frac{3}{2}-\frac{4}{3}} \end{pmatrix} = \begin{pmatrix} 6 & 0 \\ 0 & 2 \end{pmatrix}.$$

Then we have

$$\begin{cases} \left(\nabla^2 Q^{[1]} + \tilde{\omega}_2\right) \omega_1 = 12 > 9 = \left(\nabla^2 f^{[1]} + \omega_2\right) \tilde{\omega}_1, \\ \kappa_1^{[1]} \le \kappa^{[1]} \le \kappa_2^{[1]} \end{cases}$$

and

$$\begin{cases} \left(\nabla^2 Q^{[2]} + \tilde{\omega}_2\right) \omega_1 = 12 > 6 = \left(\nabla^2 f^{[2]} + \omega_2\right) \tilde{\omega}_1, \\ \kappa_1^{[2]} \le \kappa^{[2]} \le \kappa_2^{[2]}, \end{cases}$$

where

$$\begin{cases} \kappa_1^{[1]} = \dfrac{\left(\nabla^2 f^{[1]} + \omega_2\right)\left[(-\rho+1)\left(\nabla^2 Q^{[1]} + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q^{[1]} + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f^{[1]} + \omega_2\right)\tilde{\omega}_1} = 5, \\ \kappa_2^{[1]} = \dfrac{\left(\nabla^2 f^{[1]} + \omega_2\right)\left[(\rho+1)\left(\nabla^2 Q^{[1]} + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q^{[1]} + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f^{[1]} + \omega_2\right)\tilde{\omega}_1} = 9, \end{cases}$$

$$\begin{cases} \kappa_1^{[2]} = \dfrac{\left(\nabla^2 f^{[2]} + \omega_2\right)\left[(-\rho+1)\left(\nabla^2 Q^{[2]} + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q^{[2]} + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f^{[2]} + \omega_2\right)\tilde{\omega}_1} = \dfrac{5}{3}, \\ \kappa_2^{[2]} = \dfrac{\left(\nabla^2 f^{[2]} + \omega_2\right)\left[(\rho+1)\left(\nabla^2 Q^{[2]} + \tilde{\omega}_2\right) + \tilde{\omega}_1\right]}{\left(\nabla^2 Q^{[2]} + \tilde{\omega}_2\right)\omega_1 - \left(\nabla^2 f^{[2]} + \omega_2\right)\tilde{\omega}_1} = 3. \end{cases}$$

Thus, the sufficient condition is satisfied. Furthermore, we have

$$\begin{aligned} &\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_2 \\ &= \tilde{\omega}_1 \left(\nabla^2 Q + \tilde{\omega}_2 \boldsymbol{I}\right)^{-1} \left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_0\right) - \omega_1 \left(\nabla^2 f + \omega_2 \boldsymbol{I}\right)^{-1} \left(\boldsymbol{x}_1 - \boldsymbol{x}_0\right) + \left(\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right) \\ &= \begin{pmatrix} \frac{1}{56} \\ \frac{1}{24} \end{pmatrix}, \end{aligned}$$

and hence

$$\left\|\tilde{\boldsymbol{x}}_2 - \boldsymbol{x}_2\right\|_2 = \frac{\sqrt{\frac{29}{2}}}{84} < \frac{1}{2}\frac{\sqrt{\frac{29}{2}}}{21} = \rho \left\|\tilde{\boldsymbol{x}}_1 - \boldsymbol{x}_1\right\|_2.$$

The following example illustrates numerical observations in the one-dimensional case $n = 1$.

Example 2.8. We attempt to numerically observe the probability that the parameters satisfy the conditions in Theorem 2.29, focusing on the case where $n = 1$. We perform numerical experiments using Mathematica software.

Specifically, we integrate ω_2 and $\tilde{\omega}_2$ over the intervals $[0, q\omega_1]$ and $[0, q\tilde{\omega}_1]$, respectively, where q is a non-negative real parameter. The resulting value is then divided by $q^2\omega_1\tilde{\omega}_1$ to represent the probability, i.e.,

$$\begin{aligned}\mathrm{Prob}(\rho) = &\frac{1}{q^2\omega_1\tilde{\omega}_1}\int_0^{q\omega_1}\int_0^{q\tilde{\omega}_1} \mathrm{Boole}\left[\nabla^2 Q + \tilde{\omega}_2 \geq -\frac{(\kappa+1)\,\tilde{\omega}_1\left(\nabla^2 f + \omega_2\right)}{(\rho+1)\left(\omega_2 + \nabla^2 f\right) - \kappa\omega_1}\right] \\ &\mathrm{Boole}\left[\nabla^2 Q + \tilde{\omega}_2 \leq \frac{(\kappa+1)\,\tilde{\omega}_1\left(\nabla^2 f + \omega_2\right)}{(\rho-1)\left(\omega_2 + \nabla^2 f\right) + \kappa\omega_1}\right] d\tilde{\omega}_2 d\omega_2,\end{aligned}$$

where Boole$(\cdot)$ denotes a Boolean function that outputs 0 or 1.

Note that in this example, we define the constants as follows: $\nabla^2 Q = -1$, $\nabla^2 f = -2$, $\omega_1 = 3$, $\tilde{\omega}_1 = 3$, $\kappa = -2$, and $q = 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3$.

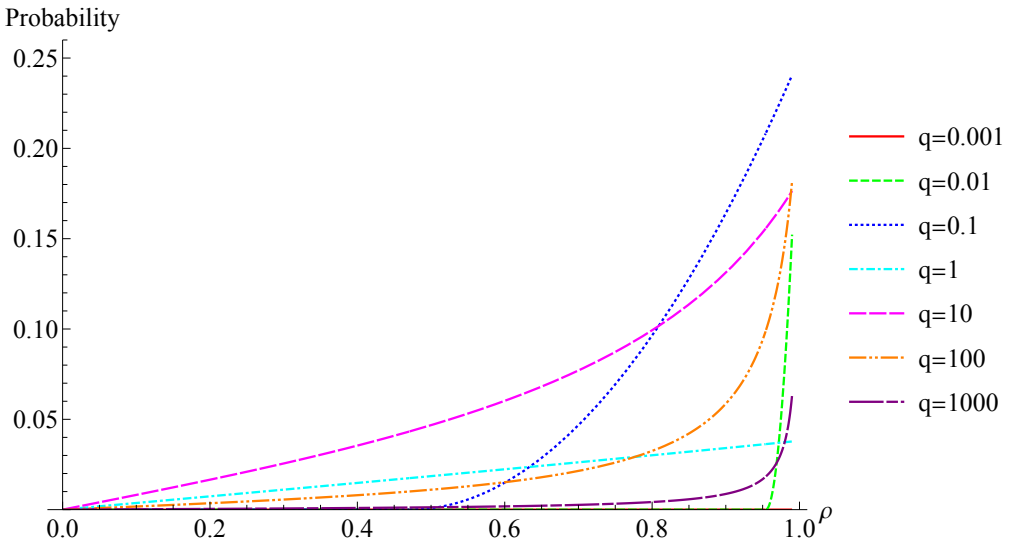


Figure 2-14 Numerical results for Example 2.8

Figure 2-14 shows the numerical results of the function Prob(ρ) as a function of the parameter ρ. Different curves correspond to different values of q. From Figure 2-14, we observe that in this one-dimensional example, the probability of finding ω_2 and $\tilde{\omega}_2$ such that distance reduction is achieved is at most approximately 25%.

2.4.3 Conclusion

This section analyzed sufficient conditions under which the distance between the trust-region minimizers of two nonconvex quadratic functions decreases after one iteration. Note that quadratic functions are commonly used to locally approximate the objective function in numerical optimization algorithms, yet in most nonlinear cases, an exact

model is not attainable. If we have multiple different quadratic surrogate models, the results in this section provide a way to analyze and reduce the distance between the minimizers of such models.

Moreover, the examples in this section show that in certain cases, the distance between the minimizers of two quadratic models may increase after one iteration. This suggests that in trust-region methods, the quadratic model should be updated after each iteration, even if the model is nonconvex and the trial step lies on the boundary of the trust region.

Chapter 3 Derivative-Free Optimization with Transformed Objective Functions and Algorithm Based on least Frobenius Norm Updating Quadratic Models

3.1 Derivative-Free Optimization with Transformed Objective Functions

This chapter focuses on how to solve derivative-free optimization problems with transformed objective functions. The unconstrained derivative-free optimization problem proposed and studied in this chapter has the following general form

$$\min_{\boldsymbol{x}\in\Re^n} \ f(\boldsymbol{x}), \tag{3-1}$$

where the black-box function f can provide the function values of m points in one iteration/query step, and the query scheme is given in Assumption 3.1. In addition, in the same iteration/query step, the m query points share the same transformation of the objective function. It transforms f into $f_k := T_k \circ f$, where the transformation T_k depends only on the current (k-th) step, as defined in Definition 3.2. Note that what we ultimately want to minimize is still the original objective function f.

Assumption 3.1 (Query scheme of derivative-free optimization with transformed objective functions). One query can obtain the function output values corresponding to m points, and this batch of query points can be selected by the optimization algorithm. Figure 3-1 shows the query scheme.

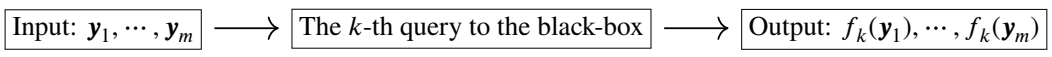


Figure 3-1 Query oracle of derivative-free optimization with transformed objective functions: the k-th query, for the queried points $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m$

It can be observed that the query scheme in Assumption 3.1 has two basic features. One feature is querying a set of points simultaneously, and the other is that the simultaneously queried points share the same transformation. Note that in derivative-free optimization, queries usually cannot be performed in a very short time or at low cost, so we also call it expensive optimization in most scenarios. Such a query scheme usually corresponds to batch interactive queries or simulation mechanisms. Moreover, more application examples will be given at the end of Section 3.1. It should be pointed out that, to the best of our knowledge, although derivative-free optimization with transformed objective functions has a wide range of applications, the related concepts and research have not yet been deeply and concretely explored. Algorithms designed for such problems, especially those based on model functions, are limited. This chapter

aims to propose such transformed problems and provide some preliminary results when solving them using algorithms based on underdetermined quadratic models. We will also answer how least Frobenius norm updating quadratic models and corresponding algorithms are affected by the transformations.

Definition 3.2. *Let T be a transformation from $\mathfrak{R}$ to $\mathfrak{R}$, we denote the transformed function by $T \circ f$, which satisfies: for a given function f and any $\boldsymbol{x} \in \mathfrak{R}^n$, $(T \circ f)(\boldsymbol{x}) = T(f(\boldsymbol{x}))$.*

Each transformation discussed below is a transformation from $\mathfrak{R}$ to $\mathfrak{R}$. The transformed objective function plays an important role in stochastic, noisy, or encrypted derivative-free/black-box optimization[1]. For example, in encrypted black-box optimization, different transformations can be regarded as different encryptions formed by adding different noise according to differential privacy theory [195–200]. Section 3.5.4 will give details of solving a special class of encrypted engineering design optimization problems. Another example is cloud-based distributed optimization problems, which aim to minimize local and cloud-based composite objective functions while protecting the privacy of the corresponding objective functions [201]. In addition, there are also encrypted black-box optimization problems in the field of personal health [202]. Furthermore, Kusner et al. discussed differential privacy Bayesian optimization [203].

In fact, derivative-free optimization with transformed objective functions has various applications, and encrypted black-box optimization is just one instance. For example, problems with regularization functions whose coefficients vary with iterations belong to derivative-free optimization problems with transformed objective functions. Grapiglia, Yuan, and Yuan proposed a derivative-free trust-region algorithm for composite nonsmooth optimization [120]. For discussions on minimizing transformed objective functions, one may review the work of Deng and Ferris [204], which provides discussion on minimizing stochastic objective functions using the UOBYQA algorithm [137].

In addition to characterizing the changes of interpolation models when transformations or perturbations exist in the objective function at each iteration, this chapter also focuses on whether the minimizer of the quadratic model within the trust region will change.

Remark 3.1. In fact, some derivative-free methods do not rely on function values (absolute magnitude), such as Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [131], the Nelder-Mead method [51] (using only comparison queries), and other algo-

[1]The general form of unconstrained encrypted black-box optimization problems can be expressed as $\min_{\boldsymbol{x}\in\mathfrak{R}^n} f(\boldsymbol{x})$, where f is an encrypted black-box function. The query cost of f is expensive, and its output values are encrypted into f_k by adding noise.

rithms that only use Boolean function comparisons [205]. This chapter focuses on developing and improving model-based algorithms, which rely on function values to solve problems with transformed objective functions. We will provide a new perspective to characterize and understand the impact of transformations on the objective function, the model, and the algorithm when using function-value-dependent derivative-free optimization methods.

Considering solving the proposed DFOTO problem using derivative-free trust-region algorithms with least Frobenius norm updating quadratic models, this chapter will make the following contributions. We will modify Powell's updating formula of least Frobenius norm updating quadratic models [93] to match the proposed query scheme for use in model-based trust-region methods when solving transformed problems. We will analyze the least Frobenius norm updating interpolation model when minimizing transformed objective functions. We will propose optimality-preserving transformations and give and prove necessary and sufficient conditions for such transformations. We will discuss positive monotonic transformations. We will give analytical expressions of least Frobenius norm updating quadratic models for affine transformed objective functions, analyze their interpolation errors, and provide further discussion. This chapter will also give preliminary convergence analysis for first-order critical points. The numerical results of this chapter will demonstrate the necessity of using the modified model updating formula and show that our method can efficiently and robustly solve most test problems and a presented practical problem, even when the transformations alter the optimality of the model function. To the best of our knowledge, this is the first work to minimize transformed objective functions using model-based algorithms that require function value information (rather than function value comparisons). Notably, the performance of the corresponding methods is satisfactory when solving order-preserving or optimality-preserving transformation problems.

The structure of this chapter is as follows. In Section 3.2, we present our algorithm and query scheme, which include least Frobenius norm updating quadratic models, the trust-region subproblem, and the related definitions of optimality-preserving transformations. The existence of optimality-preserving transformations, except for translation transformations, is proven. The necessary and sufficient conditions of such transformations are also given in this section. In Section 3.3, we introduce the properties of positive monotonic transformations, especially affine transformations. We find that affine transformations with (non-trivial) positive multiplicative coefficients are not optimality-preserving transformations. When the objective function is affinely transformed, we provide the corresponding transformation of the least Frobenius norm updating quadratic model function. Section 3.4 shows the coefficients of fully linear model interpolation errors when solving problems with affine transformations. Section 3.4

also analyzes the convergence when the algorithm minimizes the transformed objective function under provable guarantees of the model-based derivative-free framework. Section 3.5 presents numerical results of minimizing test examples, as well as performance profiles and sensitivity profiles of solving a set of test problems with random affine transformed objective functions. Such numerical results support our theoretical analysis, and the results of solving the optimal design problem of traveling-wave tube encryption engineering also demonstrate the practical advantages of our method. At the end of this chapter, we propose the "moving target" derivative-free optimization problem.

3.2 Algorithm, Query Scheme and Optimality-Preserving Transformation

Here we introduce our algorithm and query scheme and present in detail the least Frobenius norm updating quadratic models used in derivative-free optimization. In addition, we introduce the trust-region subproblem in model-based derivative-free algorithms. We also give some basic concepts, including optimality-preserving transformations.

3.2.1 Model-Based Trust-Region Algorithms and Query Schemes

We give some details of the model-based trust-region algorithms and query schemes that will be used to solve problem (3-1). We also explain the reasons for choosing such a framework, interpolation model functions, and query schemes.

The basic framework of model-based derivative-free trust-region algorithms (for the transformed objective function) is shown in Algorithm 5. For simplicity, some details are omitted in the algorithm framework. One can find the algorithmic framework of model-based derivative-free trust-region algorithms for minimizing the original objective function in the monograph by Conn, Scheinberg, and Vicente [20]. As shown in Assumption 3.1 and Table 3-1, the queries in Algorithm 5 are performed on a batch of points sharing the same transformation. The functions $f_1, \cdots, f_k, \cdots$ represent the transformed objective functions corresponding to transformations $T_1, \cdots, T_k, \cdots$ at steps $1, \cdots, k, \cdots$, respectively. When solving the trust-region subproblem, the center of the trust region is usually set to $\boldsymbol{x}_{\text{opt}}$, the point with the least function value among the current iteration interpolation points. To provide a simplified algorithmic framework, we use $\boldsymbol{x}_k$, and in (3-10) we write it as $\boldsymbol{x}_{\text{opt}}$. When updating the k-th interpolation set, we usually discard the worst interpolation point $\boldsymbol{y}_t$ at this step and replace it with $\boldsymbol{y}_{\text{new}}(= \boldsymbol{x}_{k-1} + \boldsymbol{d}_{k-1})$, which is the newly added interpolation point. Algorithm 5 adopts the traditional Λ-poisedness in model-based trust-region derivative-free methods, and the related verification refers to the same steps in the traditional algorithm framework of model-based trust-region derivative-free methods [20].

Algorithm 5 Framework of model-based derivative-free trust-region algorithm for minimizing transformed objective functions

Given an initial point $\boldsymbol{x}_{\text{int}}$ and an initial interpolation point set $\mathcal{X}_1$ satisfying $\boldsymbol{x}_{\text{int}} \in \mathcal{X}_1$, and

$$f_1(\boldsymbol{x}_{\text{int}}) = \min_{\boldsymbol{y}\in\mathcal{X}_1} f_1(\boldsymbol{y}).$$

Choose an initial trust-region radius Δ_1. Let $k = 1$.

Step 1. (Construct interpolation model)

Construct a quadratic model Q_k satisfying the interpolation condition $Q_k(\boldsymbol{y}) = f_k(\boldsymbol{y}),\ \boldsymbol{y} \in \mathcal{X}_k$.

Step 2. (Trust-region iteration)

Solve the trust-region subproblem

$$\min_{\boldsymbol{d}\in\Re^n} Q_k(\boldsymbol{x}_k + \boldsymbol{d})$$
$$\text{s. t. } \|\boldsymbol{d}\|_2 \le \Delta_k$$

and obtain its solution $\boldsymbol{d}_k$.

If $\boldsymbol{x}_k + \boldsymbol{d}_k$ is accepted, e.g., $f_{k+1}(\boldsymbol{x}_k + \boldsymbol{d}_k) < f_{k+1}(\boldsymbol{x}_k)$, then set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}_k$; otherwise, set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$.

Step 3. (Manage interpolation set)

Check whether the interpolation point set is poised. If necessary, perform model improvement steps to enhance the poisedness of the interpolation set. Update the interpolation point set to $\mathcal{X}_{k+1}$ so that it contains $\boldsymbol{x}_{k+1}$.

Step 4. (Update)

Update the trust-region radius to obtain Δ_{k+1} according to the performance of $\boldsymbol{d}_k$ and the poisedness of the interpolation set. Let $k = k + 1$. Go to **Step 1**.

We introduce here the query procedure for solving problems with transformed objective functions, which will be used in the subsequent discussion. We synchronously query the first m interpolation points. Once a new iterate is obtained, update the interpolation set according to the process shown in Table 3-1[2], and obtain queries of function values at the points in the new interpolation set. Note that each query set contains m points. The query set can be updated in different ways, for example, $\mathcal{X}_k := \mathcal{X}_{k-1}\backslash\{\arg\max_{\boldsymbol{y}\in\mathcal{X}_{k-1}} \|\boldsymbol{y} - \boldsymbol{x}_k\|_2\} \cup \{\boldsymbol{x}_k\}$ or $\mathcal{X}_k := \mathcal{X}_{k-1}\backslash\{\arg\max_{\boldsymbol{y}\in\mathcal{X}_{k-1}} f_{k-1}(\boldsymbol{y})\} \cup \{\boldsymbol{x}_k\}$.

Note that the interpolation model function in Algorithm 5 is the least Frobenius norm updating quadratic model. Next, we provide the reasons for choosing the query scheme

[2]We refer to the interpolation point set as the "interpolation set."

Table 3-1 Query and evaluation in algorithms for solving problems with transformed objective functions

Step	Set of queried points	Set of function-value queries
1	$\mathcal{X}_1$	$\{f_1(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}_1\}$
2	$\mathcal{X}_2$	$\{f_2(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}_2\}$
$\vdots$	$\vdots$	$\vdots$
k	$\mathcal{X}_k$	$\{f_k(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}_k\}$
$\vdots$	$\vdots$	$\vdots$

shown in Table 3-1 when using Algorithm 5. First, as can be found in Section 3.2.2, when we use Algorithm 5 based on the least Frobenius norm updating quadratic model to solve derivative-free optimization problems with transformed objective functions, we only need to change the vector on the right-hand side of the interpolation equations (see the details in (3-6)), in which case handling the transformations is very simple. Second, although the transformations of the function-value outputs change across different iterations, we still have reason to trust the points that were queried in the previous iteration and rely on them to find the next iterate. Otherwise, if the algorithm completely re-searches all query points, our approach will exhibit discontinuities in the iterations. In practice, our theoretical analysis and numerical results support above as well.

3.2.2 Least Frobenius Norm Updating Quadratic Models for Transformed Objective Functions

In fact, an important feature of the class of transformed problems we discuss is that the corresponding objective function changes with the iteration. Here, "changes with the iteration" means that the transformation of the objective function depends on the iteration/query step. Therefore, for handling derivative-free optimization with transformed objective functions, we will present the corresponding least Frobenius norm updating quadratic model function. The new updating formula should remain valid when the transformed objective function f_k changes with the iteration index k. Note that the model analyzed in this chapter is the least Frobenius norm updating quadratic model, rather than the least Frobenius norm quadratic model [156, 160, 189], for which different and simpler results will appear in Section 3.3. The efficient and robust numerical performance motivates us to explore more details of the class of models of interest in the presence of transformations.

For simplicity, we assume that the poised interpolation set at the k-th iteration is $\mathcal{X}_k = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\}$. A quadratic model Q_k of the transformed function f_k is obtained

by solving the subproblem

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \ \left\|\nabla^2 Q - \nabla^2 Q_{k-1}\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}) = f_k(\boldsymbol{y}),\ \forall\ \boldsymbol{y}\in\mathcal{X}_k \end{aligned} \tag{3-2}$$

and we define $D_k(\boldsymbol{x}) = Q_k(\boldsymbol{x}) - Q_{k-1}(\boldsymbol{x})$. Then, according to (3-2), we can obtain $D_k(\boldsymbol{x})$ by solving the subproblem

$$\begin{aligned} &\min_{D\in\mathcal{Q}} \ \left\|\nabla^2 D\right\|_F^2 \\ &\text{s. t. } \begin{cases} D(\boldsymbol{y}_i) = f_k(\boldsymbol{y}_i) - f_{k-1}(\boldsymbol{y}_i),\ i = 1,\cdots,t-1,t+1,\cdots,m, \\ D(\boldsymbol{y}_{\text{new}}) = f_k(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}), \end{cases} \end{aligned} \tag{3-3}$$

because, according to the framework of model-based derivative-free trust-region algorithms, the old $\boldsymbol{y}_t$ is discarded and replaced by $\boldsymbol{y}_{\text{new}}$ in the current (k-th) iteration [20]. The derivation from (3-2) to (3-3) is direct; in fact, it follows by replacing the function $Q(\boldsymbol{x}) - Q_{k-1}(\boldsymbol{x})$ with $D(\boldsymbol{x})$.

Let $\lambda_j, j = 1,2,\cdots,m$, be the Lagrange multipliers in the KKT conditions of optimization problem (3-3). As pointed out by Powell [93], they satisfy

$$\begin{cases} \sum_{j=1}^{m} \lambda_j = 0, \\ \sum_{j=1}^{m} \lambda_j \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right) = \boldsymbol{0}_n, \\ \nabla^2 D_k = \sum_{j=1}^{m} \lambda_j \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)^{\top}, \end{cases} \tag{3-4}$$

where $\boldsymbol{0}_n \in \mathfrak{R}^n$, and $\boldsymbol{x}_0$ is a base point used to reduce numerical error, which is set to the initial input point at the beginning. The quadratic function $D_k(\boldsymbol{x})$ can be written as

$$D_k(\boldsymbol{x}) = c + (\boldsymbol{x} - \boldsymbol{x}_0)^{\top}\boldsymbol{g} + \frac{1}{2}\sum_{j=1}^{m} \lambda_j \left((\boldsymbol{x} - \boldsymbol{x}_0)^{\top}(\boldsymbol{y}_j - \boldsymbol{x}_0)\right)^2,\ \boldsymbol{x}\in\mathfrak{R}^n. \tag{3-5}$$

Once the parameters $\boldsymbol{\lambda}^{\top} = (\lambda_1,\cdots,\lambda_m)^{\top} \in \mathfrak{R}^m$, $c\in\mathfrak{R}$, and $\boldsymbol{g}\in\mathfrak{R}^n$ are determined, we can determine the unique function $D_k(\boldsymbol{x})$, thereby obtaining the new quadratic model function $Q_k(\boldsymbol{x})$. It is easy to see that the coefficients of $D(\boldsymbol{x})$ are given by the solution of the linear system

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{X} \\ \boldsymbol{X}^{\top} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ c \\ \boldsymbol{g} \end{pmatrix} = \begin{pmatrix} \boldsymbol{r} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where the matrix $\mathbf{0} \in \Re^{(n+1)\times(n+1)}$ is the zero matrix. The elements of the matrices $\boldsymbol{A} \in \Re^{m\times m}$ and $\boldsymbol{X} \in \Re^{m\times(n+1)}$ are, respectively,

$$\boldsymbol{A}_{ij} = \frac{1}{2}\left(\left(\boldsymbol{y}_i - \boldsymbol{x}_0\right)^{\top}\left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right)^2$$

and

$$\boldsymbol{X} = \begin{pmatrix} 1 & \cdots & 1 \\ \boldsymbol{y}_1 - \boldsymbol{x}_0 & \cdots & \boldsymbol{y}_m - \boldsymbol{x}_0 \end{pmatrix}^{\top},$$

where $1 \le i, j \le m$. The vector $\boldsymbol{r} \in \Re^m$ has the form

$$\boldsymbol{r} = \begin{pmatrix} f_k(\boldsymbol{y}_1) - f_{k-1}(\boldsymbol{y}_1) \\ \vdots \\ f_k(\boldsymbol{y}_{t-1}) - f_{k-1}(\boldsymbol{y}_{t-1}) \\ f_k(\boldsymbol{y}_{\text{new}}) - Q_{k-1}(\boldsymbol{y}_{\text{new}}) \\ f_k(\boldsymbol{y}_{t+1}) - f_{k-1}(\boldsymbol{y}_{t+1}) \\ \vdots \\ f_k(\boldsymbol{y}_m) - f_{k-1}(\boldsymbol{y}_m) \end{pmatrix}. \tag{3-6}$$

Remark 3.2. The vector $\boldsymbol{r}$ is the main difference between the updating formulas for minimizing transformed objective functions and for minimizing objective functions without transformations. This form is very natural and is crucial for obtaining the least Frobenius norm updating quadratic model for transformed objective functions.

We continue to denote the KKT matrix by $\boldsymbol{W}$ and write

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{X} \\ \boldsymbol{X}^{\top} & \mathbf{0} \end{pmatrix}.$$

If $\boldsymbol{W}$ is invertible, then $\boldsymbol{\lambda}, c, \boldsymbol{g}$ can be obtained via

$$\begin{pmatrix} \boldsymbol{\lambda} \\ c \\ \boldsymbol{g} \end{pmatrix} = \boldsymbol{V}\begin{pmatrix} \boldsymbol{r} \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{3-7}$$

where $\boldsymbol{V} = \boldsymbol{W}^{-1}$. The invertibility of $\boldsymbol{W}$ depends on the locations of the interpolation points and is related to the poisedness of the set, which was discussed in depth by Powell [174]. The initial interpolation points guarantee the invertibility of the initial $\boldsymbol{W}$, while the invertibility of the iterative $\boldsymbol{W}$ and the numerical accuracy of formula (3-7) are ensured iteratively by selecting suitable interpolation points during the model-improvement steps. This part follows the same discussion and methodology as in Powell's work [94, 174]. In the discussion below, we assume that the matrix $\boldsymbol{W}$ is invertible.

Remark 3.3. If we obtain the k-th model function by solving

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \left\|\nabla^2 Q - \nabla^2 Q_{k-1}\right\|_F^2 + \sigma \left\|\nabla Q - \nabla Q_{k-1}\right\|_2^2 \\ &\text{s. t. } Q(\boldsymbol{y}) = f_k(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}_k \end{aligned} \tag{3-8}$$

with weight coefficient $\sigma \geq 0$, then the above results and analysis remain valid, with the only difference being that $\boldsymbol{W}$ becomes

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{A} & & \boldsymbol{X} \\ \boldsymbol{X}^\top & 0 & \boldsymbol{0}_n^\top \\ & \boldsymbol{0}_n & -\frac{\sigma}{2}\boldsymbol{I} \end{pmatrix}, \tag{3-9}$$

where $\boldsymbol{I} \in \Re^{n\times n}$ is the identity matrix. In fact, the conclusions of this chapter also apply to other similar least norm updating quadratic models under different norms, including the least H^2 norm updating quadratic model given in Chapter 2 and models that take into account the optimality of the model function and the properties of the previous trust-region iteration.

We may use the updating formulas for the matrix $\boldsymbol{V}$ given by Powell [174]. Ultimately, we obtain $D_k(\boldsymbol{x})$ of the form (3-5), whose parameters $\boldsymbol{\lambda}$, c, and $\boldsymbol{g}$ are given by (3-7), and $\boldsymbol{r}$ is given by (3-6). We then obtain the k-th model $Q_k = Q_{k-1} + D_k$.

3.2.3 Trust-Region Subproblem

Model-based derivative-free trust-region algorithms compute a trial step by solving the trust-region subproblem of the current quadratic model function, namely

$$\begin{aligned} &\min_{\boldsymbol{d}\in\Re^n} Q_k(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \leq \Delta_k. \end{aligned} \tag{3-10}$$

In (3-10), $\boldsymbol{x}_{\text{opt}}$ denotes the interpolation point in the k-th interpolation set $\mathcal{X}_k$ with the optimal function output value.

The framework details of such algorithms can be found in the monograph by Conn, Scheinberg, and Vicente [20]. One termination condition of the subroutine that solves the quadratic trust-region subproblem in model-based derivative-free trust-region algorithms is $\|\boldsymbol{d}_k\|_2 < \hat{\rho}_k$, where $\boldsymbol{d}_k$ is the solution to the trust-region subproblem (3-10), and the parameter $\hat{\rho}_k$ is the lower bound of the trust-region radius, which is used to keep sufficient distance between interpolation points; the details are omitted here.

It should be noted that, for derivative-free optimization problems without transformations, the objective function itself does not change; only the trust region changes as the iterations increase. However, in derivative-free optimization problems with transformations, as the iterations proceed, both f_k and the trust region change, the quadratic

model Q_k is continually updated to approximate f_k, and thus the solution $\boldsymbol{d}_k$ of the subproblem may change according to its definition. In this case, the condition $\|\boldsymbol{d}_k\|_2 < \hat{\rho}_k$ may not be satisfied at all, which would affect the termination of the algorithm. If the termination condition is not met, the algorithmic iteration cannot leave the loop of solving the trust-region subproblem. The number of iterations may grow, implying a high cost of function-value queries. Moreover, the model-improvement steps of the method can hardly be invoked, and consequently the algorithm cannot effectively reduce the interpolation error of the model.

3.2.4 Optimality-Preserving Transformations

In this part, we present the analysis and discussion of optimality-preserving transformations.

Given a black-box function $f : \mathfrak{R}^n \to \mathfrak{R}$, if a quadratic function Q satisfies $Q(\boldsymbol{x}) = f(\boldsymbol{x}), \forall\, \boldsymbol{x} \in \mathcal{X}$, then it is called a quadratic interpolation model of f on the interpolation set $\mathcal{X} \subset \mathfrak{R}^n$. We should note that each transformation discussed below is a transformation from $\mathfrak{R}$ to $\mathfrak{R}$. To introduce more details, we first give the following definition.

Definition 3.3 (least Frobenius norm updating quadratic model of h based on Q_α on $\mathcal{X}$)**.** *Given a function $h : \mathfrak{R}^n \to \mathfrak{R}$, a quadratic function Q_α, and a poised set $\mathcal{X} \subset \mathfrak{R}^n$, where $n+1 \le |\mathcal{X}| < \frac{1}{2}(n+1)(n+2)$, we call a quadratic model function the least Frobenius norm updating quadratic model of h based on Q_α on $\mathcal{X}$ if it is the solution of*

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \ \left\|\nabla^2 Q - \nabla^2 Q_\alpha\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}) = h(\boldsymbol{y}),\ \forall\, \boldsymbol{y} \in \mathcal{X} \end{aligned} \tag{3-11}$$

We denote the above model by the mapping $\mathcal{M}^{\mathcal{X}}_{Q_\alpha}$, i.e., we denote the solution of (3-11) by $\mathcal{M}^{\mathcal{X}}_{Q_\alpha}(h)$.

Definition 3.4 (Subproblem of Q with trust-region radius Δ)**.** *Given a point*[3] *$\boldsymbol{x}_{\text{opt}} \in \mathfrak{R}^n$, a quadratic function Q, and $\Delta \in \mathfrak{R}$, we call the problem*

$$\begin{aligned} &\min_{\boldsymbol{d}\in\mathfrak{R}^n} \ Q(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}) \\ &\text{s. t. } \|\boldsymbol{d}\|_2 \le \Delta \end{aligned}$$

the subproblem of Q with trust-region radius Δ and center $\boldsymbol{x}_{\text{opt}}$. Note that, if it is unnecessary to emphasize the center, we will omit the word "center" in the corresponding statements.

We now give the definition of optimality-preserving transformations for models.

[3]The point $\boldsymbol{x}_{\text{opt}}$ is usually set to be the interpolation point in the interpolation set with the optimal function value.

Definition 3.5 (Optimality-preserving transformation for models)**.** *Assume a poised set* $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$*, and let* Q_α *be a quadratic function. A transformation* T *is called an optimality-preserving transformation with trust-region radius* Δ *if the solution of the subproblem of the least Frobenius norm updating quadratic model of* f *based on* Q_α *on* $\mathcal{X}$ *is the same as the solution of the subproblem of the least Frobenius norm updating quadratic model of* $T \circ f$ *based on* Q_α *on* $\mathcal{X}$*. That is, given a point* $\boldsymbol{x}_{\text{opt}} \in \mathfrak{R}^n$*, if*

$$\arg\min_{\|\boldsymbol{d}\|_2 \le \Delta} Q_{\text{orig}}(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}) = \arg\min_{\|\boldsymbol{d}\|_2 \le \Delta} Q_{\text{trans}}(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}),$$

where

$$Q_{\text{orig}} := \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(f),$$
$$Q_{\text{trans}} := \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(T \circ f),$$

then the transformation T *is called an optimality-preserving transformation with trust-region radius* Δ*.*

Assumption 3.6. Given a function $f : \mathfrak{R}^n \to \mathfrak{R}$, $\boldsymbol{x}_{\text{opt}} \in \mathfrak{R}^n$, a quadratic function Q_α, and a poised interpolation set $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$, where $n + 1 \le |\mathcal{X}| = m < \frac{1}{2}(n+1)(n+2)$, we assume that $\boldsymbol{d}^* \in \mathfrak{R}^n$ is the solution of the subproblem with trust-region radius Δ for the least Frobenius norm updating quadratic model of the function f based on Q_α on $\mathcal{X}$, where $\|\boldsymbol{d}^*\|_2 < \Delta$, and we also assume that this model is strictly convex.

We attempt to present some theoretical results, including necessary and sufficient conditions for optimality-preserving transformations for models.

Theorem 3.7. *Assume that Assumption 3.6 holds. Then a transformation* T *is an optimality-preserving transformation for models if and only if* $(T(f(\boldsymbol{y}_1)), \cdots, T(f(\boldsymbol{y}_m)))^\top$ *is a solution to the linear system*

$$\begin{aligned}
&\sum_{j=1}^{m} \left(\left(\boldsymbol{y}_j - \boldsymbol{x}_{\text{opt}}\right) \left(\boldsymbol{y}_j - \boldsymbol{x}_{\text{opt}}\right)^\top \boldsymbol{d}^* \right) \boldsymbol{V}_j \begin{pmatrix} T(f(\boldsymbol{y}_1)) - Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ T(f(\boldsymbol{y}_m)) - Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \nabla^2 Q_\alpha \boldsymbol{d}^* \\
&= -\begin{pmatrix} \boldsymbol{V}_{m+2} \\ \vdots \\ \boldsymbol{V}_{m+n+1} \end{pmatrix} \begin{pmatrix} T(f(\boldsymbol{y}_1)) - Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ T(f(\boldsymbol{y}_m)) - Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix} - \nabla Q_\alpha(\boldsymbol{x}_{\text{opt}}),
\end{aligned} \tag{3-12}$$

where $\boldsymbol{V}$ *is the inverse of the KKT matrix, and* $\boldsymbol{V}_j$ *denotes the* j*-th row of* $\boldsymbol{V}$*.*

Proof. Assume that the least Frobenius norm updating quadratic model of $T \circ f$ based on Q_α on $\mathcal{X}$ is

$$Q_u(\boldsymbol{x}) = Q_\alpha(\boldsymbol{x}) + c_u + \left(\boldsymbol{x} - \boldsymbol{x}_{\text{opt}}\right)^\top \boldsymbol{g}_u + \frac{1}{2}\sum_{j=1}^{m}(\boldsymbol{\lambda}_u)_j \left(\left(\boldsymbol{x} - \boldsymbol{x}_{\text{opt}}\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_{\text{opt}}\right)\right)^2.$$

We obtain

$$\begin{pmatrix} \boldsymbol{\lambda}_u \\ c_u \\ \boldsymbol{g}_u \end{pmatrix} = \boldsymbol{V} \begin{pmatrix} T(f(\boldsymbol{y}_1)) - Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ T(f(\boldsymbol{y}_m)) - Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

and, for simplicity, the base point $\boldsymbol{x}_0$ appearing in the KKT matrix is set to $\boldsymbol{x}_{\text{opt}}$. We know that $\boldsymbol{d}^*$ is feasible and satisfies $\nabla^2 Q_u \boldsymbol{d}^* = -\boldsymbol{g}_u$, because $\|\boldsymbol{d}^*\|_2 < \Delta$. Combining (3-4) and the definition of $\boldsymbol{V}$ yields the above necessary and sufficient condition for optimality-preserving transformations for models. □

Remark 3.4. Assume that Assumption 3.6 holds. For any $c_2 \in \mathfrak{R}$, $(f(\boldsymbol{y}_1)+c_2, \cdots, f(\boldsymbol{y}_m)+c_2)^\top$ is a solution to the linear system (3-12); therefore, the translation transformation T satisfying $T \circ f = f + c_2$ is an optimality-preserving transformation for models. This conclusion is consistent with Corollary 3.14. In addition, if $|\mathcal{X}| \geq n+2$, then by Theorem 3.7, there exist more optimality-preserving transformations for models.

The solution space of (3-12) contains a linear subspace of translations with dimension at least $m - n$. The least Frobenius norm updating quadratic model used in NEWUOA by Powell [94] requires $m \geq n+2$. For Remark 3.4, if $n+2 \leq m < \frac{1}{2}(n+1)(n+2)$, the optimality-preserving transformation for models can be other transformations beyond those satisfying

$$\begin{pmatrix} T(f(\boldsymbol{y}_1)) \\ \vdots \\ T(f(\boldsymbol{y}_m)) \end{pmatrix} = \begin{pmatrix} f(\boldsymbol{y}_1) + c_2 \\ \vdots \\ f(\boldsymbol{y}_m) + c_2 \end{pmatrix} \tag{3-13}$$

for any $c_2 \in \mathfrak{R}$. In fact, to obtain the fully linear property, the monograph by Conn, Scheinberg, and Vicente [20] points out that at least $n+1$ interpolation points are needed. Below, before proceeding further, we give a natural assumption.

Assumption 3.8. Assume that the homogeneous linear equations of (3-12) with respect to $(T(f(\boldsymbol{y}_1)), \cdots, T(f(\boldsymbol{y}_m)))^\top$ are linearly independent.

We then obtain the following corollary.

Corollary 3.9. *Assume that Assumption 3.6 and Assumption 3.8 hold. If $m = n + 1$, then a transformation T is an optimality-preserving transformation for models if and only if it satisfies (3-13).*

Proof. When Assumption 3.8 holds, if $m = n + 1$, then the dimension of the solution space of (3-12) is 1. Hence the conclusion follows. □

Remark 3.5. If the subproblem used to obtain the quadratic model function is chosen as (3-8), then Theorem 3.7, Corollary 3.9, and the above analysis still apply, with the corresponding matrix $\boldsymbol{V}$ being the inverse of the KKT matrix in (3-9). In addition, when $m \leq n$ and the remaining parts of Assumption 3.6 and Assumption 3.8 hold, a transformation T is an optimality-preserving transformation for models if and only if it satisfies

$$\begin{pmatrix} T(f(\boldsymbol{y}_1)) \\ \vdots \\ T(f(\boldsymbol{y}_m)) \end{pmatrix} = \begin{pmatrix} f(\boldsymbol{y}_1) \\ \vdots \\ f(\boldsymbol{y}_m) \end{pmatrix}.$$

Considering that if $m \leq n$, the solution of (3-12) is unique, the above conclusion holds directly.

An example of an optimality-preserving transformation for models is as follows:

Example 3.1. Assume that the original black-box objective function is

$$f(x, y) = \frac{1}{2}\left((x - y)^2 + (x - 1)^2 + (y - 1)^2\right),$$

where x and y denote the components of a 2D variable. Moreover, the base point $\boldsymbol{x}_0$ and the initial interpolation points $\boldsymbol{y}_1$, $\boldsymbol{y}_2$, $\boldsymbol{y}_3$, $\boldsymbol{y}_4$, $\boldsymbol{y}_5$ are

$$\boldsymbol{x}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ \boldsymbol{y}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ \boldsymbol{y}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \boldsymbol{y}_3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ \boldsymbol{y}_4 = \begin{pmatrix} -1 \\ 0 \end{pmatrix},\ \boldsymbol{y}_5 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}.$$

In this example, we set the trust-region radius to 10. We have

$$f(\boldsymbol{y}_1) = 1,\ f(\boldsymbol{y}_2) = 1,\ f(\boldsymbol{y}_3) = 1,\ f(\boldsymbol{y}_4) = 3,\ f(\boldsymbol{y}_5) = 3,\ \boldsymbol{x}_{\text{opt}}^{(1)} = \boldsymbol{y}_1,$$

and after computing the inverse $\boldsymbol{V}$ of the KKT matrix, we obtain

$$\boldsymbol{\lambda} = \begin{pmatrix} -4 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix},\ c = 1,\ \boldsymbol{g} = \begin{pmatrix} -1 \\ -1 \end{pmatrix},$$

as well as

$$Q_1(x, y) = 1 - x - y + x^2 + y^2.$$

We then add the model minimizer in the trust region, i.e., $\boldsymbol{y}_{\text{new}} = (\frac{1}{2}, \frac{1}{2})^\top$, into the interpolation set and discard the point $\boldsymbol{y}_5$. Moreover, we know that $\boldsymbol{x}_{\text{opt}}^{(2)} = \boldsymbol{y}_{\text{new}}$; for simplicity, we set the base point $\boldsymbol{x}_0$ to $\boldsymbol{x}_{\text{opt}}^{(2)}$. The inverse $\boldsymbol{V}_{\text{new}}$ of the new KKT matrix can be obtained, and we have $f(\boldsymbol{y}_{\text{new}}) = \frac{1}{4}, Q_1(\boldsymbol{y}_{\text{new}}) = \frac{1}{2}$, from which we obtain

$$\boldsymbol{\lambda}^+ = \begin{pmatrix} \frac{2}{3} \\ 1 \\ \frac{4}{3} \\ -\frac{1}{3} \\ -\frac{8}{3} \end{pmatrix}, \; c^+ = -\frac{1}{4}, \; \boldsymbol{g}^+ = \begin{pmatrix} -\frac{1}{3} \\ -\frac{1}{3} \end{pmatrix},$$

$$D(x, y) = -\frac{2}{3}xy + \frac{1}{3}y^2 - \frac{1}{3}y,$$

and

$$Q_2(x, y) = Q_1(x, y) + D(x, y) = x^2 - \frac{2}{3}xy - x + \frac{4}{3}y^2 - \frac{4}{3}y + 1.$$

We then obtain the minimizer of the model function in the trust region $\{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}_{\text{opt}}^{(2)}\|_2 \leq 10\}$, namely $\boldsymbol{d}^* = (\frac{5}{22}, \frac{2}{11})^\top$, and next compute that the next iteration (interpolation) point is $(\frac{8}{11}, \frac{15}{22})^\top$. Substituting $\boldsymbol{d}^*$ into equation (3-12), we obtain the necessary and sufficient conditions

$$\begin{aligned} T(f(\boldsymbol{y}_4)) &= 2 + \frac{9}{10}T(f(\boldsymbol{y}_1)) - \frac{27}{5}T(f(\boldsymbol{y}_2)) + \frac{11}{2}T(f(\boldsymbol{y}_3)), \\ T(f(\boldsymbol{y}_{\text{new}})) &= -\frac{3}{4} + \frac{33}{40}T(f(\boldsymbol{y}_1)) + \frac{21}{20}T(f(\boldsymbol{y}_2)) - \frac{7}{8}T(f(\boldsymbol{y}_3)), \end{aligned} \tag{3-14}$$

whose solution space is

$$\begin{pmatrix} T(f(\boldsymbol{y}_1)) \\ T(f(\boldsymbol{y}_2)) \\ T(f(\boldsymbol{y}_3)) \\ T(f(\boldsymbol{y}_4)) \\ T(f(\boldsymbol{y}_{\text{new}})) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ -\frac{3}{4} \end{pmatrix} + k_1 \begin{pmatrix} 40 \\ 0 \\ 0 \\ 36 \\ 33 \end{pmatrix} + k_2 \begin{pmatrix} 0 \\ 20 \\ 0 \\ -108 \\ 21 \end{pmatrix} + k_3 \begin{pmatrix} 0 \\ 0 \\ 8 \\ 44 \\ -7 \end{pmatrix},$$

where $k_1, k_2, k_3 \in \mathfrak{R}$. We can see that it contains translation transformations (corresponding to constants satisfying $k_2 = 2k_1, k_3 = 5k_1$). Note that the original function values $f(\boldsymbol{y}_1), f(\boldsymbol{y}_2), f(\boldsymbol{y}_3), f(\boldsymbol{y}_4), f(\boldsymbol{y}_{\text{new}})$ also satisfy (3-14).

In Figure 3-2, the top part contains the iteration/interpolation points and the original objective function values at the first iteration. The bottom part contains the iteration/interpolation points at the second iteration and the objective function values after an optimality-preserving transformation for models.

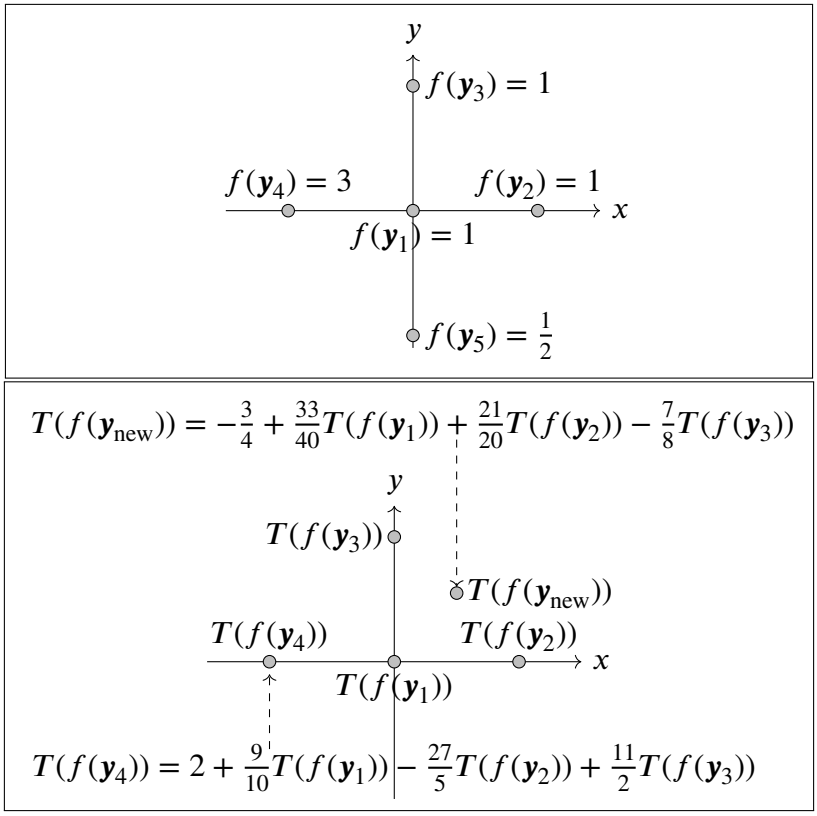


Figure 3-2 Model optimality-preserving transformations in Example 3.1

3.3 Positive Monotonic Transformations and Affine Transformations

Please note that the solution $\boldsymbol{d}_k$ of the trust-region subproblem corresponding to Q_k is an approximation to the subproblem solution corresponding to f_k, because Q_k is a local quadratic interpolation model function of f_k. Therefore, we give below the definition of optimality-preserving transformations for objective functions.

Definition 3.10 (Optimality-preserving transformation for objective functions)**.** *If the subproblem solution of the objective function f under trust-region radius Δ is the same as that of $T \circ f$, then the transformation T is called an optimality-preserving transformation for objective functions under trust-region radius Δ. That is, given a point $\boldsymbol{x}_{\text{opt}} \in \Re^n$, if we have*

$$\arg\min_{\|\boldsymbol{d}\|_2 \le \Delta} f(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}) = \arg\min_{\|\boldsymbol{d}\|_2 \le \Delta} (T \circ f)(\boldsymbol{x}_{\text{opt}} + \boldsymbol{d}),$$

then the transformation T is an optimality-preserving transformation for objective functions corresponding to trust-region radius Δ.

This section will present the objective functions under some basic transformations and the corresponding least Frobenius norm updating quadratic model functions. We first give the definition of positive monotonic transformations.

Definition 3.11. *If a transformation $T : \mathfrak{R} \to \mathfrak{R}$ preserves the order of magnitudes, i.e., for $\theta_1 > \theta_2$ we have $T(\theta_1) > T(\theta_2)$, and for $\theta_1 = \theta_2$ we have $T(\theta_1) = T(\theta_2)$, then we call T a positive monotonic transformation.*

We can directly obtain the following proposition.

Proposition 3.12. *If the transformation T is a positive monotonic transformation, then T is an optimality-preserving transformation for objective functions under any trust-region radius.*

Proof. The conclusion follows directly from Definition 3.11. □

A positive monotonic transformation can be any strictly increasing function, such as a linear function with a positive coefficient (multiplicative coefficient), an exponential function, or a power function with a positive odd exponent. Here, we give the simplest example: an affine transformation.

Example 3.2. An affine transformation T satisfying $T \circ f = c_1 f + c_2$, where $c_1, c_2 \in \mathfrak{R}$ and $c_1 > 0$, is a positive monotonic transformation.

We can see that even if the objective function f is affinely transformed to $c_1 f + c_2$ at some step with $c_1 > 0$, its least Frobenius norm updating quadratic model does not necessarily result from applying the same transformation as for the original objective function. In other words, as we can observe in the previous section, affine transformations are generally not optimality-preserving transformations for models. However, the case where the objective function is affinely transformed before output is fundamental and practically meaningful. Therefore, we will further discuss the objective function after an affine transformation, the analytical expression of its model function, and the constants of the fully linear interpolation model. In addition, we will present the corresponding numerical experiments and attempt to use this as a typical example of an optimality-preserving transformation for objective functions to test and demonstrate the numerical performance of our method.

We provide the following theorem to obtain the expression of the model corresponding to the objective function after an affine transformation. For simplicity, in the remainder of this chapter we use $\boldsymbol{y}_t$ to denote $\boldsymbol{y}_{\text{new}}$, because $\boldsymbol{y}_{\text{new}}$ has been placed in the t-th position of the interpolation set before obtaining the k-th model.

Theorem 3.13. *Assume that Q_α is a quadratic function and that $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$ is a poised set, where $n + 1 \leq m < \frac{1}{2}(n+1)(n+2)$. Then for $c_1, c_2 \in \mathfrak{R}$, we have*

$$\mathcal{M}^{\mathcal{X}}_{Q_\alpha}(c_1 f + c_2) = \left(c_1 \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(f) + c_2\right) + \left(c_1 - 1\right)\left(\mathcal{M}^{\mathcal{X}}_0(Q_\alpha) - Q_\alpha\right), \tag{3-15}$$

where $\mathcal{M}_0^{\mathcal{X}}(Q_\alpha)$ denotes the least Frobenius norm updating quadratic model of Q_α based on the zero function on $\mathcal{X}$, i.e., the least Frobenius norm quadratic model of Q_α.

Proof. Let $Q_\beta := \mathcal{M}_{Q_\alpha}^{\mathcal{X}}(f)$, $\hat{Q}_\beta := \mathcal{M}_{Q_\alpha}^{\mathcal{X}}(c_1 f + c_2)$, and $\tilde{Q} := \mathcal{M}_0^{\mathcal{X}}(Q_\alpha)$. Let $D_\beta := Q_\beta - Q_\alpha$ and $\hat{D}_\beta := \hat{Q}_\beta - Q_\alpha$. Then the quadratic function D_β is the solution of

$$\begin{aligned} &\min_{D \in \mathcal{Q}} \ \left\|\nabla^2 D\right\|_F^2 \\ &\text{s. t. } D(\boldsymbol{y}) = f(\boldsymbol{y}) - Q_\alpha(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}, \end{aligned}$$

and the quadratic function $\hat{D}_\beta$ is the solution of

$$\begin{aligned} &\min_{D \in \mathcal{Q}} \ \left\|\nabla^2 D\right\|_F^2 \\ &\text{s. t. } D(\boldsymbol{y}) = c_1 f(\boldsymbol{y}) + c_2 - Q_\alpha(\boldsymbol{y}),\ \forall\ \boldsymbol{y} \in \mathcal{X}. \end{aligned}$$

We denote the parameters of the quadratic functions D_β and $\hat{D}_\beta$ as $\boldsymbol{\lambda}_D \in \mathfrak{R}^m$, $c_D \in \mathfrak{R}$, $\boldsymbol{g}_D \in \mathfrak{R}^n$ and $\boldsymbol{\lambda}_{\hat{D}} \in \mathfrak{R}^m$, $c_{\hat{D}} \in \mathfrak{R}$, $\boldsymbol{g}_{\hat{D}} \in \mathfrak{R}^n$. Moreover, $(\boldsymbol{\lambda}_D^\top, c_D, \boldsymbol{g}_D^\top)^\top$ and $(\boldsymbol{\lambda}_{\hat{D}}^\top, c_{\hat{D}}, \boldsymbol{g}_{\hat{D}}^\top)^\top$ share the same inverse matrix $\boldsymbol{V}$ of the KKT matrix, that is,

$$\begin{pmatrix} \boldsymbol{\lambda}_D \\ c_D \\ \boldsymbol{g}_D \end{pmatrix} = \boldsymbol{V} \begin{pmatrix} f(\boldsymbol{y}_1) - Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ f(\boldsymbol{y}_m) - Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

$$\begin{pmatrix} \boldsymbol{\lambda}_{\hat{D}} \\ c_{\hat{D}} \\ \boldsymbol{g}_{\hat{D}} \end{pmatrix} = \boldsymbol{V} \begin{pmatrix} c_1 f(\boldsymbol{y}_1) + c_2 - Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ c_1 f(\boldsymbol{y}_m) + c_2 - Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

We can directly obtain

$$\begin{pmatrix} \boldsymbol{\lambda}_{\hat{D}} \\ c_{\hat{D}} \\ \boldsymbol{g}_{\hat{D}} \end{pmatrix} = c_1 \begin{pmatrix} \boldsymbol{\lambda}_D \\ c_D \\ \boldsymbol{g}_D \end{pmatrix} + \boldsymbol{V} \begin{pmatrix} c_2 \\ \vdots \\ c_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + c_1 \boldsymbol{V} \begin{pmatrix} Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix} - \boldsymbol{V} \begin{pmatrix} Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Therefore,

$$\hat{D}_\beta = \left(c_1 D_\beta + c_2\right) + \left(c_1 - 1\right)\tilde{Q},$$

where $\tilde{Q}$ is the solution of the problem

$$\begin{aligned} &\min_{Q \in \mathcal{Q}} \ \left\|\nabla^2 Q\right\|_F^2 \\ &\text{s. t. } Q(\boldsymbol{y}) = Q_\alpha(\boldsymbol{y}),\ \forall\, \boldsymbol{y} \in \mathcal{X}. \end{aligned}$$

Hence,

$$\begin{aligned} \hat{Q}_\beta &= Q_\alpha + \hat{D}_\beta \\ &= Q_\alpha + \left(c_1\left(Q_\beta - Q_\alpha\right) + c_2\right) + \left(c_1 - 1\right)\tilde{Q} \\ &= c_1 Q_\beta + c_2 + \left(c_1 - 1\right)\left(\tilde{Q} - Q_\alpha\right). \end{aligned}$$

Thus, (3-15) holds, and the theorem is proved. □

We can obtain the following corollary.

Corollary 3.14. *Assume $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$ is a poised set, where $n+1 \le m < \frac{1}{2}(n+1)(n+2)$. If L_α is a linear function, then*

$$\mathcal{M}^{\mathcal{X}}_{L_\alpha}(c_1 f + c_2) = c_1 \mathcal{M}^{\mathcal{X}}_{L_\alpha}(f) + c_2 \tag{3-16}$$

holds for $c_1, c_2 \in \mathfrak{R}$.

Proof. Since $|\mathcal{X}| \ge n+1$ and L_α is a linear function, by interpolation we have $\mathcal{M}^{\mathcal{X}}_0(L_\alpha) = L_\alpha$. According to (3-15), (3-16) holds, and the corollary is proved. □

The above corollary corresponds to constructing the least Frobenius norm quadratic model, since $\nabla^2 L_\alpha$ is the zero matrix. In general, $\mathcal{M}^{\mathcal{X}}_0(Q_\alpha) \ne Q_\alpha$. Therefore, the least Frobenius norm updated quadratic model of the function $c_1 f + c_2$ based on Q_α over $\mathcal{X}$ may not be obtained through the same affine transformation, unless $c_1 = 1$. The above analysis also shows that for $c_2 \in \mathfrak{R}$, the translation transformation satisfying $T \circ f = f + c_2$ is a model-optimality-preserving transformation. To further analyze the relationship between affine transformations and model functions, we give the following theorem.

Theorem 3.15. *Suppose Q_α is a quadratic function, and $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$ is a poised set, where $n+1 \le m < \frac{1}{2}(n+1)(n+2)$. Given constants $\nu_1, \nu_2 \in \mathfrak{R}$, we have*

$$\mathcal{M}^{\mathcal{X}}_{\nu_1 Q_\alpha + \nu_2}(f) = \nu_1 \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(f) + (1 - \nu_1)\mathcal{M}^{\mathcal{X}}_0(f). \tag{3-17}$$

Proof. Let $Q_\gamma := \mathcal{M}^{\mathcal{X}}_{\nu_1 Q_\alpha + \nu_2}(f)$, $Q_\beta := \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(f)$, and $Q_\phi := \mathcal{M}^{\mathcal{X}}_0(f)$. Denote

$$Q_\gamma(\boldsymbol{x}) - \nu_1 Q_\alpha(\boldsymbol{x}) - \nu_2 = c_\gamma + \left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{g}_\gamma + \frac{1}{2}\sum_{j=1}^{m} \left(\boldsymbol{\lambda}_\gamma\right)_j \left(\left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right)^2,$$

$$Q_\beta(\boldsymbol{x}) - Q_\alpha(\boldsymbol{x}) = c_\beta + \left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{g}_\beta + \frac{1}{2}\sum_{j=1}^{m} \left(\boldsymbol{\lambda}_\beta\right)_j \left(\left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right)^2,$$

$$Q_\phi(\boldsymbol{x}) = c_\phi + \left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \boldsymbol{g}_\phi + \frac{1}{2}\sum_{j=1}^{m} \left(\boldsymbol{\lambda}_\phi\right)_j \left(\left(\boldsymbol{x} - \boldsymbol{x}_0\right)^\top \left(\boldsymbol{y}_j - \boldsymbol{x}_0\right)\right)^2.$$

We define $\boldsymbol{q}_1 \in \mathfrak{R}^{m+n+1}$ and $\boldsymbol{q}_2 \in \mathfrak{R}^{m+n+1}$ as

$$\boldsymbol{q}_1 = \begin{pmatrix} f(\boldsymbol{y}_1) \\ \vdots \\ f(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \boldsymbol{q}_2 = \begin{pmatrix} Q_\alpha(\boldsymbol{y}_1) \\ \vdots \\ Q_\alpha(\boldsymbol{y}_m) \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Using the expression of the inverse of the KKT matrix, we have

$$\begin{pmatrix} \boldsymbol{\lambda}_\gamma \\ c_\gamma + \nu_2 \\ \boldsymbol{g}_\gamma \end{pmatrix} = \boldsymbol{V}\left(\boldsymbol{q}_1 - \nu_1 \boldsymbol{q}_2\right), \quad \begin{pmatrix} \boldsymbol{\lambda}_\beta \\ c_\beta \\ \boldsymbol{g}_\beta \end{pmatrix} = \boldsymbol{V}\left(\boldsymbol{q}_1 - \boldsymbol{q}_2\right), \quad \begin{pmatrix} \boldsymbol{\lambda}_\phi \\ c_\phi \\ \boldsymbol{g}_\phi \end{pmatrix} = \boldsymbol{V}\boldsymbol{q}_1.$$

Therefore we obtain

$$\begin{pmatrix} \boldsymbol{\lambda}_\gamma \\ c_\gamma + \nu_2 \\ \boldsymbol{g}_\gamma \end{pmatrix} = \nu_1 \begin{pmatrix} \boldsymbol{\lambda}_\beta \\ c_\beta \\ \boldsymbol{g}_\beta \end{pmatrix} + \left(1 - \nu_1\right) \begin{pmatrix} \boldsymbol{\lambda}_\phi \\ c_\phi \\ \boldsymbol{g}_\phi \end{pmatrix}.$$

Thus,

$$Q_\gamma - \nu_1 Q_\alpha = \nu_1\left(Q_\beta - Q_\alpha\right) + \left(1 - \nu_1\right) Q_\phi,$$

and hence (3-17) holds. The theorem is proved. □

To analyze the model function corresponding to the affinely transformed objective function, we derive the following corollary based on Theorem 3.15.

Corollary 3.16. *Suppose $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$ is a poised set, and $\hat{Q}_\alpha$ is the quadratic interpolation model of f on $\mathcal{X}\backslash\{\boldsymbol{y}_t\}$, where $n+1 \le m < \frac{1}{2}(n+1)(n+2)$. Then for any $c_1, c_2 \in \mathfrak{R}$, $c_1\hat{Q}_\alpha + c_2$ is a quadratic interpolation model of $c_1 f + c_2$ on $\mathcal{X}\backslash\{\boldsymbol{y}_t\}$. Furthermore, for any $c_1, c_2 \in \mathfrak{R}$ we have*

$$\mathcal{M}^{\mathcal{X}}_{c_1\hat{Q}_\alpha + c_2}(c_1 f + c_2) - \mathcal{M}^{\mathcal{X}}_0(c_1 f + c_2) = c_1\Big(\mathcal{M}^{\mathcal{X}}_{\hat{Q}_\alpha}(c_1 f + c_2) - \mathcal{M}^{\mathcal{X}}_0(c_1 f + c_2)\Big),$$

where $\mathcal{M}^{\mathcal{X}}_0(c_1 f + c_2)$ is exactly the least Frobenius norm quadratic model of $c_1 f + c_2$.

Proof. This is a direct result of Theorem 3.15 with $\nu_1 = c_1, \nu_2 = c_2$. □

Remark 3.6. Corollary 3.16 discusses the relationship between obtaining the least Frobenius norm updated quadratic model based on the original objective function f and obtaining the updated model based on the transformed objective function $c_1 f + c_2$.

3.4 Fully Linear Models and Convergence Analysis

The convergence analysis in this section is for the standard provable algorithmic framework, namely Algorithm 10.1 in the monograph of Conn, Scheinberg, and Vicente [20], with the difference that we use our least Frobenius norm updated quadratic model to minimize the transformed objective function. The only change in the provable algorithmic framework is the transformed output function values and the use of our model. In other words, the function values used by the algorithm are the transformed values at the newly added points during the iteration. Considering that our model can provide fully linear models, we study the global convergence to first-order critical points in detail. To explore the behavior of the interpolation models under a given affine transformation, we first present the fully linear error constants of the least Frobenius norm updated quadratic model when the objective function is affinely transformed.

3.4.1 Fully Linear Error Constants

We provide the following assumptions and theorem regarding the interpolation error between affinely transformed objective functions and underdetermined quadratic interpolation models.

Assumption 3.17. Assume $\mathcal{X} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_m\} \subset \mathfrak{R}^n$ is a sample/interpolation set contained in $\mathcal{B}_\Delta(\boldsymbol{y}_c)$, and it is well-poised in the sense of linear interpolation or regression, where $\boldsymbol{y}_c \in \mathcal{X}$ and $n+1 \leq |\mathcal{X}| = m < \frac{1}{2}(n+1)(n+2)$.

In addition, we define $\hat{\boldsymbol{L}} = \frac{1}{\Delta}\boldsymbol{L} = \frac{1}{\Delta}(\boldsymbol{y}_1 - \boldsymbol{y}_c, \cdots, \boldsymbol{y}_{c-1} - \boldsymbol{y}_c, \boldsymbol{y}_{c+1} - \boldsymbol{y}_c, \cdots, \boldsymbol{y}_m - \boldsymbol{y}_c)^\top \in \mathfrak{R}^{(m-1)\times n}$ and $\hat{\boldsymbol{L}}^\dagger = (\hat{\boldsymbol{L}}^\top \hat{\boldsymbol{L}})^{-1}\hat{\boldsymbol{L}}^\top$.

Assumption 3.18. Assume Q_α is a quadratic function, and the quadratic model $Q_\beta := \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(f)$ is a fully linear model of function f with error constants κ_g and κ_f [20, 21], that is,

$$\begin{aligned} \left\|\nabla Q_\beta(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\right\|_2 &\leq \kappa_g \Delta, \ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c), \\ \left|Q_\beta(\boldsymbol{x}) - f(\boldsymbol{x})\right| &\leq \kappa_f \Delta^2, \ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c). \end{aligned}$$

Theorem 3.19. *Assume Assumptions 3.17 and 3.18 hold. Then the quadratic model function $\hat{Q}_\beta := \mathcal{M}^{\mathcal{X}}_{Q_\alpha}(c_1 f + c_2)$ is a fully linear model of $c_1 f + c_2$, and as a fully linear model, it has error constants for any $c_1, c_2 \in \mathfrak{R}$:*

$$\begin{aligned} \hat{\kappa}_g &= |c_1|\kappa_g + |c_1 - 1|\left(\frac{5\sqrt{m-1}}{2}\left\|\hat{\boldsymbol{L}}^\dagger\right\|_2\left(\mu_\alpha + \left\|\nabla^2 \tilde{Q}\right\|_2\right)\right), \\ \hat{\kappa}_f &= |c_1|\kappa_f + |c_1 - 1|\left(\frac{5\sqrt{m-1}}{2}\left\|\hat{\boldsymbol{L}}^\dagger\right\|_2 + \frac{1}{2}\right)\left(\mu_\alpha + \left\|\nabla^2 \tilde{Q}\right\|_2\right), \end{aligned}$$

where $\tilde{Q} := \mathcal{M}_0^{\mathcal{X}}(Q_\alpha)$*, and* μ_α *is the Lipschitz constant of the linear function* ∇Q_α*. In other words, we have*

$$\left\|\nabla \hat{Q}_\beta(\boldsymbol{x}) - \nabla(c_1 f(\boldsymbol{x}) + c_2)\right\|_2 \leq \hat{\kappa}_g \Delta,\ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c),$$
$$\left|\hat{Q}_\beta(\boldsymbol{x}) - (c_1 f(\boldsymbol{x}) + c_2)\right| \leq \hat{\kappa}_f \Delta^2,\ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c).$$

Proof. According to Theorem 5.4 in the monograph by Conn, Scheinberg, and Vicente [20], we have

$$\left\|\nabla Q_\alpha(\boldsymbol{x}) - \nabla \tilde{Q}(\boldsymbol{x})\right\|_2 \leq \frac{5\sqrt{m-1}}{2} \left\|\hat{\boldsymbol{L}}^\dagger\right\|_2 \left(\mu_\alpha + \left\|\nabla^2 \tilde{Q}\right\|_2\right) \Delta,\ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c),$$
$$\left|Q_\alpha(\boldsymbol{x}) - \tilde{Q}(\boldsymbol{x})\right| \leq \left(\frac{5\sqrt{m-1}}{2} \left\|\hat{\boldsymbol{L}}^\dagger\right\|_2 + \frac{1}{2}\right) \left(\mu_\alpha + \left\|\nabla^2 \tilde{Q}\right\|_2\right) \Delta^2,\ \forall\, \boldsymbol{x} \in \mathcal{B}_\Delta(\boldsymbol{y}_c).$$

Therefore, combining Theorem 3.13 and Assumption 3.18, the theorem follows. □

3.4.2 Global Convergence to First-Order Critical Points

We now turn to the convergence of our method. We assume that the fully linear error constants of the corresponding models have a uniform upper bound. To avoid confusion, it should be noted that our convergence analysis targets general positive monotonic transformations, not only the affine transformations discussed in Section 3.4.1. We assume that the transformed functions f_k and their gradients are Lipschitz continuous over the corresponding domains.

Assumption 3.20. Assume a given initial point $\boldsymbol{x}_{\text{int}} \in \mathfrak{R}^n$ and an upper bound on the trust-region radius, namely $\Delta_{\max}$. Assume that f and all f_k are continuously differentiable in a region containing the set $\mathcal{L}_{\text{enl}}(\boldsymbol{x}_{\text{int}})$, where

$$\mathcal{L}_{\text{enl}}(\boldsymbol{x}_0) = \bigcup_{\boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_{\text{int}})} \mathcal{B}_{\Delta_{\max}}(\boldsymbol{x}),$$

and $\mathcal{L}(\boldsymbol{x}_{\text{int}}) = \{\boldsymbol{x} \in \mathfrak{R}^n : f(\boldsymbol{x}) \leq f(\boldsymbol{x}_{\text{int}})\}$.

We assume that each transformed function f_k is bounded below, as follows.

Assumption 3.21. Assume that f and all f_k are bounded below on $\mathcal{L}(\boldsymbol{x}_{\text{int}})$, i.e., there exists a constant κ_* such that for all $\boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_{\text{int}})$, $f(\boldsymbol{x}) \geq \kappa_*$ and $f_k(\boldsymbol{x}) \geq \kappa_*$, $\forall\, k \in \mathbb{N}^+$.

For simplicity, we assume that the Hessian matrices of the model functions (i.e., $\nabla^2 Q_k$) are uniformly bounded, as detailed below.

Assumption 3.22. There exists a constant $\kappa_{\text{bhm}} > 0$ such that for all iterations generated by the algorithm, we have $\|\nabla^2 Q_k\|_2 \leq \kappa_{\text{bhm}}$.

Referring to the (same) proof process of the convergence analysis of Algorithm 10.1 in Chapter 10 of the monograph by Conn, Scheinberg, and Vicente [20] (but with the

transformed function f_k in place of the original function f), we can directly obtain the following convergence theorem for the algorithm minimizing the transformed objective function.

Theorem 3.23. *Assume that Assumptions 3.20, 3.21, and 3.22 hold. Assume that for each $k \in \mathbb{N}^+$, the transformation T_k is a positive monotonic transformation, and that the fully linear error constants of the models produced by the algorithm and the Lipschitz constants of the model gradients have uniform upper bounds. Then*

$$\lim_{k\to\infty} \nabla f_k(\boldsymbol{x}_k) = \mathbf{0} \tag{3-18}$$

holds, where $f_k(\boldsymbol{x}) = T_k(f(\boldsymbol{x}))$. Moreover, we have

$$\lim_{k\to\infty} \nabla f(\boldsymbol{x}_k) = \mathbf{0}. \tag{3-19}$$

Proof. The proof of (3-18) is the same as the convergence analysis of model-based derivative-free trust-region methods in Section 10.4 of the monograph by Conn, Scheinberg, and Vicente [20]. Note that the assumption that the fully linear error constants of the models and the Lipschitz constants of the model gradients have uniform upper bounds guarantees the results corresponding to Lemma 10.5 and Lemma 10.6 in the book. In addition, given the positive monotonic transformations, we know that there exists $\varepsilon > 0$ such that

$$\liminf_{k\to\infty} \frac{df_k}{df} > \varepsilon,$$

and thus

$$\nabla f_k(\boldsymbol{x}_k) = \frac{df_k}{df} \nabla f(\boldsymbol{x}_k).$$

Therefore, (3-19) holds. The theorem is proved. □

The transformations in Theorem 3.23 are positive monotonic; they include random affine transformations with positive multiplicative coefficients, such as the random affine transformations corresponding to (3-20).

Considering that Powell's NEWUOA algorithm is a classical and efficient model-based algorithm using the least Frobenius norm updating quadratic model, Section 3.5 will present the results of our improved version based on NEWUOA, named NEWUOA-Trans. It should be emphasized that the above convergence analysis is based on a provable framework, rather than specifically for NEWUOA or NEWUOA-Trans. This is because the complex structure of NEWUOA's code makes its convergence analysis quite difficult, even in the case of minimizing untransformed objective functions, which remains an open problem.

NEWUOA-Trans shares the same framework as NEWUOA, but it updates the corresponding model through (3-7), which can be understood as a direct extension of Powell's

least Frobenius norm updating. In both NEWUOA and NEWUOA-Trans, the model improvement step first attempts to replace interpolation points that are too far from the current $\boldsymbol{x}_{\text{opt}}$ and other interpolation points (for example, replacing points outside the trust region centered at $\boldsymbol{x}_{\text{opt}}$ with radius $2\Delta_k$). When all points in the interpolation set are sufficiently close to each other, NEWUOA-Trans checks the well-posedness of the interpolation set. The model improvement step then finds new interpolation points by maximizing the absolute value related to the corresponding Lagrange polynomial or by updating the denominator of the KKT matrix inverse formula. This process is unaffected by the transformation. If the interpolation set is well-posed, then neither NEWUOA nor NEWUOA-Trans requires further model improvement. In the case where the interpolation set is not well-posed, one point in the set is replaced at each step. Referring to Theorem 6.3 in the monograph by Conn, Scheinberg, and Vicente [20], this guarantees the acquisition of a well-posed interpolation set within the interpolation region, and thus a fully linear model. In fact, the model improvement step ensures that a fully linear model can be produced within a finite number of iterations. Therefore, the interpolation updates of NEWUOA and NEWUOA-Trans guarantee that a fully linear model is constructed within a finite and uniformly bounded number of steps.

3.5 Numerical Results

The previous analysis shows that if $c_1 \neq 1$, then the affine transformation T satisfying $T \circ f = c_1 f + c_2$ with $c_1 > 0$ is generally not a model-optimality-preserving transformation. However, affine transformations are extremely fundamental and important, with practical application value. For example, affine transformations correspond to additive and multiplicative noise mechanisms with different privacy protection schemes in encrypted black-box optimization. In fact, in the previous section, we theoretically analyzed the analytical expression and interpolation error of the least Frobenius norm updating quadratic model for affine-transformed objective functions. In this section, we further observe the performance of our method through numerical experiments.

As mentioned earlier, to solve derivative-free optimization problems with transformed objective functions, we implemented a derivative-free algorithm based on Powell's NEWUOA algorithm [94], and named it NEWUOA-Trans[4]. The underdetermined models used in NEWUOA-Trans are updated via (3-7). This part presents numerical results for solving certain derivative-free optimization problems with transformed objective functions using NEWUOA-Trans. The numerical results illustrate the main characteristics and advantages of NEWUOA-Trans. Overall, NEWUOA-Trans is a robust and efficient algorithm that can be used to minimize transformed objective functions. The code modifications of NEWUOA-Trans mainly occur in the parts updating the Hes-

[4]"-Trans" indicates that it is designed to solve problems with transformed objective functions.

sian and gradient of the model (handling such complex code is not easy). Note that the other parts of NEWUOA-Trans refer to the corresponding parts of NEWUOA.

3.5.1 Algorithm Comparison and Related Transformations

In the numerical experiments, we used the algorithms listed in Table 3-2 to solve derivative-free optimization problems with transformed objective functions. The objective functions of all problems were transformed as shown in (3-20). In addition, we also tested NEWUOA-N. Note that NEWUOA-N solves problems without noise, i.e., the objective functions are untransformed. Here "-N" indicates no noise, and NEWUOA-N can be regarded as a baseline to some extent. Moreover, the comparison between NEWUOA-Trans and NEWUOA-N can indicate whether NEWUOA-Trans can reduce or overcome the impact of transformations on the objective functions. See Table 3-2 for details. In NEWUOA-Trans, NEWUOA-N, and NEWUOA, we set $\hat{\rho}_{\text{beg}} = 10^{-1}$, $\hat{\rho}_{\text{end}} = 10^{-8}$, $m = 2n + 1$ (note that Powell's original notation was ρ_{beg} and ρ_{end}). More details of the NEWUOA framework can be found in Figure 1 of Powell's paper [94].

Table 3-2 Compared algorithms

Algorithm	Model	Problem
NEWUOA-Trans	Our model	Transformed objective
NEWUOA	Powell's model [94]	Transformed objective
NEWUOA-N	Powell's model	Original objective (no transformation)

In the numerical experiments of Section 3.5.2 and Section 3.5.3, at the k-th step, the objective function value $f(\boldsymbol{x})$ at any $\boldsymbol{x}$ in the k-th batch of sampling points will be transformed as

$$f_k(\boldsymbol{x}) = (\gamma_k + 1) f(\boldsymbol{x}) + C\eta_k, \tag{3-20}$$

where $\eta_k \sim \text{Lap}(b_k)$, $b_k > 0$, and $\gamma_k \sim \text{U}(-u_k, u_k)$, $0 < u_k < 1$. The probability density function of $\text{Lap}(b_k)$ is $p(x) = \frac{1}{2b_k} e^{-\frac{|x|}{b_k}}$. Moreover, U denotes the uniform distribution, with the probability density function

$$p(x) = \begin{cases} \dfrac{1}{2u_k}, & \text{if } x \in [-u_k, u_k], \\ 0, & \text{otherwise.} \end{cases}$$

3.5.2 Transformation Attack on the NEWUOA Algorithm: A Simple Example

The following simple example shows that transformations in the objective function (even affine transformations) can cause the unmodified NEWUOA to fail during the solution process. In other words, transformations act as an attack or disturbance on the NEWUOA algorithm.

Example 3.3. In the numerical experiments corresponding to Table 3-3, the objective function is

$$f(\boldsymbol{y}) = \sum_{i=1}^{10} y_i^4 + \sum_{i=1}^{10} y_i^2,$$

where $\boldsymbol{y} = (y_1, \cdots, y_n)^\top$. In this example, the problem dimension n is 10. In addition, the initial point is $(10, \cdots, 10)^\top$, and in (3-20) the constant $C = 1$. The analytic solution of the numerical experiment is $(0, \cdots, 0)^\top$, with the corresponding least function value 0. In Table 3-3, the symbols ✓ and × indicate whether the algorithm successfully solved the problem. The symbol ✓ means that the numerical optimal function value f_{opt} obtained by the algorithm is less than 10^{-3}, while the symbol × means that this accuracy was not achieved. The symbol NF denotes the number of function value evaluations obtained when the iteration terminated. Moreover, NEWUOA-N can find a point with a function value less than 10^{-16} using only 990 function evaluations.

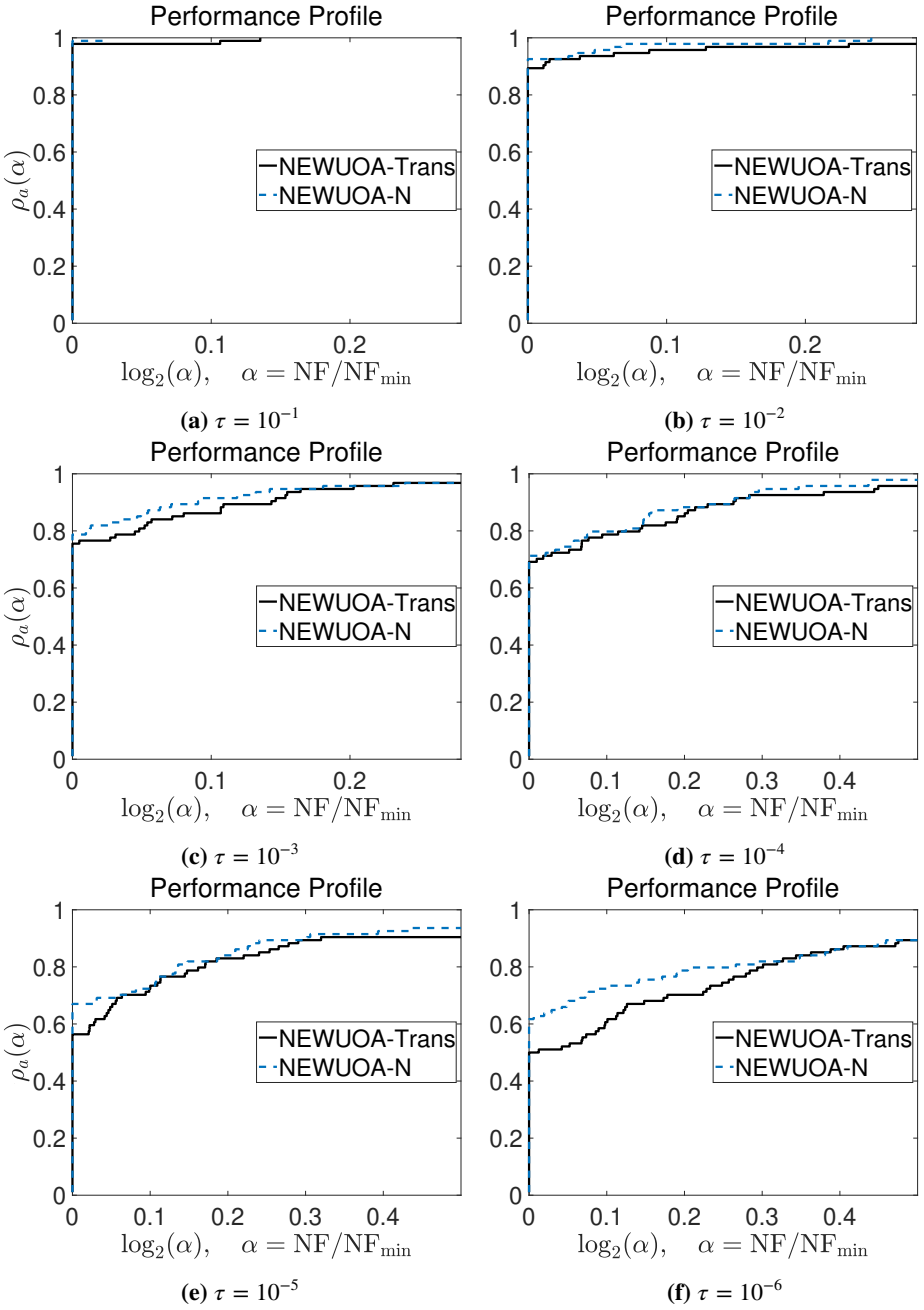
Table 3-3 Numerical results for Example 3.3

Transform para.	$\eta_k \sim \text{Lap}(\frac{1}{k}), \gamma_k = 0$			$\eta_k \sim \text{Lap}(\frac{100}{k}), \gamma_k = 0$		
Algorithm	NF	f_{opt}		NF	f_{opt}	
NEWUOA-Trans	1033	1.5626×10^{-13}	✓	1046	7.7485×10^{-13}	✓
NEWUOA	613	0.1375	×	348	7.2318	×
Transform para.	$\eta_k \sim \text{Lap}(\frac{10}{k}), \gamma_k = 0$			$\eta_k = 0, \gamma_k \sim \text{U}(-\frac{1}{k}, \frac{1}{k})$		
Algorithm	NF	f_{opt}		NF	f_{opt}	
NEWUOA-Trans	847	2.6014×10^{-13}	✓	1055	3.1489×10^{-13}	✓
NEWUOA	542	1.5818	×	408	0.7345	×
Transform para.	$\eta_k \sim \text{Lap}(\frac{100}{k}), \gamma_k \sim \text{U}(-\frac{1}{k}, \frac{1}{k})$			$\eta_k \sim \text{Lap}(\frac{100}{k}), \gamma_k \sim \text{U}(-\frac{k}{10^4}, \frac{k}{10^4})$		
Algorithm	NF	f_{opt}		NF	f_{opt}	
NEWUOA-Trans	1056	4.1928×10^{-13}	✓	948	1.1924×10^{-13}	✓
NEWUOA	432	6.5330	×	409	4.0762	×

From Table 3-3, it can be seen that NEWUOA almost never succeeds in solving problems with simply transformed objective functions. In other words, it performs poorly when solving derivative-free optimization problems with transformed objective functions, which is precisely due to the influence of transformations/noise. Moreover, the results of NEWUOA-N and NEWUOA-Trans are close. Considering that NEWUOA-N serves as a baseline, this shows that NEWUOA-Trans performs satisfactorily when solving optimization problems with transformed objective functions.

3.5.3 Algorithm Performance

We use Performance Profiles to compare different algorithms. The test problems and numerical results shown in Figures 3-3 and 3-4 are listed in Table 4-4. Their dimensions range from 2 to 100 and are drawn from classical and commonly used sets of unconstrained optimization test functions [92, 177, 178, 180, 181, 183–186]. For each algorithm, the maximum number of function evaluations is set to 10000.



(a) $\tau = 10^{-1}$ **(b)** $\tau = 10^{-2}$

(c) $\tau = 10^{-3}$ **(d)** $\tau = 10^{-4}$

(e) $\tau = 10^{-5}$ **(f)** $\tau = 10^{-6}$

Figure 3-3 The comparison of algorithms solving the test problems: Performance Profile

As shown in Example 3.3, the unmodified NEWUOA is not suitable for solving DFOTO problems. Here we compare NEWUOA-Trans and NEWUOA-N. The transformation parameters are set as $C = 100$, $\eta_k \sim \mathrm{Lap}(\frac{100}{k})$, $\gamma_k \sim \mathrm{U}(-\frac{1}{k}, \frac{1}{k})$, and both algo-

Table 3-4 Test problems for Figure 3-3

ARGLINA	ARGLINA4	ARGLINB	ARGLINC	ARGTRIG
ARWHEAD	BDQRTIC	BDQRTICP	BDALUE	BROWNAL
BROYDN3D	BROYDN7D	BRYBND	CHAINWOO	CHEBQUAD
CHNROSNBZ	CHPOWELLB	CHPOWELLS	CHROSEN	COSINECUBE
CURLY10	CURLY20	CURLY30	DIXMAANE	DIXMAANF
DIXMAANG	DIXMAANH	DIXMAANI	DIXMAANJ	DIXMAANK
DIXMAANL	DIXMAANM	DIXMAANN	DIXMAANO	DIXMAANP
DQRTIC	EDENSCH	ENGVAL1	ERRINROS	EXPSUM
EXTROSNB	EXTTET	FIROSE	FLETCBV2	FLETCBV3
FLETCHCR	FMINSRF2	FREUROTH	GENBROWN	GENHUMPS
GENROSE	INDEF	INTEGREQ	LIARWHD	LILIFUN3
LILIFUN4	MOREBV	MOREBVL	NCB20	NCB20B
NONCVXU2	NONCVXUN	NONDIA	NONDQUAR	PENALTY1
PENALTY2	PENALTY3	PENALTY3P	POWELLSG	POWER
ROSENBROCK	SBRYBND	SBRYBNDL	SCHMVETT	SCOSINE
SCOSINEL	SEROSE	SINQUAD	SPARSINE	SPARSQUR
SPHRPTS	SPMSRTLS	SROSENBR	STMOD	TOINTGSS
TOINTTRIG	TQUARTIC	TRIGSABS	TRIGSSQS	TRIROSE1
TRIROSE2	VARDIM	WOODS	-	-

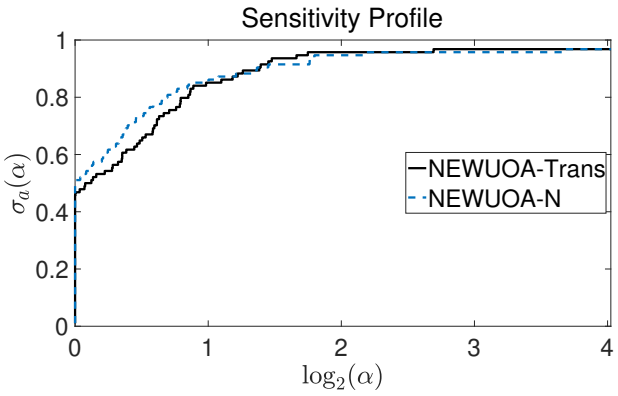


Figure 3-4 The comparison of algorithms solving the test problems: Sensitivity Profile

rithms share the same initial point. In Figures 3-3a to 3-3f, it can be observed that when the accuracy $\tau = 10^{-1}, \cdots, 10^{-6}$, the performance of NEWUOA-Trans and NEWUOA-N is very close. NEWUOA-Trans performs well on problems with transformed objective functions. The comparisons in Figures 3-3a to 3-3f demonstrate that NEWUOA-Trans can successfully solve most derivative-free black-box optimization problems with transformed objective functions. The slight differences between NEWUOA-Trans and NEWUOA-N (the benchmark solving noiseless problems) come from the impact of random noise on the models.

In the numerical results reported in Figure 3-4, the objective functions are selected from the test problems listed in Table 4-4. In addition, $C = 100$ and $\tau = 10^{-4}$. A higher value of $\sigma_a(\alpha)$ indicates stronger stability of the algorithm in the Sensitivity Profile [46]. Figure 3-4 shows that the performance of NEWUOA-Trans is close to NEWUOA-N, which means that the rounding errors of NEWUOA-Trans are comparable to those of our benchmark NEWUOA-N. In fact, the Sensitivity Profile is another important criterion we use to evaluate algorithm stability. We denote by $\boldsymbol{P}_i \in \Re^{n\times n}$, $i = 1, 2, \cdots, M$, random permutation matrices. In the experiment, $M = 100$, and an example of a random permutation matrix is $\boldsymbol{P}_1 = \left(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_4, \boldsymbol{e}_3, \boldsymbol{e}_8, \boldsymbol{e}_5, \boldsymbol{e}_9, \boldsymbol{e}_{10}, \boldsymbol{e}_6, \boldsymbol{e}_7\right)^{\top}$. Furthermore, we define

$$\mathrm{NF} = \left(\mathrm{NF}_1, \cdots, \mathrm{NF}_M\right),$$

where NF_i denotes the number of function evaluations required to solve the problem $\min\limits_{\boldsymbol{x}\in\Re^n} f(\boldsymbol{P}_i\boldsymbol{x})$. We define $\mathrm{mean(NF)} = \frac{1}{M}\sum_{i=1}^{M}\mathrm{NF}_i$, and the standard deviation $\mathrm{std(NF)} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\mathrm{NF}_i - \mathrm{mean(NF)})^2}$. In the Performance Profile, we use std(NF) to replace $N_{a,p}$, corresponding to algorithm a solving problem p, to ultimately obtain $\sigma_a(\alpha)$ and thus construct the Sensitivity Profile.

We can see that the curves of NEWUOA-Trans and NEWUOA-N are close to each other, which shows that the interference from the transformations hardly affects NEWUOA-Trans.

Our numerical experiments demonstrate that NEWUOA-Trans can efficiently and stably solve most derivative-free optimization problems with such transformed objective functions. The performance of NEWUOA-Trans is close to that of NEWUOA-N, indicating that NEWUOA-Trans successfully handles optimization problems with transformed objective functions and overcomes the interference introduced by the transformations.

3.5.4 Experiments on a Real-World Problem

We apply our method to the engineering optimal design of space traveling-wave tubes (TWTs) with privacy protection [206]. A traveling-wave tube is a critical vacuum electronic device [207], which affects signal quality and strength and is widely used in com-

munication, transportation, navigation, meteorological measurement, forecasting, and other fields.

Considering the special working environment of space TWTs, efficiency is a key factor. In the design of space TWTs, design parameters are crucial to efficiency, which is precisely what we aim to maximize. However, the analytical expression of efficiency is unknown, and the relationship between the parameters and the objective function is difficult to analyze. In addition, TWT data are difficult to obtain. Most of the data regarding efficiency performance must come from experiments or simulations, which involve very high cost or time consumption; furthermore, in practice, some data are encrypted. Therefore, solving such an expensive optimization problem is a typical derivative-free optimization problem. As an important industrial product related to security, copyright, and commercial interests, the true values of the efficiency of certain special types of TWTs are encrypted during the optimization process (especially for the public and for third-party designers of optimization algorithms). Hence, optimizing the efficiency of these special types of TWTs is a derivative-free black-box optimization problem with privacy protection, belonging to DFOTO problems with transformations.

We conducted the following numerical experiment, with test data for the space TWT provided by the Beijing Vacuum Electronics Research Institute. The goal of this numerical experiment is to find the optimal design parameters of a space TWT with privacy protection, such that the efficiency is maximized. This is formulated as an unconstrained derivative-free optimization problem with privacy protection[5]:

$$\max_{\mathcal{P}_{\text{input}}} \ \text{Efficiency}\left(\mathcal{P}_{\text{input}}\right),$$

where $\mathcal{P}_{\text{input}}$ is a 10-dimensional vector representing the design parameters of the space TWT. To protect the true values of efficiency (which are directly related to the objective function), the designer of the TWT applies a random affine transformation at each evaluation step to encrypt the true function values during the probing process. The basic probing procedure follows Assumption 3.1 and Table 3-1. We apply NEWUOA-Trans to solve this problem, choosing the initial input $10^{-1} \times (2, \cdots, 2)^{\mathrm{T}}$ as the starting point, with $\hat{\rho}_{\text{beg}} = 10^{-1}$ and $\hat{\rho}_{\text{end}} = 10^{-4}$. NEWUOA-Trans terminated after 226 iterations. The iteration process can be seen in Table 3-5, which shows the Euclidean distance between the best iteration point $\boldsymbol{x}_k$ at the k-th step and the final solution $\boldsymbol{x}^*$.

In order to verify our results, we can use large-scale simulation software CST for simulation design. We found that, in the working frequency band, the efficiency corresponding to the final parameters we obtained shows a significant improvement compared with the best settings based on expert knowledge and experience, as shown in Table 3-6.

[5]For simplicity, some constraints have been adjusted and removed in advance.

Table 3-5 The distance between the best iteration point at the k-th step and the final solution: $\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_2$

Iter.	10	20	30	40	50	60	70	80
Dist.	84.4577	42.6845	19.0530	13.7870	7.7990	4.7825	0.9851	0.8116
Iter.	90	100	110	120	130	140	150	160
Dist.	0.7110	0.5525	0.5106	0.4705	0.4034	0.3035	0.1318	0.1102
Iter.	170	180	190	200	210	220		
Dist.	0.0800	0.0560	0.0370	0.0102	0.0025	0		

Table 3-6 Efficiency increment

Frequency point (GHz)	94	97	100
Efficiency increment (‰)	53	62	66

According to the corresponding industry evaluation mechanism, the parameters obtained by solving the transformed derivative-free optimization problem using NEWUOA-Trans achieve the maximum efficiency of this special space TWT design, and the resulting maximum efficiency is satisfactory within the industry. This demonstrates the strong practicality of our method. The industry also potentially favors the privacy protection feature of our method, mainly because in some cases the data provider can (and only needs to) output the transformed function values. The above preliminary application also inspires us to apply our method in broader fields.

3.6 Conclusion

Before concluding this chapter, we propose an extended transformed optimization problem, which is a new and challenging mathematical programming problem.

Open problem 3.24 (Derivative-free methods for minimizing "moving-target" type objective functions). Attempt to design practical numerical optimization algorithms for the unconstrained problem (3-1), where $f(\boldsymbol{x}, t)$ is the actual output value of the black-box function f at $\boldsymbol{x} \in \mathfrak{R}^n$ and given $t \in \mathfrak{R}$, where t strictly depends on the probing order of the current point $\boldsymbol{x}$, or equivalently, t can be viewed as time. In other words, the set of probed function values will take the form $\{f(\boldsymbol{x}_1, t_1), f(\boldsymbol{x}_2, t_2), \cdots, f(\boldsymbol{x}_k, t_k), \cdots\}$, where t_k may correspond to discrete probing times.

This chapter discussed derivative-free optimization with transformed objective functions. We proposed a corresponding probing scheme. For strictly convex models with a unique minimizer in the trust region, we proved that, besides translation transformations, there exist other model-optimality-preserving transformations. This

chapter proposed sufficient and necessary conditions for transformed function values to preserve model optimality. We obtained the corresponding quadratic models for affine-transformed objective functions and proved that some positive monotone transformations (even affine transformations with positive multiplicative coefficients) are not model-optimality-preserving transformations. We also provided an interpolation error analysis for the corresponding model functions of given affine-transformed objective functions. Convergence analysis for first-order critical points was also provided. The results for test problems and real-world applications numerically demonstrated the advantages of our method.

This chapter represents an initial attempt in this direction, and much remains to be studied regarding derivative-free optimization with transformations. In the future, we will investigate and explore more applications, including details of constructing least Frobenius norm updated quadratic models for derivative-free optimization with transformed objective functions in more engineering applications (e.g., encrypted black-box optimization with noise-adding mechanisms). As discussed in Section 3.4, fully linear error constants may vary at each iteration and lack a uniform bound; in such cases, if the multiplicative coefficient c_1 is unbounded, they may grow unboundedly during iterations. Therefore, analyzing convergence for solving transformed problems under weaker assumptions than those used in Section 3.4 remains an open and challenging problem. Additionally, the open Problem 3.24 on minimizing "moving-target" type objective functions is also interesting and valuable.

Chapter 4 Subspace Methods and Parallel Methods

This chapter focuses on unconstrained derivative-free optimization and proposes a new subspace derivative-free optimization algorithm that can be used to solve large-scale problems. In addition, we also propose a new parallel method that combines trust-region methods and line-search methods.

4.1 Derivative-free Subspace Trust-region Method 2D-MoSub

To solve large-scale unconstrained derivative-free optimization problems, this section introduces a new derivative-free optimization method that effectively and iteratively searches for the optimal solution using subspace techniques and low-dimensional quadratic interpolation models.

Zhang [46] introduced the application of subspace techniques in derivative-free optimization and proposed a framework for a class of derivative-free subspace algorithms. This section introduces the framework and computational details of our newly proposed derivative-free optimization method (2D-MoSub), as well as coordinate transformations related to subspaces. We also discuss the poisedness and quality of interpolation sets, analyze some properties of 2D-MoSub, including approximation error with projection properties and convergence, and present numerical results for solving large-scale problems.

4.1.1 2D-MoSub Algorithm

In this section, we introduce our proposed 2D-MoSub algorithm. Its framework is shown in Algorithm 6.

Algorithm 6 2D-MoSub Algorithm

Input: $\boldsymbol{x}_{\text{int}} \in \Re^n, \Delta_1, \Delta_{\text{low}}, \gamma_1, \gamma_2, \eta, \eta_0, \boldsymbol{d}^{(1)} \in \Re^n$, let $k = 1$.

Step 0. (Initialization)

Obtain $\boldsymbol{y}_a, \boldsymbol{y}_b, \boldsymbol{y}_c$ and $\boldsymbol{d}_1^{(1)}$. Construct the initial 1D quadratic model Q_1^{sub} in the 1D space $\boldsymbol{x}_1 + \text{span}\{\boldsymbol{d}_1^{(1)}\}$.

Step 1. (Construct interpolation set)

Obtain a unit direction $\boldsymbol{d}_2^{(k)} \in \Re^n$ such that $\langle \boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)} \rangle = 0$. Obtain $\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}$.

Step 2. (Construct quadratic interpolation model)

Construct the 2D quadratic model Q_k in the 2D space $\boldsymbol{x}_k + \text{span}\{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}\}$.

Step 3. (Trust-region trial step)

Solve the trust-region subproblem of Q_k and optionally solve the trust-region

subproblem of the modified model Q_k^{mod}, then obtain $\boldsymbol{x}_k^+$. Compute

$$\rho_k = \frac{f(\boldsymbol{x}_k^+) - f(\boldsymbol{x}_k)}{Q_k(\boldsymbol{x}_k^+) - Q_k(\boldsymbol{x}_k)}. \tag{4-1}$$

Update and obtain $\boldsymbol{x}_{k+1}$ and $\boldsymbol{d}_1^{(k+1)}$. Go to **Step 4**.

Step 4. (Update)

If $\Delta_k < \Delta_{\text{low}}$, then terminate. Otherwise, update Δ_{k+1}, let

$$\boldsymbol{d}_1^{(k+1)} = \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\left\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right\|_2},$$

and construct Q_{k+1}^{sub} in the 1D space $\boldsymbol{x}_{k+1} + \text{span}\{\boldsymbol{d}_1^{(k+1)}\}$ as the function Q_k. Let $k = k + 1$ and return to **Step 1**.

We will introduce the details of each part of Algorithm 6 below. Before further discussion, we provide the following remark and definition.

Remark 4.1. In our method, when we discuss the corresponding 1D interpolation model and 2D interpolation model, points in $\mathfrak{R}^n$ are correspondingly viewed as points in a 1D subspace or a 2D subspace.

To meet computational needs, given $\boldsymbol{a} \in \mathfrak{R}^n$ and $\boldsymbol{b} \in \mathfrak{R}^n$, we define the following transformations to represent coordinate mappings from the 2D subspace $S_{\boldsymbol{a},\boldsymbol{b}}^{(k)} = \boldsymbol{x}_k + \text{span}\{\boldsymbol{a}, \boldsymbol{b}\}$ and the 1D subspace $\hat{S}_{\boldsymbol{a}}^{(k)} = \boldsymbol{x}_k + \text{span}\{\boldsymbol{a}\}$ to $\mathfrak{R}^2$ or $\mathfrak{R}$, respectively.

Definition 4.1. *Let $\mathcal{T}_{\boldsymbol{a},\boldsymbol{b}}^{(k)}$ be the transformation from $S_{\boldsymbol{a},\boldsymbol{b}}^{(k)}$ to $\mathfrak{R}^2$, defined as*

$$\mathcal{T}_{\boldsymbol{a},\boldsymbol{b}}^{(k)} : \ \boldsymbol{y} \mapsto \left(\langle \boldsymbol{y} - \boldsymbol{x}_k, \boldsymbol{a}\rangle, \langle \boldsymbol{y} - \boldsymbol{x}_k, \boldsymbol{b}\rangle\right)^\top.$$

Let $\hat{\mathcal{T}}_{\boldsymbol{a}}^{(k)}$ be the transformation from $\hat{S}_{\boldsymbol{a}}^{(k)}$ to $\mathfrak{R}$, defined as

$$\hat{\mathcal{T}}_{\boldsymbol{a}}^{(k)} : \ \boldsymbol{y} \mapsto \langle \boldsymbol{y} - \boldsymbol{x}_k, \boldsymbol{a}\rangle.$$

The algorithm 2D-MoSub begins with the initialization of input parameters and vectors. It first constructs the initial one-dimensional quadratic interpolation model Q_1^{sub}. To start the algorithm, we initialize using the initial point $\boldsymbol{x}_0$ and set various parameters: the trust-region parameters Δ_1 and Δ_{low}, γ_1 and γ_2, as well as the thresholds for successful steps η and η_0. The direction $\boldsymbol{d}^{(1)}$ can be any vector in $\mathfrak{R}^n$. For example, we may choose $\boldsymbol{d}^{(1)} = (1, 0, \cdots, 0)^\top$. Based on extensive numerical experiments, we observe that the choice of $\boldsymbol{d}^{(1)}$ does not fundamentally affect the overall performance of the algorithm.

Algorithm 7 Step 0. Initialization

1: **Input:** Obtain the initial point $\boldsymbol{x}_{\text{int}}$, trust-region parameters Δ_1, Δ_{low}, γ_1, γ_2, as well as η and η_0. Choose a $\boldsymbol{d}^{(1)} \in \mathfrak{R}^n$.

2: Obtain three points: $\boldsymbol{y}_a = \boldsymbol{x}_{\text{int}}$, $\boldsymbol{y}_b = \boldsymbol{x}_{\text{int}} + \Delta_1 \boldsymbol{d}^{(1)}$, and $\boldsymbol{y}_c$ based on the relative values of $f(\boldsymbol{y}_a)$ and $f(\boldsymbol{y}_b)$, namely

$$\boldsymbol{y}_c = \begin{cases} \boldsymbol{y}_a + 2\Delta_1 \boldsymbol{d}^{(1)}, & \text{if } f(\boldsymbol{y}_b) \leq f(\boldsymbol{y}_a), \\ \boldsymbol{y}_a - \Delta_1 \boldsymbol{d}^{(1)}, & \text{otherwise.} \end{cases} \tag{4-2}$$

3: Let $\boldsymbol{x}_1$ be the point among $\boldsymbol{y}_a$, $\boldsymbol{y}_b$, and $\boldsymbol{y}_c$ with the smallest function value, i.e.,

$$\boldsymbol{x}_1 = \arg \min_{\boldsymbol{y} \in \{\boldsymbol{y}_a, \boldsymbol{y}_b, \boldsymbol{y}_c\}} f(\boldsymbol{y}).$$

4: Let $\boldsymbol{y}_{\text{max},1}^{(1)}$ be the point among $\boldsymbol{y}_a$, $\boldsymbol{y}_b$, and $\boldsymbol{y}_c$ with the largest function value, i.e.,

$$\boldsymbol{y}_{\text{max},1}^{(1)} = \arg \max_{\boldsymbol{y} \in \{\boldsymbol{y}_a, \boldsymbol{y}_b, \boldsymbol{y}_c\}} f(\boldsymbol{y}).$$

5: Define $\boldsymbol{d}_1^{(1)}$ as the normalized vector from $\boldsymbol{y}_{\text{max},1}^{(1)}$ to $\boldsymbol{x}_1$, i.e.,

$$\boldsymbol{d}_1^{(1)} = \frac{\boldsymbol{x}_1 - \boldsymbol{y}_{\text{max},1}^{(1)}}{\left\| \boldsymbol{x}_1 - \boldsymbol{y}_{\text{max},1}^{(1)} \right\|_2}.$$

6: Construct the initial one-dimensional quadratic interpolation model Q_1^{sub} in the 1D space $\boldsymbol{x}_1 + \text{span}\{\boldsymbol{d}_1^{(1)}\}$, namely

$$Q_1^{\text{sub}}(\alpha) = f(\boldsymbol{x}_1) + a^{(1)}\alpha + b^{(1)}\alpha^2, \tag{4-3}$$

where $a^{(1)}, b^{(1)} \in \mathfrak{R}$ are determined by the interpolation conditions

$$Q_1^{\text{sub}}(\hat{f}_{\boldsymbol{d}_1^{(1)}}^{(1)}(\boldsymbol{y})) = f(\boldsymbol{y}), \ \forall\, \boldsymbol{y} \in \left\{\boldsymbol{y}_a, \boldsymbol{y}_b, \boldsymbol{y}_c\right\}. \tag{4-4}$$

Once the above initialization of parameters is complete, 2D-MoSub obtains three points: $\boldsymbol{y}_a = \boldsymbol{x}_{\text{int}}$, $\boldsymbol{y}_b = \boldsymbol{x}_{\text{int}} + \Delta_1 \boldsymbol{d}^{(1)}$, and $\boldsymbol{y}_c$ determined by (4-2) according to the relative size of $f(\boldsymbol{y}_a)$ and $f(\boldsymbol{y}_b)$. With these points, 2D-MoSub sets $\boldsymbol{x}_1$ as the one among $\boldsymbol{y}_a$, $\boldsymbol{y}_b$, and $\boldsymbol{y}_c$ with the smallest function value. At the same time, 2D-MoSub sets $\boldsymbol{y}_{\text{max},1}^{(1)}$ as the one among the same set of points with the largest function value. Note that if they have identical function values, we adopt a tie-breaking rule to ensure $\boldsymbol{x}_1 \neq \boldsymbol{y}_{\text{max},1}^{(1)}$. We assume in this section that such selection operations can always be performed correctly, which does not affect the overall concept and effectiveness of the algorithm. Using $\boldsymbol{x}_1$ and $\boldsymbol{y}_{\text{max},1}^{(1)}$, we define $\boldsymbol{d}_1^{(1)}$ as the normalized vector pointing from $\boldsymbol{x}_1$ to $\boldsymbol{y}_{\text{max},1}^{(1)}$. Finally, 2D-MoSub constructs the initial one-dimensional quadratic interpolation model Q_1^{sub} on the space $\boldsymbol{x}_1 + \text{span}\{\boldsymbol{d}_1^{(1)}\}$ according to (4-3) and interpolation condition (4-4). The

Algorithm 8 Step 1. Constructing the Interpolation Set

1: **Input:** $\boldsymbol{x}_k$, $\boldsymbol{d}_1^{(k)}$, Δ_k

2: Select a unit vector $\boldsymbol{d}_2^{(k)}$ such that $\langle \boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)} \rangle = 0$.

3: Define $\boldsymbol{y}_1^{(k)}$ as

$$\boldsymbol{y}_1^{(k)} = \boldsymbol{x}_k + \Delta_k \boldsymbol{d}_2^{(k)}. \tag{4-5}$$

4: Determine $\boldsymbol{y}_2^{(k)}$ based on the relative size of $f(\boldsymbol{y}_1^{(k)})$ and $f(\boldsymbol{x}_k)$:

$$\boldsymbol{y}_2^{(k)} = \begin{cases} \boldsymbol{x}_k + 2\Delta_k \boldsymbol{d}_2^{(k)}, & \text{if } f(\boldsymbol{y}_1^{(k)}) \le f(\boldsymbol{x}_k), \\ \boldsymbol{x}_k - \Delta_k \boldsymbol{d}_2^{(k)}, & \text{otherwise.} \end{cases} \tag{4-6}$$

5: Let $\boldsymbol{y}_{\min,2}^{(k)}$ be the one with the smaller function value between $\boldsymbol{y}_1^{(k)}$ and $\boldsymbol{y}_2^{(k)}$, i.e.,

$$\boldsymbol{y}_{\min,2}^{(k)} = \arg \min_{\boldsymbol{y} \in \{\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}\}} f(\boldsymbol{y}).$$

6: Define $\boldsymbol{y}_3^{(k)}$ as

$$\boldsymbol{y}_3^{(k)} = \boldsymbol{y}_{\min,2}^{(k)} + \Delta_k \boldsymbol{d}_1^{(k)}. \tag{4-7}$$

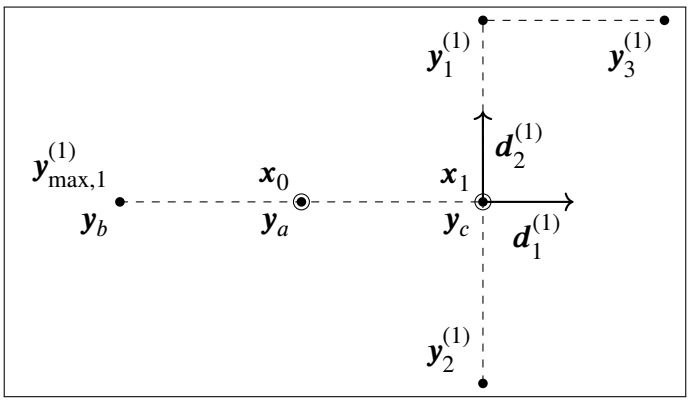


Figure 4-1 The initial case and the subspace $\boldsymbol{x}_1 + \mathbf{span}\{\boldsymbol{d}_1^{(1)}, \boldsymbol{d}_2^{(1)}\}$

details are provided in the pseudocode of the initialization step, which is Step 0 of 2D-MoSub.

Based on the interpolation set in the 2D subspace $\boldsymbol{x}_k + \text{span}\{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}\}$ and the 1D quadratic interpolation model Q_k^{sub}, 2D-MoSub constructs a 2D quadratic interpolation model Q_k. The model Q_k approximates the objective function within the 2D subspace spanned by the selected directions. In addition, if the trial point obtained by minimizing the model Q_k within the trust region does not achieve a sufficiently small function value, our method constructs a modified model Q_k^{mod} based on the already probed points. We now provide the details of constructing the model separately.

After discussing the initial 1D model Q_1^{sub} based on the interpolation condition (4-4), we now introduce how to obtain the k-th model Q_k from Q_k^{sub}.

We construct the 2D quadratic interpolation model Q_k in the 2D subspace $\boldsymbol{x}_k + \text{span}\{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}\}$, namely

$$Q_k(\alpha, \beta) = f(\boldsymbol{x}_k) + a^{(k)}\alpha + b^{(k)}\alpha^2 + c^{(k)}\beta + d^{(k)}\beta^2 + e^{(k)}\alpha\beta, \tag{4-8}$$

where the coefficients $a^{(k)}$ and $b^{(k)}$ are inherited directly from Q_k^{sub}, while the coefficients $c^{(k)}$, $d^{(k)}$, and $e^{(k)}$ are determined by the interpolation condition

$$Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \left\{ \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)} \right\}. \tag{4-9}$$

The details are provided in the algorithmic pseudocode for the quadratic interpolation model construction step, denoted as Step 2 of 2D-MoSub.

Algorithm 9 Step 2. Constructing the Quadratic Interpolation Model

1: In the 2D space $\boldsymbol{x}_k + \text{span}\{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}\}$, construct the 2D quadratic interpolation model Q_k (as in (4-8)), where $c^{(k)}$, $d^{(k)}$, and $e^{(k)}$ are determined by the interpolation condition (4-9).

We consider the above way of obtaining Q_k to be reasonable and reliable, because when $\boldsymbol{x}_k$ is a successful trust-region trial step, the vector $\boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ numerically provides a sufficiently good direction and the corresponding 1D subspace. Moreover, in the previous 2D subspace, the model Q_{k-1}^{+} along this 1D subspace is also a sufficiently good approximation. In most cases, this 1D subspace is the intersection between the $(k-1)$-th 2D subspace and the k-th 2D subspace. In other words, Q_k maintains optimality consistency with the previous good model within the corresponding 1D subspace.

We now introduce how our algorithm obtains the $(k+1)$-th model function Q_{k+1}^{sub} on the 1D subspace based on the k-th model function Q_k. At Step k, we already have the model function Q_k with determined coefficients. However, after obtaining the iterate $\boldsymbol{x}_{k+1}$ and its function value $f(\boldsymbol{x}_{k+1})$, we usually have

$$Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}}(\boldsymbol{x}_{k+1})) \neq f(\boldsymbol{x}_{k+1}),$$

which indicates that Q_k is not a sufficiently good interpolation of the function f after incorporating information at $\boldsymbol{x}_{k+1}$. Therefore, before constructing the model Q_{k+1}^{sub}, we tend to update Q_k to the following Q_k^+, namely

$$Q_k^+(\alpha,\beta) = f(\boldsymbol{x}_{k+1}) + \bar{a}^{(k)}\alpha + \bar{b}^{(k)}\alpha^2 + \bar{c}^{(k)}\beta + \bar{d}^{(k)}\beta^2 + \bar{e}^{(k)}\alpha\beta, \tag{4-10}$$

which satisfies the interpolation condition

$$Q_k^+(\mathcal{T}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)},\boldsymbol{d}_*^{(k+1)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \left\{\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\right\},$$

where $\boldsymbol{d}_*^{(k+1)} \in \text{span}\{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}\}$ satisfies $\langle \boldsymbol{d}_*^{(k+1)}, \boldsymbol{d}_1^{(k+1)}\rangle = 0$. We provide more details on constructing the quadratic interpolation model Q_k^+ below.

Note that in some cases, the set of interpolation points in (4-10) is not well-posed. Therefore, 2D-MoSub checks whether the corresponding coefficient matrix of each interpolation system is invertible. If the current interpolation set is not well-posed in this sense, 2D-MoSub prepares different interpolation points as alternatives. It tests the invertibility of the coefficient matrices associated with these sets and uses a well-posed set to obtain the quadratic model Q_k^+ by solving the interpolation equations. The alternative set of interpolation points is composed of six points selected from

$$\mathcal{Y}_k^+ = \left\{\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}, \boldsymbol{y}_4^{(k)}, \boldsymbol{y}_5^{(k)}\right\},$$

where $\boldsymbol{y}_4^{(k)} = \boldsymbol{x}_k + \frac{\sqrt{2}}{2}\Delta_k \boldsymbol{d}_1^{(k)} + \frac{\sqrt{2}}{2}\Delta_k \boldsymbol{d}_2^{(k)}$, and $\boldsymbol{y}_5^{(k)} = \boldsymbol{x}_k + \Delta_k \boldsymbol{d}_1^{(k)}$. Note that the points $\boldsymbol{x}_k, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}, \boldsymbol{y}_4^{(k)}, \boldsymbol{y}_5^{(k)}$ are fixed in distribution, so such alternatives are feasible. Therefore, we obtain Q_k^+ according to the interpolation condition

$$Q_k^+(\mathcal{T}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)},\boldsymbol{d}_*^{(k+1)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \mathcal{Y} \tag{4-11}$$

where $\mathcal{Y} \subset \mathcal{Y}_k^+$ and $\mathcal{Y}$ is a well-posed set containing six interpolation points.

The above correction ensures that the interpolation set used in constructing the quadratic model Q_k^+ is well-posed. Consequently, we can obtain a reliable 2D quadratic model Q_k^+. Then, the 1D model Q_{k+1}^{sub} is set as

$$Q_{k+1}^{\text{sub}}(\alpha) = Q_k^+(\alpha, 0),\ \forall\ \alpha \in \mathfrak{R}. \tag{4-12}$$

In fact, 2D-MoSub saves computational cost in a certain sense, since it does not require executing a complicated subroutine to implement model improvement as in traditional algorithms.

Remark 4.2. In successful steps there is another way to obtain the corrected model Q_k^+. It is based on the updated interpolation set $\mathcal{X}_k^+ = \mathcal{X}_k \cup \{\boldsymbol{x}_{k+1}\}\backslash\{\boldsymbol{x}_{k-1}\}$ using the minimum norm updating method. In this case, we only need to solve the KKT system corresponding to the minimum norm updated quadratic model subproblem, and the user can freely choose how to perform the correction.

If the trial point $\boldsymbol{x}_k^+$ obtained by solving the trust-region subproblem of the model function Q_k fails to achieve sufficient reduction in the function value, provided that $\boldsymbol{x}_k^+ \notin \{\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\}$, our method will re-solve the trust-region subproblem of the corrected quadratic model Q_k^{mod}. In this case, we construct the corrected model Q_k^{mod} via the interpolation condition

$$Q_k^{\text{mod}}(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \mathcal{Y}_k^{\text{mod}}, \tag{4-13}$$

where

$$\mathcal{Y}_k^{\text{mod}} = \begin{cases} \left\{\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_k^+, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\right\}, \text{ if } \boldsymbol{x}_k \neq \boldsymbol{x}_{k-1}, \\ \left\{\boldsymbol{x}_k, \boldsymbol{x}_k^+, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}, \boldsymbol{y}_4^{(k)}\right\}, \text{ if } \boldsymbol{x}_k = \boldsymbol{x}_{k-1} \text{ and } \boldsymbol{x}_k^+ \neq \boldsymbol{y}_4^{(k)}, \\ \left\{\boldsymbol{x}_k, \boldsymbol{x}_k^+, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}, \boldsymbol{y}_5^{(k)}\right\}, \text{ otherwise.} \end{cases}$$

Note that the function values of all interpolation points above have already been evaluated, so such interpolation does not require additional function evaluations.

The quadratic models are very important for characterizing the nature of the iterations obtained by solving the 2D trust-region subproblem. Table 4-1 presents the interpolation conditions of the models used in our method.

Table 4-1 Interpolation conditions for models used in 2D-MoSub

Model	Dim.	Interpolation condition
Q_k^{sub}	1	$Q_k^{\text{sub}}(\alpha) = Q_{k-1}^+(\alpha, 0)$
Q_k	2	$Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \{\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\}$ & $Q_k(\alpha, 0) = Q_k^{\text{sub}}(\alpha)$
Q_k^+	2	$Q_k^+(\mathcal{T}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)},\boldsymbol{d}_*^{(k+1)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \mathcal{Y}$, where $\mathcal{Y} \subset \mathcal{Y}_k^+$ and $\lvert\mathcal{Y}\rvert = 6$
Q_k^{mod}	2	$Q_k^{\text{mod}}(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\ \boldsymbol{z} \in \mathcal{Y}_k^{\text{mod}}$

Considering that in most cases we have $\boldsymbol{d}_1^{(k)} = \frac{\boldsymbol{x}_k - \boldsymbol{x}_{k-1}}{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2}$, which is approximately a gradient descent direction, it can be regarded that the new model Q_k inherits the good properties of the models Q_{k-1} and Q_{k-1}^+ along the approximate gradient descent direction.

Similar to traditional trust-region methods, 2D-MoSub finds the optimal trial step within the corresponding trust region by solving a 2D trust-region subproblem. It then evaluates the quality of the trial step by using the ratio of the actual reduction in the function value to the predicted reduction in the model value. Based on this ratio and predefined thresholds, the algorithm updates the subspace, interpolation set, trust-region parameters, and iteration points.

Algorithm 10 Step 3. Trust-region trial step

1: Solve the trust-region subproblem

$$\min_{\alpha,\beta} Q_k(\alpha,\beta)$$
$$\text{s. t. } \alpha^2+\beta^2 \le \Delta_k^2$$

and obtain $\alpha^{(k)}$ and $\beta^{(k)}$. Then set

$$\boldsymbol{x}_k^{\text{pre}} = \boldsymbol{x}_k + \alpha^{(k)}\boldsymbol{d}_1^{(k)} + \beta^{(k)}\boldsymbol{d}_2^{(k)},$$
$$\boldsymbol{x}_k^{+} = \min_{\boldsymbol{x}\in\{\boldsymbol{x}_k,\boldsymbol{x}_k^{\text{pre}},\boldsymbol{y}_1^{(k)},\boldsymbol{y}_2^{(k)},\boldsymbol{y}_3^{(k)}\}} f(\boldsymbol{x}).$$

2: If $\boldsymbol{x}_k^{+} \in \{\boldsymbol{x}_k, \boldsymbol{x}_{k-1}\}$, then set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, $\Delta_{k+1} = \Delta_k$ without applying (4-14), and set $\boldsymbol{d}_1^{(k+1)} = \boldsymbol{d}_1^{(k)}$ without applying (4-15), then go to **Step 4** of the 2D-MoSub framework.

3: Otherwise, compute

$$\rho_k = \frac{f(\boldsymbol{x}_k^{+}) - f(\boldsymbol{x}_k)}{Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{x}_k^{+})) - Q_k(0,0)}.$$

4: If $\rho_k \ge \eta$ or $\boldsymbol{x}_k^{+} \in \{\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\}$, then set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k^{+}$ and go to **Step 4** of the 2D-MoSub framework.

5: Otherwise, obtain the modified model Q_k^{mod} through (4-13) and solve the trust-region subproblem

$$\min_{\alpha,\beta} Q_k^{\text{mod}}(\alpha,\beta)$$
$$\text{s. t. } \alpha^2+\beta^2 \le \Delta_k^2,$$

and obtain $\alpha^{(k,\text{mod})}$ and $\beta^{(k,\text{mod})}$. Then set

$$\boldsymbol{x}_k^{\text{mod}} = \boldsymbol{x}_k + \alpha^{(k,\text{mod})}\boldsymbol{d}_1^{(k)} + \beta^{(k,\text{mod})}\boldsymbol{d}_2^{(k)}.$$

6: If $\boldsymbol{x}_k^{\text{mod}} \in \{\boldsymbol{x}_k, \boldsymbol{x}_{k-1}\}$, then set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, $\Delta_{k+1} = \Delta_k$ without applying (4-14), and set $\boldsymbol{d}_1^{(k+1)} = \boldsymbol{d}_1^{(k)}$ without applying (4-15), then go to **Step 4** of the 2D-MoSub framework.

7: Otherwise, set

$$\boldsymbol{x}_k^{+} = \arg\min_{\boldsymbol{x}\in\{\boldsymbol{x}_k^{+},\boldsymbol{x}_k^{\text{mod}}\}} f(\boldsymbol{x}).$$

Compute

$$\rho_k = \frac{f(\boldsymbol{x}_k^{+}) - f(\boldsymbol{x}_k)}{Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{x}_k^{+})) - Q_k(\boldsymbol{x}_k)}.$$

8: If $\rho_k \ge \eta_0$, then set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k^{+}$ and go to **Step 4** of the 2D-MoSub framework.

9: Otherwise, set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, $\boldsymbol{d}_1^{(k+1)} = \boldsymbol{d}_1^{(k)}$ without applying (4-15), then go to **Step 4** of the 2D-MoSub framework.

In the trust-region trial step, we perform the following operations. First, we solve the trust-region subproblem to find the minimizer of the quadratic model $Q_k(\alpha, \beta)$ within the trust region. Next, we compute the corresponding decrease in the function value based on Q_k and the objective function f. Then, we calculate the ratio between the actual reduction in the objective function and the predicted reduction in the model function, and compare it with predefined thresholds. We then update Δ_k and $\boldsymbol{d}_1^{(k)}$ accordingly. The above explanation only gives the basic idea of the trust-region trial step; further details are provided in the pseudocode of Step 3 of the algorithm.

In the trust-region trial step, we follow the pseudocode in Step 3 of the algorithmic framework. In our current test implementation, the algorithm solves the trust-region subproblem using the truncated conjugate gradient method [161, 162].

Similar to traditional trust-region methods, 2D-MoSub updates the trust-region radius based on the quality of the trial step. If the radius becomes smaller than the lower bound, 2D-MoSub terminates. Otherwise, the algorithm updates the subspace by computing a new direction based on the updated solution. 2D-MoSub then constructs a new 1D quadratic interpolation model in the updated subspace. The details are given in the pseudocode for the update step, which is listed as Step 4 of 2D-MoSub.

Algorithm 11 Step 4. Update

1: If $\Delta_k < \Delta_{\text{low}}$, then terminate.

2: Otherwise, update Δ_{k+1} as

$$\Delta_{k+1} = \begin{cases} \gamma_1 \Delta_k, \text{ if } \rho_k \geq \eta, \\ \gamma_2 \Delta_k, \text{ otherwise.} \end{cases} \tag{4-14}$$

3: Set

$$\boldsymbol{d}_1^{(k+1)} = \frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{\left\| \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \right\|_2}. \tag{4-15}$$

4: Update Q_k according to the interpolation condition (4-11), and obtain Q_k^+.

5: Obtain the 1D model Q_{k+1}^{sub} satisfying (4-12).

6: Let $k = k + 1$, and go to **Step 1** of the 2D-MoSub framework.

Figure 4-2 illustrates the k-th iteration of 2D-MoSub.

4.1.2 Poisedness and Quality of the Interpolation Set

In this section, we discuss and analyze the poisedness and quality of the interpolation set used at each step when constructing the quadratic interpolation model Q_k. As we have noted, the quality of an interpolation model in a given region is determined by the

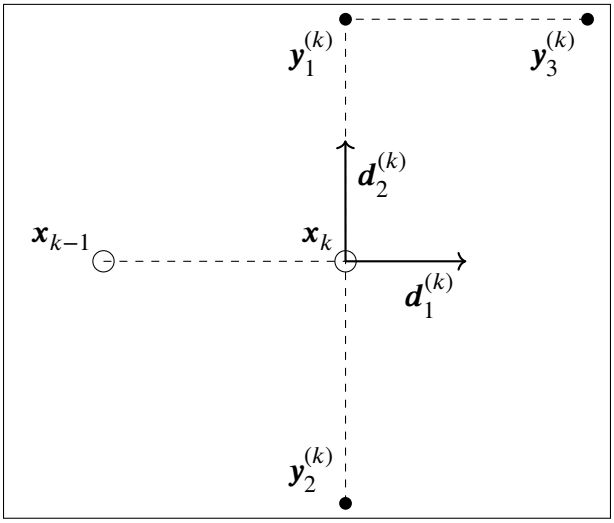


Figure 4-2 The iterative case at the k-th step and the subspace $x_k + \mathbf{span}\{d_1^{(k)}, d_2^{(k)}\}$

locations of its interpolation points. It is well known that Λ-poisedness is a concept that measures how well a set of points is distributed, which ultimately determines how well the interpolation model approximates the objective function.

As mentioned earlier, the most commonly used measure of the poisedness of points in the region of interest at the current iteration is based on the Lagrange polynomials. Given a set $\mathcal{Y} = \{y_1, \cdots, y_p\}$ containing p points, the basis of the Lagrange polynomials satisfies

$$l_j(y_i) = \begin{cases} 1, \text{ if } i = j, \\ 0, \text{ otherwise.} \end{cases}$$

Next, we briefly recall the most classical definition of Λ-poisedness [20].

Definition 4.2 (Λ-poisedness)**.** *If a set of points $\mathcal{Y}$ is linearly independent over the set $\mathcal{B}$, and the corresponding Lagrange polynomials $\{l_1, \cdots, l_p\}$ satisfy*

$$\Lambda \geq \max_{1 \leq i \leq p} \max_{x \in \mathcal{B}} \left| l_i(x) \right|,$$

then we say that $\mathcal{Y}$ is Λ-poised on $\mathcal{B}$.

In our case, considering that our algorithm constructs a new 2D model Q_k with 3 fixed coefficients and 3 unknown coefficients determined by the interpolation conditions (4-9), we give the following definition and discussion.

Definition 4.3 (Basis functions)**.** *Given $y_1^{(k)}, y_2^{(k)}, y_3^{(k)}, x_k$ and $d_1^{(k)}, d_2^{(k)} \in \Re^n$, let*

$$(\alpha_i^{(k)}, \beta_i^{(k)})^\top = \mathcal{T}^{(k)}_{d_1^{(k)}, d_2^{(k)}}(y_i^{(k)}), \ i = 1, 2, 3,$$

and the basis matrix[1]

$$\boldsymbol{\Psi} = \begin{pmatrix} \beta_1^{(k)} & (\beta_1^{(k)})^2 & \alpha_1^{(k)}\beta_1^{(k)} \\ \beta_2^{(k)} & (\beta_2^{(k)})^2 & \alpha_2^{(k)}\beta_2^{(k)} \\ \beta_3^{(k)} & (\beta_3^{(k)})^2 & \alpha_3^{(k)}\beta_3^{(k)} \end{pmatrix},$$

with $c_0^{(i)} = f(\boldsymbol{x}_k) + a^{(k)}\alpha_i^{(k)} + b^{(k)}(\alpha_i^{(k)})^2, i = 1, 2, 3$, *and*

$$\begin{pmatrix} h_1 \\ d_1 \\ e_1 \end{pmatrix} = \boldsymbol{\Psi}^{-1} \begin{pmatrix} 1 - c_0^{(1)} \\ -c_0^{(1)} \\ -c_0^{(1)} \end{pmatrix}, \begin{pmatrix} h_2 \\ d_2 \\ e_2 \end{pmatrix} = \boldsymbol{\Psi}^{-1} \begin{pmatrix} -c_0^{(2)} \\ 1 - c_0^{(2)} \\ -c_0^{(2)} \end{pmatrix}, \begin{pmatrix} h_3 \\ d_3 \\ e_3 \end{pmatrix} = \boldsymbol{\Psi}^{-1} \begin{pmatrix} -c_0^{(3)} \\ -c_0^{(3)} \\ 1 - c_0^{(3)} \end{pmatrix}.$$

The basis functions are

$$l_i(\alpha, \beta) = c_0^{(i)} + h_i\beta + d_i\beta^2 + e_i\alpha\beta, \ i = 1, 2, 3. \tag{4-16}$$

We now define Λ-poisedness in our specific setting.

Definition 4.4 (Λ-poisedness with 3 known coefficients in 2D)**.** *If a set* $\mathcal{Y} \subset \mathfrak{R}^2$ *containing 3 points is linearly independent, and the basis polynomials* $\{l_1, l_2, l_3\}$ *associated with* $\mathcal{Y}$ *satisfy*

$$\Lambda \geq \max_{1 \leq i \leq 3} \max_{\|(\alpha,\beta)^\top\|_\infty \leq \Delta_k} |l_i(\alpha, \beta)|, \tag{4-17}$$

then we say that $\mathcal{Y}$ *is* Λ*-poised over the set* $\{(\alpha, \beta)^\top : \left\|(\alpha, \beta)^\top\right\|_\infty \leq \Delta_k\}$.

Remark 4.3. Under this measure, the most poised interpolation set is 1-poised. Note that in Definition 4.4, the region is an ℓ_∞-norm ball, which is adopted to obtain a closed-form analytic solution without loss of generality.

Theorem 4.5. *During each iteration in the quadratic interpolation model construction step, 2D-MoSub has the following Lagrange basis functions for computation.*

In the case where $f(\boldsymbol{x}_k) \leq f(\boldsymbol{y}_1^{(k)})$*, we have*

$$\begin{cases} l_1(\alpha, \beta) = c_0^{(1)} + \dfrac{1}{2\Delta_k}\beta + \dfrac{1 - 2c_0^{(1)}}{2\Delta_k^2}\beta^2 + \left(-\dfrac{1}{\Delta_k^2}\right)\alpha\beta, \\ l_2(\alpha, \beta) = c_0^{(2)} + \left(-\dfrac{1}{2\Delta_k}\right)\beta + \dfrac{1 - 2c_0^{(2)}}{2\Delta_k^2}\beta^2, \\ l_3(\alpha, \beta) = c_0^{(3)} + \left(-\dfrac{c_0^{(3)}}{\Delta_k^2}\right)\beta^2 + \dfrac{1}{\Delta_k^2}\alpha\beta, \end{cases} \tag{4-18}$$

[1] We consider the invertible case.

In the case where $f(\boldsymbol{x}_k) > f(\boldsymbol{y}_1^{(k)})$*, we have*

$$\begin{cases} l_1(\alpha,\beta) = c_0^{(1)} + \dfrac{4-3c_0^{(1)}}{2\Delta_k}\beta + \dfrac{-2+c_0^{(1)}}{2\Delta_k^2}\beta^2 + \left(-\dfrac{1}{\Delta_k^2}\right)\alpha\beta, \\ l_2(\alpha,\beta) = c_0^{(2)} + \left(-\dfrac{1+3c_0^{(2)}}{2\Delta_k}\right)\beta + \dfrac{1+c_0^{(2)}}{2\Delta_k^2}\beta^2, \\ l_3(\alpha,\beta) = c_0^{(3)} + \left(-\dfrac{3c_0^{(3)}}{2\Delta_k}\right)\beta + \dfrac{c_0^{(3)}}{2\Delta_k^2}\beta^2 + \dfrac{1}{\Delta_k^2}\alpha\beta, \end{cases} \tag{4-19}$$

where $c_0^{(1)} = f(\boldsymbol{x}_k),\ c_0^{(2)} = f(\boldsymbol{x}_k),\ c_0^{(3)} = f(\boldsymbol{x}_k) + a^{(k)}\Delta_k + b^{(k)}\Delta_k^2$.

Proof. In the case where $f(\boldsymbol{x}_k) \le f(\boldsymbol{y}_1^{(k)})$, we have

$$\boldsymbol{\Psi}_1 = \begin{pmatrix} \Delta_k & \Delta_k^2 & 0 \\ -\Delta_k & \Delta_k^2 & 0 \\ \Delta_k & \Delta_k^2 & \Delta_k^2 \end{pmatrix}, \tag{4-20}$$

and

$$\begin{pmatrix} h_1 \\ d_1 \\ e_1 \end{pmatrix} = \boldsymbol{\Psi}_1^{-1}\begin{pmatrix} 1-c_0^{(1)} \\ -c_0^{(1)} \\ -c_0^{(1)} \end{pmatrix}, \begin{pmatrix} h_2 \\ d_2 \\ e_2 \end{pmatrix} = \boldsymbol{\Psi}_1^{-1}\begin{pmatrix} -c_0^{(2)} \\ 1-c_0^{(2)} \\ -c_0^{(2)} \end{pmatrix}, \begin{pmatrix} h_3 \\ d_3 \\ e_3 \end{pmatrix} = \boldsymbol{\Psi}_1^{-1}\begin{pmatrix} -c_0^{(3)} \\ -c_0^{(3)} \\ 1-c_0^{(3)} \end{pmatrix}.$$

In the case where $f(\boldsymbol{x}_k) > f(\boldsymbol{y}_1^{(k)})$, we have

$$\boldsymbol{\Psi}_2 = \begin{pmatrix} \Delta_k & \Delta_k^2 & 0 \\ 2\Delta_k & 4\Delta_k^2 & 0 \\ \Delta_k & \Delta_k^2 & \Delta_k^2 \end{pmatrix}, \tag{4-21}$$

and

$$\begin{pmatrix} h_1 \\ d_1 \\ e_1 \end{pmatrix} = \boldsymbol{\Psi}_2^{-1}\begin{pmatrix} 1-c_0^{(1)} \\ -c_0^{(1)} \\ -c_0^{(1)} \end{pmatrix}, \begin{pmatrix} h_2 \\ d_2 \\ e_2 \end{pmatrix} = \boldsymbol{\Psi}_2^{-1}\begin{pmatrix} -c_0^{(2)} \\ 1-c_0^{(2)} \\ -c_0^{(2)} \end{pmatrix}, \begin{pmatrix} h_3 \\ d_3 \\ e_3 \end{pmatrix} = \boldsymbol{\Psi}_2^{-1}\begin{pmatrix} -c_0^{(3)} \\ -c_0^{(3)} \\ 1-c_0^{(3)} \end{pmatrix}.$$

Thus, we can conclude (4-18) and (4-19). □

Figure 4-3 illustrates the different cases of $\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}$.

Proposition 4.6. *In the case* $f(\boldsymbol{x}_k) \le f(\boldsymbol{y}_1^{(k)})$*, the interpolation set* $\{\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\}$ *is* Λ_1*-poised with* $\Lambda_1 = 2$*. In the case* $f(\boldsymbol{x}_k) > f(\boldsymbol{y}_1^{(k)})$*, the interpolation set* $\{\boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\}$ *is* Λ_2*-poised with* $\Lambda_2 \le \max\{4, 1 + 3\Delta_k(|a^{(k)}| + |b^{(k)}|\Delta_k)\}$.

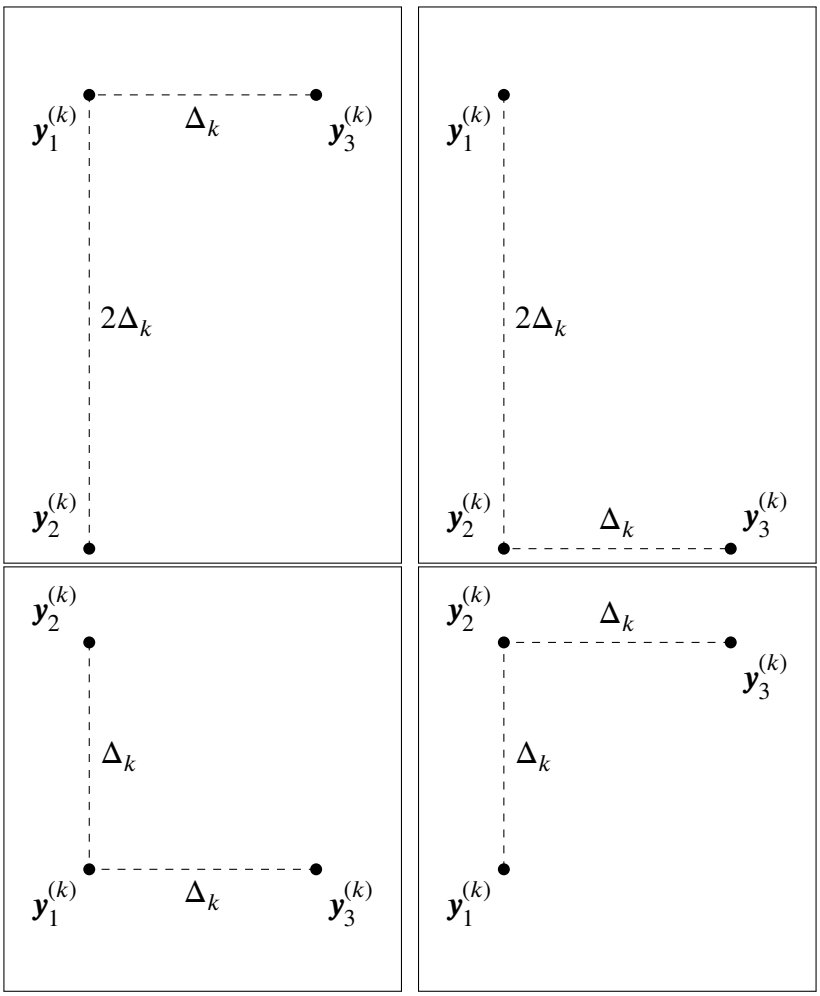


Figure 4-3 Different cases for $y_1^{(k)}, y_2^{(k)}, y_3^{(k)}$

Proof. According to Theorem 4.5 and the closed-form solutions of the two-dimensional problems in this proposition,

$$\max_{1\leq i\leq 3}\ \max_{\left\|(\alpha,\beta)^\top\right\|_\infty\leq\Delta_k}\ \left|l_i(\alpha,\beta)\right|$$

the conclusion follows directly after computation. □

In view of Remark 4.3, the interpolation set used by the 2D-MoSub algorithm is sufficiently poised on the 2D subspace.

4.1.3 Some Properties of 2D-MoSub

The main idea of our newly proposed subspace derivative-free optimization method 2D-MoSub is to obtain an iterate at each step by minimizing a quadratic model function within a trust region on a 2D subspace. The quadratic model in the current 2D subspace, together with the model defined along one dimension of the 2D subspace, inherits the good properties of the previous subspaces, models, and iterates.

To construct a determined quadratic interpolation model function at each step, 2D-MoSub acquires 3 new interpolation points to determine the other 3 undetermined coefficients of the 2D quadratic model—those coefficients not already fixed by the previous model.

We now discuss relevant properties of our method. Note that, in theory, for $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha^{(k)}\boldsymbol{d}_1^{(k)} + \beta^{(k)}\boldsymbol{d}_2^{(k)}$, the pair $\alpha^{(k)}$ and $\beta^{(k)}$ satisfies

$$(\alpha^{(k)},\beta^{(k)})^\top \in \left\{ \arg\min_{\alpha,\beta}\ Q_k(\alpha,\beta),\ \text{s. t. } \alpha^2+\beta^2\leq\Delta_k^2 \right\}.$$

We have the following proposition.

Proposition 4.7. *We have*

$$\begin{cases} \min\limits_{\alpha^2+\beta^2\leq\Delta_k^2}\ Q_k^+(\mathcal{T}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)},\boldsymbol{d}_*^{(k+1)}}(\boldsymbol{x}_k+\alpha\boldsymbol{d}_1^{(k)}+\beta\boldsymbol{d}_2^{(k)}))\leq f(\boldsymbol{x}_{k+1}), \\ \min\limits_{-\Delta_k\leq\alpha\leq\Delta_k}\ Q_{k+1}^{\text{sub}}(\hat{\mathcal{T}}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)}}(\boldsymbol{x}_k+\alpha\boldsymbol{d}_1^{(k+1)}))\leq f(\boldsymbol{x}_{k+1}), \\ \min\limits_{\alpha^2+\beta^2\leq\Delta_{k+1}^2}\ Q_{k+1}(\mathcal{T}^{(k+1)}_{\boldsymbol{d}_1^{(k+1)},\boldsymbol{d}_2^{(k+1)}}(\boldsymbol{x}_{k+1}+\alpha\boldsymbol{d}_1^{(k+1)}+\beta\boldsymbol{d}_2^{(k+1)}))\leq f(\boldsymbol{x}_{k+1}). \end{cases} \tag{4-22}$$

Proof. By definition, the proposition follows directly. □

Updating our new model in the manner described above has two advantages. One advantage is that the model Q_{k+1} sufficiently accounts for the behavior of Q_k along the one-dimensional subspace $\boldsymbol{x}_{k+1}+\text{span}\{\boldsymbol{d}_1^{(k+1)}\}$, because $\boldsymbol{x}_{k+1}$ itself has already been obtained as a successful step produced by Q_k. Another advantage is that minimizing

Q_{k+1} over the trust region yields iterates with nonincreasing model values according to (4-22).

Moreover, the quadratic model Q_k obtained by 2D-MoSub is precisely the solution of the subproblem

$$\begin{aligned} &\min_{Q\in\mathcal{Q}} \int_{-\infty}^{\infty} \left(Q(\alpha,0) - Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha \\ &\text{s. t. } Q(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{z})) = f(\boldsymbol{z}),\ \forall\, \boldsymbol{z} \in \{\boldsymbol{x}_k, \boldsymbol{y}_1^{(k)}, \boldsymbol{y}_2^{(k)}, \boldsymbol{y}_3^{(k)}\} \end{aligned} \tag{4-23}$$

in the sense that Q_k coincides with Q_{k-1}^{+} along the direction $\boldsymbol{d}_1^{(k)}$, which is in fact a numerically approximate gradient-descent direction.

We next establish the convexity of subproblem (4-23).

Theorem 4.8. *For any quadratic function Q satisfying the interpolation conditions in subproblem (4-23), the subproblem (4-23) is strictly convex.*

Proof. For $0 < c < 1$ and two distinct 2D quadratic functions Q_a and Q_b that satisfy the interpolation conditions in (4-23), we have

$$\begin{aligned} &\int_{-\infty}^{\infty} \left(cQ_a(\alpha,0) + (1-c)Q_b(\alpha,0) - Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha \\ <\, &c\int_{-\infty}^{\infty} \left(Q_a(\alpha,0) - Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha + (1-c)\int_{-\infty}^{\infty} \left(Q_b(\alpha,0) - Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha. \end{aligned} \tag{4-24}$$

In fact, the difference between the right-hand side and the left-hand side of (4-24) is

$$\begin{aligned} &-2c(1-c)\int_{-\infty}^{\infty} Q_a(\alpha,0)Q_b(\alpha,0)d\alpha + (c-c^2)\int_{-\infty}^{\infty} (Q_a(\alpha,0))^2 d\alpha \\ &+ (1-c-(1-c)^2)\int_{-\infty}^{\infty} (Q_b(\alpha,0))^2 d\alpha \\ =\, &(c-c^2)\int_{-\infty}^{\infty} (Q_a(\alpha,0) - Q_b(\alpha,0))^2 d\alpha < 0, \end{aligned}$$

because $0 < c < 1$ and $Q_a(\alpha,0) \neq Q_b(\alpha,0)$ (which follows from $Q_a \neq Q_b$ together with the interpolation conditions they satisfy). Therefore, we obtain the strict convexity of the objective of the subproblem. □

The above theorem shows that the model Q_k obtained by 2D-MoSub is indeed the unique solution of subproblem (4-23).

The above discussion highlights the advantages of the 2D-MoSub algorithm when $\boldsymbol{d}_1^{(k)}$ is an approximate gradient descent direction. We can also establish the following result.

Theorem 4.9. *If Q_k is the solution of subproblem (4-23), then for a quadratic function f, we have*

$$\begin{aligned}\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-\tilde{f}(\alpha,0)\right)^2 d\alpha = &\int_{-\infty}^{\infty}\left(Q_{k-1}^{+}(\alpha,0)-\tilde{f}(\alpha,0)\right)^2 d\alpha \\ &-\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha,\end{aligned} \tag{4-25}$$

where $\tilde{f} = f \circ (\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}})^{-1}$.

Proof. Let $Q_t = Q_k + t\left(Q_k - \tilde{f}\right)$, where $t \in \mathfrak{R}$. Then Q_t is a quadratic function that satisfies the interpolation conditions of subproblem (4-23). By the optimality of Q_k, the quadratic function

$$\varphi(t) := \int_{-\infty}^{\infty}\left(Q_t(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha$$

attains its minimum at $t = 0$. Expanding $\varphi(t)$, we obtain

$$\begin{aligned}\varphi(t) = t^2 &\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-\tilde{f}(\alpha,0)\right)^2 d\alpha \\ &+ 2t\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-\tilde{f}(\alpha,0)\right)\left(Q_k(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right) d\alpha \\ &+ \int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha,\end{aligned}$$

which implies

$$\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-\tilde{f}(\alpha,0)\right)\left(Q_k(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right) d\alpha = 0.$$

Considering $\varphi(-1)$, the conclusion follows. □

Moreover, the following corollary holds.

Corollary 4.10. *If Q_k is the solution of subproblem (4-23), then for a quadratic function f, we have*

$$\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-\tilde{f}(\alpha,0)\right)^2 d\alpha \le \int_{-\infty}^{\infty}\left(Q_{k-1}^{+}(\alpha,0)-\tilde{f}(\alpha,0)\right)^2 d\alpha, \tag{4-26}$$

where $\tilde{f} = f \circ (\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}})^{-1}$.

Proof. From equation (4-25) and the inequality

$$\int_{-\infty}^{\infty}\left(Q_k(\alpha,0)-Q_{k-1}^{+}(\alpha,0)\right)^2 d\alpha \ge 0,$$

inequality (4-26) immediately follows. □

The above results indicate that the model Q_k generated by our 2D-MoSub algorithm provides, along direction $\boldsymbol{d}_1^{(k)}$, a better approximation to the objective function than the corrected model Q_{k-1}^{+} from the previous step, in a certain sense.

Next, we present the function value decrease behavior of 2D-MoSub. We have the following proposition.

Proposition 4.11. *The iterates produced by 2D-MoSub yield nonincreasing objective values, i.e.,*

$$f(\boldsymbol{x}_{k+1}) \leq f(\boldsymbol{x}_k),$$

and on successful steps,

$$f(\boldsymbol{x}_{k+1}) \leq f(\boldsymbol{x}_k) - \eta \left(Q_k(0,0) - Q_k(\mathcal{T}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}(\boldsymbol{x}_{k+1})) \right)$$

holds.

Proof. The result follows directly from the algorithmic framework and the criterion associated with (4-1). □

The following result establishes the convergence of 2D-MoSub. Before stating it, we present the basic assumptions.

Assumption 4.12. The objective function f is bounded below, twice continuously differentiable, and has bounded second derivatives. Let C denote an upper bound of $\left\|\nabla^2 f\right\|_2$. There exists an infinite index set $\mathcal{I} \subseteq \mathbb{N}^+$ such that

(1) There exists $\varepsilon_1 > 0$ such that $\left\|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\right\|_2 \geq \varepsilon_1 \left\|\nabla f(\boldsymbol{x}_k)\right\|_2$ holds for $k \in \mathcal{I}$, where $\boldsymbol{P}_k$ is the orthogonal projector from $\mathfrak{R}^n$ onto the two-dimensional subspace $\mathcal{S}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}$;

(2) $f(\boldsymbol{x}_{k+1}) - \inf_{\boldsymbol{d} \in \mathcal{S}^{(k)}_{\boldsymbol{d}_1^{(k)},\boldsymbol{d}_2^{(k)}}} f(\boldsymbol{x}_k + \boldsymbol{d}) \to 0$ as $k \in \mathcal{K}$ and $k \to \infty$.

The convergence theorem below for our method follows and extends Theorem 5.7 in Zhang [46].

Theorem 4.13. *If the objective function f and 2D-MoSub satisfy Assumption 4.12, then*

$$\liminf_{k \to \infty} \left\|\nabla f(\boldsymbol{x}_k)\right\|_2 = 0,$$

where each $\boldsymbol{x}_k$ is the iterate generated by 2D-MoSub.

Proof. The proof here follows and extends the proof of Theorem 5.7 in Zhang [46]. In fact, we aim to prove that when $k \in \mathcal{I}$ and $k \to \infty$, it holds that $\left\|\nabla f(\boldsymbol{x}_k)\right\|_2 \to 0$. We proceed by contradiction. Suppose the conclusion does not hold. Then there exist $\varepsilon_2 > 0$ and an infinite subset $\mathcal{I}_{\text{sub}}$ of $\mathcal{I}$ such that: for all $k \in \mathcal{I}_{\text{sub}}$, $\left\|\nabla f(\boldsymbol{x}_k)\right\|_2 \geq \varepsilon_2$. By Assumption 4.12, without loss of generality, we may assume that for all $k \in \mathcal{I}_{\text{sub}}$,

$$\left\|\boldsymbol{P}_k \nabla f(\boldsymbol{x}_k)\right\|_2 \geq \varepsilon_1 \left\|\nabla f(\boldsymbol{x}_k)\right\|_2,$$

and for any $k \in \mathcal{I}_{\text{sub}}$,

$$f(\boldsymbol{x}_{k+1}) - \inf_{x \in S^{(k)}_{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}}} f(\boldsymbol{x}_k + \boldsymbol{d}) \leq \frac{\epsilon_1^2 \epsilon_2^2}{4C}. \tag{4-27}$$

Based on Lemma 5.5 in Zhang [46], we have

$$f(\boldsymbol{x}_k) - \inf_{\boldsymbol{d} \in S^{(k)}_{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}}} f(\boldsymbol{x}_k + \boldsymbol{d}) \geq \frac{1}{2C} \left\| \boldsymbol{P}_k \nabla f(\boldsymbol{x}_k) \right\|_2^2. \tag{4-28}$$

Subtracting $\inf_{\boldsymbol{d} \in S^{(k)}_{\boldsymbol{d}_1^{(k)}, \boldsymbol{d}_2^{(k)}}} f(\boldsymbol{x}_k + \boldsymbol{d})$ from $f(\boldsymbol{x}_k)$ and $f(\boldsymbol{x}_{k+1})$ respectively and then taking the difference, together with (4-27) and (4-28), yields

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1}) \geq \frac{1}{2C} \left\| \boldsymbol{P}_k \nabla f(\boldsymbol{x}_k) \right\|_2^2 - \frac{\epsilon_1^2 \epsilon_2^2}{4C} \geq \frac{\epsilon_1^2 \epsilon_2^2}{2C} - \frac{\epsilon_1^2 \epsilon_2^2}{4C} = \frac{\epsilon_1^2 \epsilon_2^2}{4C}.$$

This contradicts the facts that $\mathcal{I}_{\text{sub}}$ is an infinite subset and f is bounded below. □

Note that Assumption 4.12 is a sufficient condition for the convergence of our 2D-MoSub method. Below we state, without proof, the result of Lemma 5.5 from Zhang [46].

Proposition 4.14. *Assume the objective function f is bounded below, twice continuously differentiable, and has bounded second derivatives. Let C denote an upper bound of $\left\| \nabla^2 f \right\|$, and let S be a subspace of $\mathfrak{R}^n$. Then*

$$f(\boldsymbol{x}) - \inf_{\boldsymbol{d} \in S} f(\boldsymbol{x} + \boldsymbol{d}) \geq \frac{1}{2C} \left\| \boldsymbol{P} \nabla f(\boldsymbol{x}) \right\|_2^2,$$

where $\boldsymbol{P}$ is the orthogonal projector from $\mathfrak{R}^n$ onto S.

4.1.4 Numerical Results

We now present some experimental results to demonstrate the performance of the 2D-MoSub algorithm on the tested optimization problems. Table 4-2 shows the parameter settings used when testing our algorithm.

To illustrate the general numerical performance of our subspace method, we tested several classical benchmark problems and used Performance Profiles and Data Profiles to present the results. The test problems are listed in Table 4-3, and are drawn from classical collections of unconstrained optimization test functions, including CUTEr and CUTEst [177, 208], among others.

The problem dimensions range from 10 to 20000. Moreover, all algorithms start from the same initial point $\boldsymbol{x}_{\text{int}}$, and the accuracy tolerance τ in the Profile plots is set to 10^{-1}, 10^{-3}, and 10^{-5}, respectively. We compared 2D-MoSub with Nelder-Mead

Table 4-2 2D-MoSub Paremeters

Parameter	Value	Description
Δ_1	1	Initial trust-region radius
Δ_{low}	1×10^{-4}	Lower bound of trust-region radius
Δ_{upper}	1×10^{4}	Upper bound of trust-region radius
γ_1	10	Trust-region expansion factor
γ_2	0.1	Trust-region contraction factor
η	0.2	Threshold for successful step
η_0	0.1	Threshold for modified successful step
$\boldsymbol{d}^{(1)}$	$(1, 0, \cdots, 0)$	Initial direction

[191], NEWUOA [94], DFBGN [141], and CMA-ES [131]. Note that this selection of comparison algorithms is motivated by the fact that, in addition to the two classical derivative-free optimization methods Nelder-Mead and NEWUOA, the algorithms DFBGN and CMA-ES incorporate techniques designed for large-scale derivative-free optimization.

From Figures 4-4 and 4-5, we observe that 2D-MoSub is able to solve most problems better than the other algorithms considered. It should be noted that the test set includes both easy and difficult problems (depending on dimension, starting point, and function structure). For simple problems, all methods achieve solutions within relatively few evaluations, and the differences are not significant (as seen in the early rise of the Data Profile curves, which reflects specially selected simple problems). However, for more difficult problems, the differences in performance become much more pronounced.

4.1.5 Summary

In this section, we proposed a new large-scale derivative-free optimization method, 2D-MoSub, based on trust-region techniques and subspace strategies. Our method generates iterations by solving two-dimensional trust-region subproblems. In addition, we defined the two-dimensional subspace Λ-poisedness for interpolation sets at iteration k with three known coefficients. We presented the main steps of the algorithm and analyzed its theoretical properties. Numerical results demonstrated the advantages of applying 2D-MoSub to derivative-free optimization problems. Future work includes designing new strategies for subspace selection and extending the method to large-scale constrained derivative-free optimization problems.

Table 4-3 Test problems for Figure 4-4 and Figure 4-5

ARGLINA	ARGLINA4	ARGLINB	ARGLINC	ARGTRIG
ARWHEAD	BDQRTIC	BDQRTICP	BDVALUE	BROWNAL
BROYDN3D	BROYDN7D	BRYBND	CHAINWOO	CHEBQUAD
CHNROSNB	CHPOWELLB	CHPOWELLS	CHROSEN	COSINE
CRAGGLVY	CUBE	CURLY10	CURLY20	CURLY30
DIXMAANE	DIXMAANF	DIXMAANG	DIXMAANH	DIXMAANI
DIXMAANJ	DIXMAANK	DIXMAANL	DIXMAANM	DIXMAANN
DIXMAANO	DIXMAANP	DQRTIC	EDENSCH	ENGVAL1
ERRINROS	EXPSUM	EXTROSNB	EXTTET	FIROSE
FLETCBV2	FLETCBV3	FLETCHCR	FREUROTH	GENBROWN
GENHUMPS	GENROSE	INDEF	INTEGREQ	LIARWHD
LILIFUN3	LILIFUN4	MOREBV	MOREBVL	NCB20
NCB20B	NONCVXU2	NONCVXUN	NONDIA	NONDQUAR
PENALTY1	PENALTY2	PENALTY3	PENALTY3P	POWELLSG
POWER	ROSENBROCK	SBRYBND	SBRYBNDL	SCHMVETT
SCOSINE	SCOSINEL	SEROSE	SINQUAD	SPARSINE
SPARSQUR	SPMSRTLS	SROSENBR	STMOD	TOINTGSS
TOINTTRIG	TQUARTIC	TRIGSABS	TRIGSSQS	TRIROSE1
TRIROSE2	VARDIM	WOODS	-	-

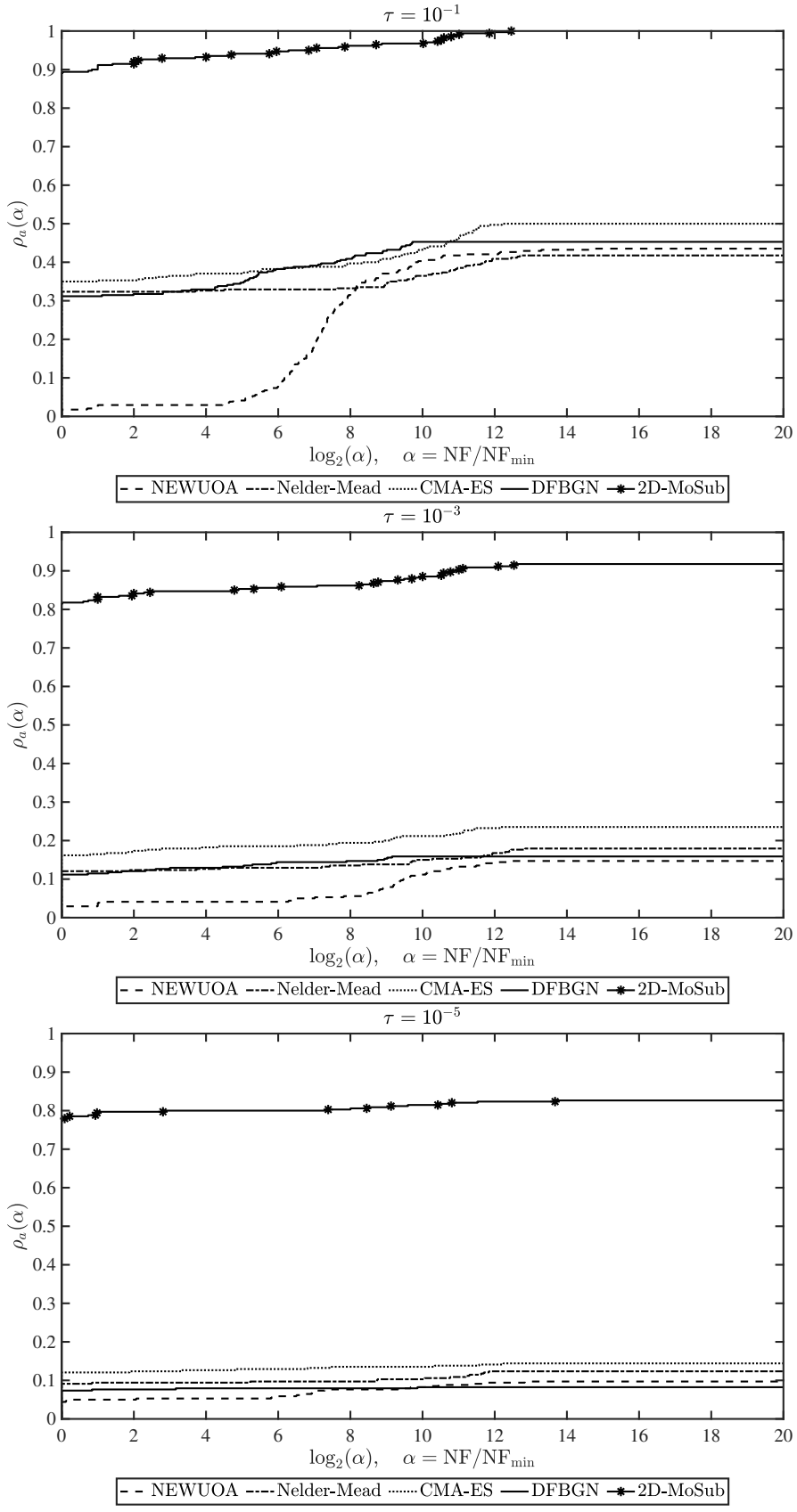


Figure 4-4 Performance Profile of solving test large-scale problems

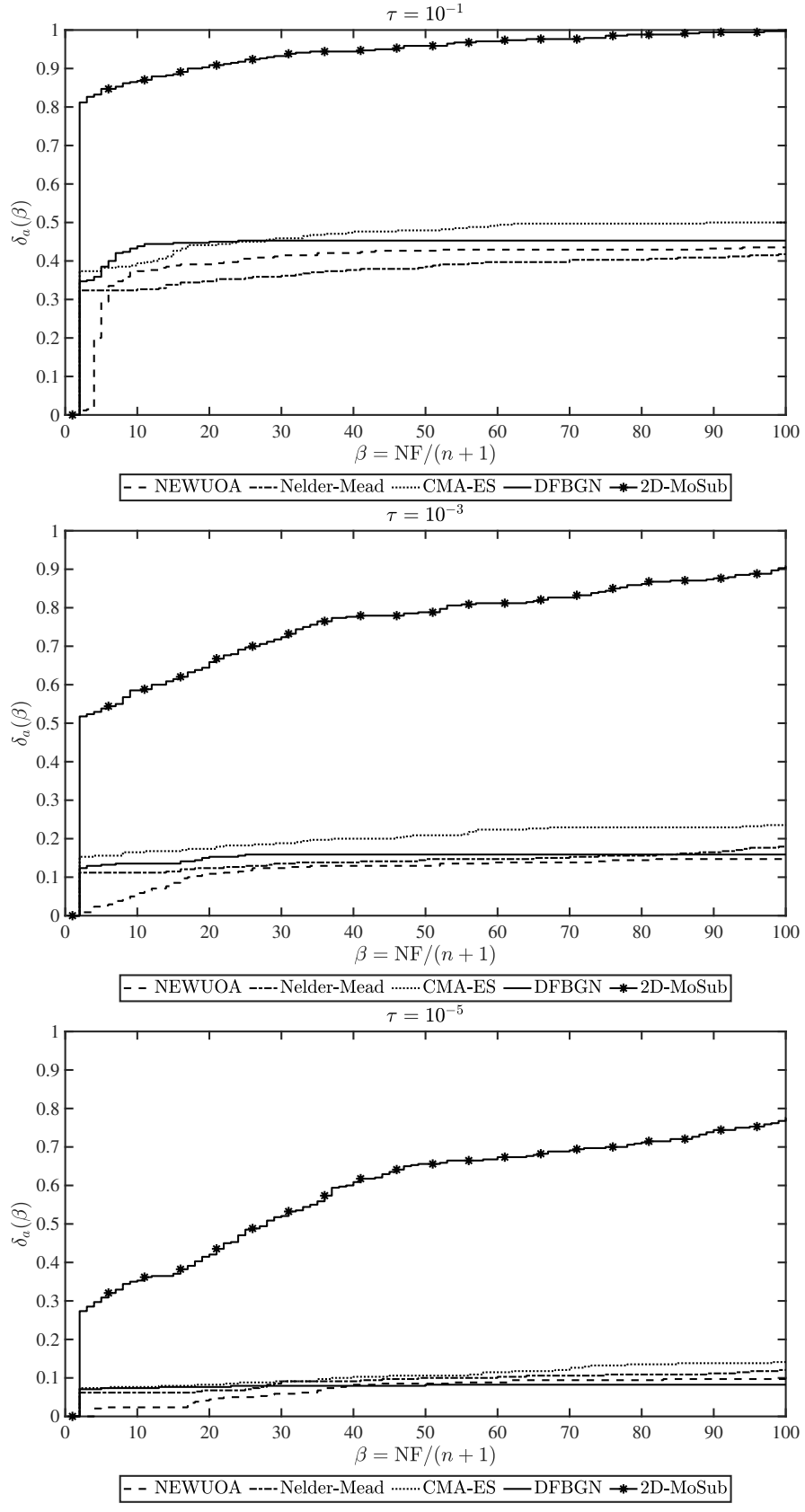


Figure 4-5 Data Profile of solving test large-scale problems

4.2 A Derivative-Free Optimization Algorithm Combining Line Search and Trust-Region Techniques

The speed-up slow-down (SUSD) direction is a new search direction that has been shown, under certain conditions, to converge to the gradient descent direction. In this section, we propose a derivative-free optimization algorithm, called SUSD-TR, which combines the SUSD direction (based on the covariance matrix of interpolation points) with the trust-region subproblem solution of the interpolation model function at the current iteration. We analyze the optimization dynamical system of SUSD-TR and the stability of its search directions, and we describe the trial step and structure step in detail. Numerical results demonstrate the advantages of our algorithm and show that SUSD-TR significantly outperforms methods that rely solely on searching along the SUSD direction. Compared with state-of-the-art derivative-free optimization algorithms, our algorithm is competitive.

4.2.1 Background and Motivation

Consider the unconstrained optimization problem

$$\min_{\boldsymbol{x}\in\mathfrak{R}^n} f(\boldsymbol{x}), \tag{4-29}$$

where $f : \mathfrak{R}^n \to \mathfrak{R}$ is the objective function. As discussed earlier, no derivative information of f is available. It is well known that both line search methods and trust-region methods are widely used for solving optimization problems. Line search methods seek an (optimal) step size along the current search direction (for SUSD-based algorithms, this refers to the step size for each point in the group). However, when the search direction is inefficient, line search methods may lead to slow convergence. In contrast, trust-region methods construct a local quadratic model function and seek its minimizer within a trust region, which can mitigate the inefficiency of poor search directions at the expense of higher computational cost. Nocedal and Yuan [209] discussed how to combine line search and trust-region methods in gradient-based optimization to better adapt to different search directions.

For a group of trial points moving simultaneously along the common SUSD direction [117], we refer to the iteration process as a large-scale step. In practice, the internal structure of the iterates (especially in large-scale steps) may not be well improved. In brief, algorithms using the SUSD direction typically probe a set of points simultaneously, then determine a common SUSD direction based on their distribution, and finally update each point by moving along this direction with step sizes determined by their function values.

Our idea is to modify this group of trial points before continuing along the SUSD direction by introducing a new point inside the trust region and discarding one point.

This new point is obtained as a minimizer of the local interpolation model within the trust region. It can adjust or even reverse the search direction of the group, particularly when the direction has deviated significantly. Thus, in addition to large-scale moves, we incorporate trust-region model-based corrections, combining line search and trust-region techniques.

The remainder of this section is organized as follows. Section 4.2.2 presents the SUSD-TR algorithm that combines the SUSD direction with trust-region interpolation. Section 4.2.3 analyzes the dynamical system of SUSD-TR and the stability of its search directions. Section 4.2.4 describes the trial step and structure step in more detail.

4.2.2 Combination of SUSD Direction and Trust-Region Interpolation

Our algorithm mainly updates iterates using two steps: a trust-region step and a line search step. The trust-region step is obtained by solving a trust-region subproblem of the interpolation model function at each iteration. The line search step moves the search points along the SUSD direction, denoted by $\boldsymbol{v}_1$. Here, the term search points refers to the set of trial points at the current iteration.

Suppose we have m search points, each being a candidate solution $\boldsymbol{x}_i \in \mathfrak{R}^n$, $i = 1, \cdots, m$, with $m \geq n$, where n is the dimension of problem (4-29). We define the covariance matrix $\boldsymbol{C} \in \mathfrak{R}^{n\times n}$ as

$$\boldsymbol{C} = \sum_{i=1}^{m} \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^{\top}, \tag{4-30}$$

where $\boldsymbol{x}_c = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$ is the center of the search points. Let $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n$ denote the unit eigenvectors of $\boldsymbol{C}$, corresponding to the eigenvalues $\mu_1, \cdots, \mu_n$, sorted from the smallest μ_1 to the largest μ_n (with $\mu_1 \neq 0$). The vector $\boldsymbol{v}_1$ is the SUSD direction [117]. Algorithm 12 presents the SUSD-TR algorithm, which combines the SUSD direction with interpolation and trust-region techniques. In Algorithm 12, $\bar{f}^{(k)}$ denotes the minimum function value among the interpolation (search) points at iteration k. The index k represents the iteration counter, and some parameters are expressed as functions of k.

Algorithm 12 SUSD-TR Algorithm

Input: Number of search points m, initial points $\boldsymbol{x}_i^{(0)}, i = 1, \cdots, m$, parameters β, Δ_0, termination parameters P, ε; $k = 0$

while $|\bar{f}^{(k)} - \frac{1}{P}\sum_{h=1}^{P} \bar{f}^{(k-h)}| > \varepsilon$ **do**

Compute $\boldsymbol{C}^{(k)}$ and $\boldsymbol{v}_1^{(k)}$ using standard PCA.

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

Evaluate $f_i^{(k)} := f(\boldsymbol{x}_i^{(k)})$.

end for

Compute $\Delta_k = \max_i(\|\boldsymbol{x}_i^{(k)} - \boldsymbol{x}_c^{(k)}\|_2)$.

if $\Delta_k > \kappa\Delta_{k-1}$ **then**

Structure step: Replace the point $\boldsymbol{x}_d^{(k)}$ farthest from the trust-region center with $\boldsymbol{x}_{\text{new}}^{(k)}$ (Algorithm 13).

else

Model improvement step: Use the model improvement step (Algorithm 6.3 in Conn, Scheinberg, and Vicente [20]) to check and enhance poisedness of the interpolation set.

Construct the current interpolation set $\mathcal{X}_k$ from the latest m search points, and build a linear interpolation model $L_k(\boldsymbol{x})$ (or an underdetermined quadratic interpolation model $Q_k(\boldsymbol{x})$) using (4-31) or (4-32).

Trial step: Solve the trust-region subproblem with truncated CG:

$$\begin{aligned} &\min_{\boldsymbol{x}} L_k(\boldsymbol{x}) \text{ or } Q_k(\boldsymbol{x}), \\ &\text{s. t. } \left\| \boldsymbol{x} - \boldsymbol{x}_c^{(k)} \right\|_2 \leq \Delta_k, \end{aligned}$$

and replace $\boldsymbol{x}_d^{(k)}$ (the current worst function value point) with $\boldsymbol{x}_{\text{new}}^{(k)}$. Then update the trust-region radius Δ_k.

end if

Compute $\bar{f}^{(k)} := \min_i f_i^{(k)}$.

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

Compute $\alpha(\boldsymbol{x}_i^{(k)}) = \beta[1 - \exp(\bar{f} - f(\boldsymbol{x}_i^{(k)}))]$.

Line search step: Update $\boldsymbol{x}_i^{(k+1)} = \boldsymbol{x}_i^{(k)} + \alpha(\boldsymbol{x}_i^{(k)})\boldsymbol{v}_1^{(k)}$.

end for

Set $k = k + 1$.

end while

Output: $\boldsymbol{x}^* = \boldsymbol{x}_i$, where $\boldsymbol{x}_i$ is the point with the smallest function value in the final iteration.

It can be seen that SUSD-TR is a derivative-free optimization algorithm that combines line search and trust-region techniques. The group of search points moves along the direction $\boldsymbol{v}_1$, which belongs to the line-search type, and then the algorithm solves a trust-region subproblem for correction, thereby forming the iterative cycle.

SUSD-TR has several advantages. First, if data (including function values) are transferred between different computing nodes, the algorithm can be executed in a distributed or parallelized manner. This is because in the line-search step, function evaluations at the m search points can be carried out simultaneously, significantly reducing the evaluation cost, especially for expensive-to-evaluate problems. Second, Algorithm 12 does not rely on traditional gradient estimation, and therefore can operate without explicit gradient information. Third, the optimization process can be expressed as a dynamical

system, and by exploiting the continuous-time formulation, we can derive theoretical results; this is uncommon in derivative-free optimization, but both novel and important. Last but not least, compared with traditional finite-difference estimates, the search points can be more flexibly distributed over the region of interest.

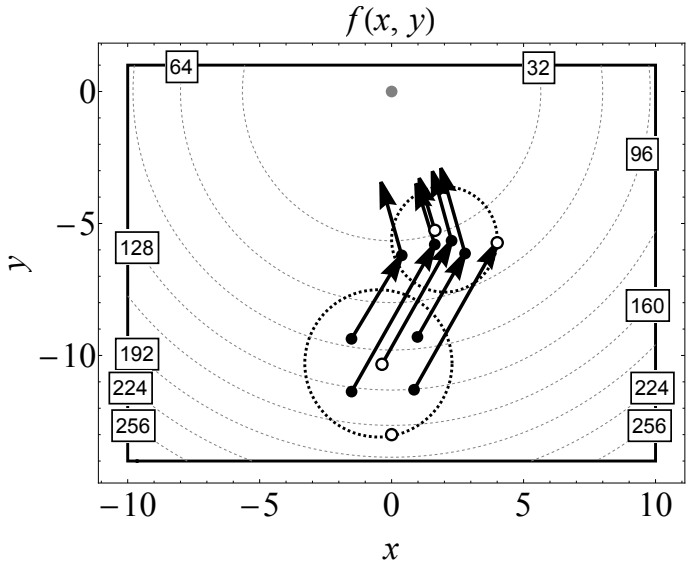


Figure 4-6 Illustration of the general framework of SUSD-TR for a 2-dimensional problem

Figure 4-6 shows the iteration process of the SUSD-TR algorithm, where an empty circle denotes the discarded point $\boldsymbol{x}_d$ at each iteration. At the k-th iteration, if the number of search (sample) points is less than $\frac{1}{2}(n+1)(n+2)$, the quadratic interpolation model Q_k can be obtained by solving the subproblem

$$\begin{aligned} &\min_{Q \in \mathcal{Q}} \ \left\| \nabla^2 Q - \nabla^2 Q_{k-1} \right\|_F^2, \\ &\text{s. t. } Q(\boldsymbol{x}_i) = f(\boldsymbol{x}_i), \ \forall \ \boldsymbol{x}_i \in \mathcal{X}_k. \end{aligned} \tag{4-31}$$

If a determined linear or quadratic interpolation model is available, then L_k or Q_k can be obtained by solving the system

$$\begin{aligned} f(\boldsymbol{x}_i) &= L_k(\boldsymbol{x}_i), \ \forall \ \boldsymbol{x}_i \in \mathcal{X}_k, \\ \text{or} \quad f(\boldsymbol{x}_i) &= Q_k(\boldsymbol{x}_i), \ \forall \ \boldsymbol{x}_i \in \mathcal{X}_k. \end{aligned} \tag{4-32}$$

In the numerical experiments, we present results using the underdetermined quadratic model in SUSD-TR. For simplicity, in the following discussion we may omit the iteration index (k).

4.2.3 Stability Analysis of Iteration Directions in SUSD-TR

The optimization process of Algorithm 12, viewed as a dynamical system (or gradient flow), can be written as

$$\begin{cases} \dot{\boldsymbol{x}}_i = \alpha(\boldsymbol{x}_i)\boldsymbol{v}_1, \quad i = 1, \cdots, d-1, d+1, \cdots, m, \\ \dot{\boldsymbol{x}}_d = \left(\alpha(\boldsymbol{x}_d) + \varepsilon_1\right)\left(\boldsymbol{v}_1 + \boldsymbol{\varepsilon}_2\right), \end{cases} \tag{4-33}$$

where $\varepsilon_1 \in \Re$ and $\boldsymbol{\varepsilon}_2 \in \Re^n$ are perturbation parameters for $\boldsymbol{x}_d$, indicating its update to $\boldsymbol{x}_{\text{new}}$ through trust-region techniques. The overdot in (4-33) denotes the derivative with respect to continuous time t (corresponding to iteration k). The step size $\alpha : \Re \to \Re$ is an exponential-type mapping [117]:

$$\alpha(\boldsymbol{x}_i) = \beta \left[1 - \exp\left(\bar{f} - f(\boldsymbol{x}_i)\right)\right], \quad i = 1, \cdots, m, \tag{4-34}$$

where $\beta > 0$ is a constant and $\bar{f}$ is the minimum function value among all search/interpolation points at the current iteration. For brevity, we denote $\alpha(\boldsymbol{x}_i)$ by α_i.

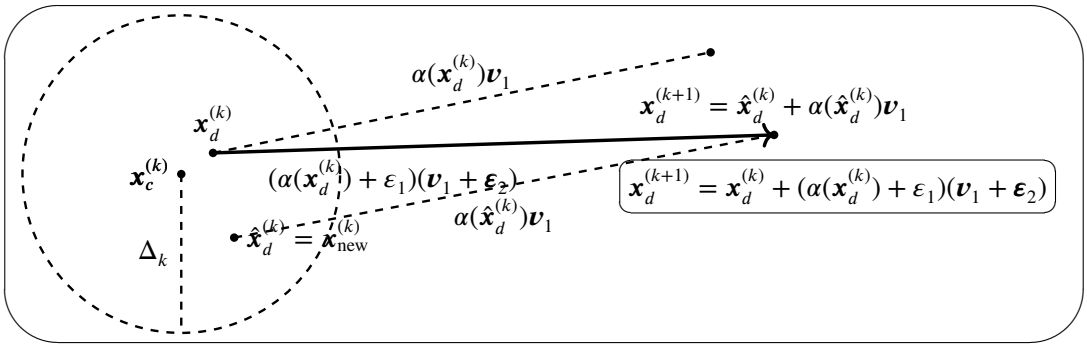


Figure 4-7 The disturbance in (4-33)

Figure 4-7 depicts the dynamical system (4-33) with discrete iterations (time), where $\boldsymbol{x}_d^{(k)}$ is replaced by $\hat{\boldsymbol{x}}_d^{(k)}$ in the trust-region step and then $\hat{\boldsymbol{x}}_d^{(k)}$ advances to $\boldsymbol{x}_d^{(k+1)}$ in the line-search step. Therefore, the new iterate can be written as

$$\boldsymbol{x}_d^{(k+1)} = \boldsymbol{x}_d^{(k)} + \left(\alpha(\boldsymbol{x}_d^{(k)}) + \varepsilon_1\right)\left(\boldsymbol{v}_1 + \boldsymbol{\varepsilon}_2\right),$$

where ε_1 denotes the perturbation of the step size and $\boldsymbol{\varepsilon}_2$ denotes the directional perturbation of $\boldsymbol{x}_d^{(k)}$. This corresponds to the dynamical system (4-33). Next, we carry out the analysis using the continuous dynamical system.

Remark 4.4. For the exponential step-size mapping (4-34), the next-step step size of the point with the smallest function value in the current iteration is zero, which differs from the case of a linear step-size mapping. Note that Figure 4-6 is intended to illustrate a generic example of the SUSD-TR framework.

Lemma 4.15. *The dynamical system for the SUSD-TR direction corresponding to (4-33) is*

$$\dot{\boldsymbol{v}}_1 = \left(\sum_{j=2}^{n} \frac{1}{\mu_1 - \mu_j} \boldsymbol{v}_j \boldsymbol{v}_j^{\top}\right) \left[\sum_{i=1}^{m} \left(\alpha_i - \alpha_a\right)\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right) + \varepsilon_1 \left(\boldsymbol{x}_d - \boldsymbol{x}_c\right) + \Phi \boldsymbol{v}_1\right], \tag{4-35}$$

where $\boldsymbol{v}_j$ is the j-th unit eigenvector of the matrix $\boldsymbol{C}$,

$$\alpha_a = \frac{1}{m} \sum_{i=1}^{m} \alpha_i + \frac{\varepsilon_1}{m},$$

where $\alpha_i = \alpha(\boldsymbol{x}_i)$, *and*

$$\Phi = \left(\alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)\left(\alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)^\top.$$

Proof. Note that $\boldsymbol{x}_c = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$. From the dynamical system (4-33), we obtain

$$\dot{\boldsymbol{x}}_c = \alpha_a \boldsymbol{v}_1 + \frac{1}{m}\left(\alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right),$$

where $\alpha_a = \frac{1}{m}\sum_{i=1}^{m} \alpha_i + \frac{\varepsilon_1}{m}$. Differentiating (4-30) with respect to time yields

$$\begin{aligned}
\dot{\boldsymbol{C}} = &\sum_{i=1}^{m}\left(\alpha_i - \alpha_a\right)\left[\boldsymbol{v}_1\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\boldsymbol{v}_1^\top\right] \\
&- \frac{\alpha_d + \varepsilon_1}{m}\sum_{i=1}^{m}\left[\boldsymbol{\varepsilon}_2\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\boldsymbol{\varepsilon}_2^\top\right] \\
&+ \left(\varepsilon_1 \boldsymbol{v}_1 + \alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)\left(\varepsilon_1 \boldsymbol{v}_1 + \alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)^\top \\
= &\sum_{i=1}^{m}\left(\alpha_i - \alpha_a\right)\left[\boldsymbol{v}_1\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\boldsymbol{v}_1^\top\right] \\
&+ \left(\varepsilon_1 \boldsymbol{v}_1 + \alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)^\top + \left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)\left(\varepsilon_1 \boldsymbol{v}_1 + \alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2\right)^\top.
\end{aligned} \tag{4-36}$$

Moreover, using $\boldsymbol{C}\boldsymbol{v}_1 = \mu_1 \boldsymbol{v}_1$, we have

$$\dot{\boldsymbol{C}}\boldsymbol{v}_1 + \boldsymbol{C}\dot{\boldsymbol{v}}_1 = \dot{\mu}_1 \boldsymbol{v}_1 + \mu_1 \dot{\boldsymbol{v}}_1.$$

Hence

$$\boldsymbol{v}_j^\top \dot{\boldsymbol{C}}\boldsymbol{v}_1 + \boldsymbol{v}_j^\top \boldsymbol{C}\dot{\boldsymbol{v}}_1 = \dot{\mu}_1 \boldsymbol{v}_j^\top \boldsymbol{v}_1 + \mu_1 \boldsymbol{v}_j^\top \dot{\boldsymbol{v}}_1, \tag{4-37}$$

and the matrix $\dot{\boldsymbol{C}}$ is symmetric. This implies

$$\boldsymbol{v}_j^\top \boldsymbol{C}\dot{\boldsymbol{v}}_1 = \left(\boldsymbol{C}\boldsymbol{v}_j\right)^\top \dot{\boldsymbol{v}}_1 = \mu_j \boldsymbol{v}_j^\top \dot{\boldsymbol{v}}_1.$$

Since $\boldsymbol{v}_j^\top \boldsymbol{v}_1 = \boldsymbol{v}_1^\top \boldsymbol{v}_j = 0$, from (4-37) we obtain

$$\boldsymbol{v}_j^\top \dot{\boldsymbol{v}}_1 = \frac{1}{\mu_1 - \mu_j}\boldsymbol{v}_j^\top \dot{\boldsymbol{C}}\boldsymbol{v}_1. \tag{4-38}$$

Because the matrix $\boldsymbol{C}$ is symmetric, we have

$$\dot{\boldsymbol{v}}_1 = \sum_{j=2}^{n} \boldsymbol{v}_j^\top \dot{\boldsymbol{v}}_1 \boldsymbol{v}_j. \tag{4-39}$$

Substituting (4-36) into (4-38), and using (4-39) together with $\boldsymbol{v}_j^\top \boldsymbol{v}_1 (\boldsymbol{x}_i - \boldsymbol{x}_c)^\top \boldsymbol{v}_1 = 0$, the lemma follows. □

Let $\alpha_c = \alpha(f(\boldsymbol{x}_c))$ and define the gradient $\nabla\alpha = \nabla\alpha(\boldsymbol{x}_c)$. We approximate $\alpha_i = \alpha(\boldsymbol{x}_i)$ by a Taylor expansion at the center $\boldsymbol{x}_c$, namely

$$\alpha_i - \alpha_c = \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top \nabla\alpha + r_i, \tag{4-40}$$

where $\alpha_c = \alpha(\boldsymbol{x}_c)$ and $r_i = \mathcal{O}\left(\|\boldsymbol{x}_i - \boldsymbol{x}_c\|_2^2\right)$. Suppose $f_c = f(\boldsymbol{x}_c)$. Let $\nabla f = \nabla f(\boldsymbol{x}_c)$ denote the gradient of f at the center $\boldsymbol{x}_c$. We obtain the following lemma.

Lemma 4.16. *Based on (4-33) and the Taylor expansion, we have*

$$\begin{aligned}\dot{\boldsymbol{v}}_1 = &\sum_{j=2}^{n} \frac{\mu_j}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\nabla\alpha + \boldsymbol{r} + \sum_{j=2}^{n} \frac{\varepsilon_1}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)\\ &+ \sum_{j=2}^{n} \frac{1}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\left(\Phi\boldsymbol{v}_1\right),\end{aligned} \tag{4-41}$$

where

$$\boldsymbol{r} = \left(\sum_{j=2}^{n} \frac{1}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\right)\left[\sum_{i=1}^{m} r_i\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\right].$$

Proof. Let $r_a = \frac{1}{m}\sum\limits_{i=1}^{m} r_i$. The definition of α_a together with (4-40) implies $\alpha_a = \alpha_c + r_a + \frac{\varepsilon_1}{m}$. Hence, from (4-40) we obtain

$$\sum_{i=1}^{m}\left(\alpha_i - \alpha_a\right)\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right) = \boldsymbol{C}\nabla\alpha + \sum_{i=1}^{m} r_i\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right). \tag{4-42}$$

Substituting (4-42) into (4-35) and using $\boldsymbol{v}_j^\top\boldsymbol{C} = \mu_j\boldsymbol{v}_j^\top$, we obtain (4-41). □

We next present the following lemma, where we continue to use ∇f to denote $\nabla f(\boldsymbol{x}_c)$.

Lemma 4.17. *According to (4-33) and the exponential step size, the dynamical system is*

$$\begin{cases}\dot{f}_c = \left\{\dfrac{\varepsilon_1}{m} + \dfrac{\beta}{m}\displaystyle\sum_{i=1}^{m}\left[1 - \exp(\bar{f} - f(\boldsymbol{x}_i(t))\right]\right\}(\nabla f)^\top\boldsymbol{v}_1\\ \qquad + \left\{\dfrac{\beta}{m}\left[1 - \exp\left(\bar{f} - f(\boldsymbol{x}_d)\right)\right] + \dfrac{\varepsilon_1}{m}\right\}(\nabla f)^\top\boldsymbol{\varepsilon}_2,\\ \dot{\boldsymbol{v}}_1 = \beta\exp\left(\bar{f} - f_c\right)\displaystyle\sum_{j=2}^{n}\frac{\mu_j}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\nabla f + \boldsymbol{r} + \sum_{j=2}^{n}\frac{\varepsilon_1}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right)\\ \qquad + \displaystyle\sum_{j=2}^{n}\frac{1}{\mu_1 - \mu_j}\boldsymbol{v}_j\boldsymbol{v}_j^\top\left(\Phi\boldsymbol{v}_1\right).\end{cases} \tag{4-43}$$

Proof. From the calculations, we have

$$\begin{aligned}\dot{\boldsymbol{x}}_c &= \frac{1}{m}\sum_{i=1}^{m}\alpha_i \boldsymbol{v}_1 + \frac{1}{m}\left(\alpha_d \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{\varepsilon}_2 + \varepsilon_1 \boldsymbol{v}_1\right) \\ &= \left\{\frac{\varepsilon_1}{m} + \frac{\beta}{m}\sum_{i=1}^{m}\left[1 - \exp\left(\bar{f} - f(\boldsymbol{x}_i)\right)\right]\right\}\boldsymbol{v}_1 \\ &\quad + \left\{\frac{\beta}{m}\left[1 - \exp\left(\bar{f} - f(\boldsymbol{x}_d)\right)\right] + \frac{\varepsilon_1}{m}\right\}\boldsymbol{\varepsilon}_2.\end{aligned}$$

Substituting the above into $\dot{f}(\boldsymbol{x}_c) = \left(\nabla f(\boldsymbol{x}_c)\right)^{\top}\dot{\boldsymbol{x}}_c$, we obtain the first equation in (4-43). Moreover, we have

$$\nabla\alpha(\boldsymbol{x}_c) = \frac{d\alpha}{df}\nabla f(\boldsymbol{x}_c) = \beta\exp\left(\bar{f} - f(\boldsymbol{x}_c)\right)\nabla f(\boldsymbol{x}_c).$$

Hence, by substituting $\nabla\alpha(\boldsymbol{x}_c)$ into (4-41), we obtain the second equation in (4-43). □

The above establishes the dynamical system associated with our algorithm. We now analyze the stability of the SUSD-TR direction.

We first recall basic definitions from control and dynamical systems theory—stability, asymptotic stability, and input-to-state stability (ISS) [210]—which will be used in our stability analysis.

Definition 4.18 (Stability)**.** *If for any $\bar{\varepsilon} > 0$, there exists $\bar{\delta}(t, \bar{\varepsilon})$ such that when $|\eta(t_0)| < \bar{\delta}$, one has $|\eta(t)| < \bar{\varepsilon}$ for all $t > t_0$, then η is said to be stable.*

Definition 4.19 (Asymptotic stability)**.** *If there exists $\bar{\delta}(t_0)$ such that $|\eta(t_0)| < \bar{\delta}$ implies $\lim_{t\to\infty}\eta(t) = 0$, then η is said to be asymptotically stable.*

Definition 4.20 ($\mathcal{K}$-class function)**.** *A scalar continuous function $g_1(r)$ defined on $r \in [0, a)$ is said to be of class $\mathcal{K}$ if it is strictly increasing and satisfies $g_1(0) = 0$.*

Definition 4.21 ($\mathcal{KL}$-class function)**.** *A scalar continuous function $g_2(r, s)$ defined on $r \in [0, a)$ and $s \in [0, \infty)$ is said to be of class $\mathcal{KL}$ if, for each fixed s, the mapping $g_2(r, s)$ in r is of class $\mathcal{K}$, and for each fixed r, the mapping $g_2(r, s)$ in s is decreasing with $g_2(r, s) \to 0$ as $s \to \infty$.*

Definition 4.22 (Input-to-state stability)**.** *If there exist a $\mathcal{KL}$-class function f_1 and a $\mathcal{K}$-class function f_2 such that, for any initial state $\eta(t_0) \in [0, 2)$ and any bounded input $\delta(t)$ with $|\delta(t)| \le U$, the solution $\eta(t)$ of the system $\dot{\eta} = \psi(t, \eta, \delta)$ is defined for all $t > t_0$ and satisfies*

$$|\eta(t)| \le f_1\left(\left|\eta(t_0)\right|, t - t_0\right) + f_2\sup_{t_0\le\tau\le t}|\delta(\tau)|,$$

then the system is input-to-state stable with respect to the equilibrium $\eta^* = 0$, the neighborhood $\eta \in [0, 2)$, and the input bound U.

The following theorem will be used to establish the input-to-state stability property relevant to our algorithm.

Theorem 4.23 (Theorem 4.19 in Khalil [210]). *Let $\mathcal{V}(t,\eta) : [0,\infty)\times[0,2) \to \mathfrak{R}$ be a continuously differentiable function. Let $\alpha_1(\eta), \alpha_2(\eta)$ be $\mathcal{K}$-class functions on $[0,2)$, $\rho(|\delta|)$ a $\mathcal{K}$-class function on $[0,U]$, and $\alpha_3(\eta)$ a continuous positive function on $[0,2)$. Suppose that for all $(t,\eta,\delta) \in [0,\infty)\times[0,2)\times[-U,U]$, $\mathcal{V}$ satisfies*

$$\alpha_1(|\eta|) \le \mathcal{V}(t,\eta) \le \alpha_2(|\eta|)$$

and, whenever $|\eta| \ge \rho(|\delta|) > 0$,

$$\frac{\partial \mathcal{V}}{\partial t} + \frac{\partial \mathcal{V}}{\partial \eta}\psi(t,\eta,\delta) \le -\alpha_3(\eta).$$

Then the system $\dot{\eta} = \psi(t,\eta,\delta)$ is input-to-state stable.

Note that, for simplicity, the symbols in the above definitions and theorem are independent of what follows. Let $\boldsymbol{g} = \nabla f/\|\nabla f\|_2$. We now show that the SUSD-TR direction $\boldsymbol{v}_1$ tends to $-\boldsymbol{g}$ under certain conditions. Define $\eta = 1 + \boldsymbol{v}_1^\top \boldsymbol{g}$, where $\eta = 0$ if and only if $\boldsymbol{v}_1 = -\boldsymbol{g}$. We can then derive the following result for η.

Corollary 4.24. *Based on (4-33) and the exponential-type step size, we obtain the dynamical system for the search iteration*

$$\dot{\eta} = \beta \exp\left(\bar{f} - f_c\right)\|\nabla f\|_2 \sum_{j=2}^{n} \frac{\mu_j}{\mu_1 - \mu_j}\left(\boldsymbol{g}^\top \boldsymbol{v}_j\right)^2 + \delta := \psi(t,\eta,\delta), \tag{4-44}$$

where

$$\delta = \boldsymbol{r}^\top \boldsymbol{g} + \boldsymbol{g}^\top \left(\sum_{j=2}^{n} \frac{\boldsymbol{v}_j \boldsymbol{v}_j^\top}{\mu_1 - \mu_j}\right)\left[\varepsilon_1\left(\boldsymbol{x}_d - \boldsymbol{x}_c\right) + \Phi \boldsymbol{v}_1\right] + \boldsymbol{v}_1^\top \dot{\boldsymbol{g}}.$$

Proof. By computing $\dot{\boldsymbol{v}}_1^\top \boldsymbol{g}$, (4-44) follows directly from (4-43). □

The parameter δ represents the external disturbance to η caused by the nonlinearity of the function (which cannot be controlled by the search points), and it includes higher-order terms. Note that when $\varepsilon_1 = 0$ and $\boldsymbol{\varepsilon}_2 = \boldsymbol{0}$, we have $\delta = \boldsymbol{r}^\top \boldsymbol{g} + \boldsymbol{v}_1^\top \dot{\boldsymbol{g}}$, which corresponds exactly to the SUSD algorithm, i.e., the method that advances solely along the SUSD direction without the trust-region iteration step in SUSD-TR.

The following result provides further details on when and how $\boldsymbol{v}_1$ tends to $-\boldsymbol{g}$.

Theorem 4.25. *Assume that $\|\nabla f(\boldsymbol{x}_c)\|_2 > \xi$, where ξ is a positive constant. Then, for (4-44), the system $\dot{\eta} = \psi(t,\eta,0)$ is asymptotically stable at the equilibrium $\eta = 0$, i.e., when $\eta(0) \in [0,2)$, one has $\eta(t) \to 0$ as $t \to \infty$, where t denotes the continuous iteration (time). Moreover, for disturbances satisfying $|\delta| < \beta\exp(\bar{f} - f_c) M \frac{\mu_1}{\mu_n - \mu_1}\xi$ with $M \in (0,1)$, the system $\psi(t,\eta,\delta)$ is locally input-to-state stable at the equilibrium $\eta = 0$.*

Proof. The proof is identical to that of Theorem 1 in the work of Al-Abri et al. [117], and is therefore omitted here. □

4.2.4 Trial Step and Structure Step

In this subsection, we present more details of the trial step and the structure step contained in the trust-region step, which are part of the implementation of Algorithm 12.

The trial step can be viewed as a small-scale correction within SUSD-TR. In this step, the algorithm obtains a new point by solving an interpolation-model subproblem inside the trust region, namely

$$\begin{aligned} &\min_{\boldsymbol{x}} L_k(\boldsymbol{x}) \text{ or } Q_k(\boldsymbol{x}) \\ &\text{s. t. } \left\| \boldsymbol{x} - \boldsymbol{x}_c^{(k)} \right\|_2 \le \Delta_k, \end{aligned}$$

and replaces the point with the largest function value among the current iteration points.

In the theoretical analysis, the direction $\boldsymbol{v}_1$ may fail to converge or may even become unstable. In such cases, we say that $\boldsymbol{v}_1$ fails. The following proposition illustrates the advantage of introducing the trial step in the SUSD-TR algorithm when $\boldsymbol{v}_1$ turns toward the gradient ascent direction $\boldsymbol{g}$.

Proposition 4.26. *Suppose that, at a given iteration, the m search points satisfy the dynamical system*

$$\begin{cases} \dot{\boldsymbol{x}}_i = \boldsymbol{g}, \quad i = 1, \cdots, d-1, d+1, \cdots, m, \\ \dot{\boldsymbol{x}}_d = -\bar{\alpha}_d \boldsymbol{g}, \end{cases} \tag{4-45}$$

with $\bar{\alpha}_d > m - 1$. Then the center point moves along the gradient descent direction.

Proof. From (4-45), the dynamical system for $\boldsymbol{x}_c$ is

$$\dot{\boldsymbol{x}}_c = \frac{m - 1 - \bar{\alpha}_d}{m} \boldsymbol{g},$$

from which the conclusion follows directly. □

The above analysis shows that the trial step can pull the center of the search points from the gradient ascent direction back toward the gradient descent direction.

In the current implementation of the algorithm, the well-poisedness of the interpolation set is checked and improved before using the model. This is achieved by calling the model improvement step, which takes into account the positions and distribution of the iteration/interpolation/search points in order to obtain a well-conditioned interpolation model. For more general discussions, see the work of Conn, Scheinberg, and Vicente [95].

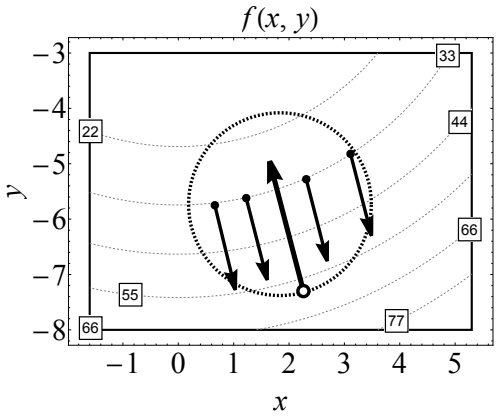


Figure 4-8 The "antidromic" point leading to a gradient descent direction

The "antidromic" point leading a gradient descent direction

In addition to the trial step discussed above, we also design a structure improvement step. Its motivation is to enlarge the region of attraction for local input-to-state stability [210], thereby making the search direction more likely to converge stably toward the gradient descent direction. Indeed, in Theorem 4.25, one of the assumptions requires that the norm of the gradient at the center point should be greater than ξ. It should be noted that, in Theorem 4.25, there exists a lower bound for ξ.

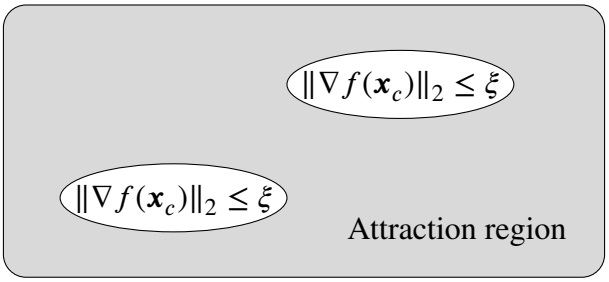


Figure 4-9 Attraction region ($\xi > \frac{|\delta|(\mu_n-\mu_1)}{\beta\exp(\bar{f}-f_c)M\mu_1}$)

Remark 4.5. Figure 4-9 illustrates, via the shaded region, the attraction region of local input-to-state stable equilibrium in Theorem 4.25 ($\mu_1 \neq 0$).

It can be observed that when the condition number of matrix $\boldsymbol{C}$ is small, the lower bound above decreases. In the algorithm implementation, we heuristically try to reduce the largest eigenvalue so that the eigenvalues become closer to each other. This heuristic step yields good numerical performance, although it does not strictly guarantee that the covariance matrix is well-conditioned.

For the covariance matrix $\boldsymbol{C}$, a radially distributed set of search points can make μ_1 and μ_n closer to each other. This typically occurs when the interpolation set is well-poised, which is considered in the model improvement step. Moreover, suppose that $\boldsymbol{v} \in \Re^n$ is a nonzero eigenvector of $\boldsymbol{C}$ corresponding to the largest eigenvalue μ_n. Then

we have

$$\begin{aligned}\mu_n &= \max_{\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^\top \boldsymbol{C}\boldsymbol{v} \\ &= \max_{\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^\top \left[\sum_{i=1}^{m} \left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top\right] \boldsymbol{v} \\ &= \max_{\|\boldsymbol{v}\|_2=1} \sum_{i=1}^{m} \left[\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top \boldsymbol{v}\right]^2.\end{aligned}$$

Hence, we obtain the following upper and lower bounds:

$$\sum_{i=1}^{m} \left\|\boldsymbol{x}_i - \boldsymbol{x}_c\right\|_2^2 \geq \max_{\|\boldsymbol{v}\|_2=1} \sum_{i=1}^{m} \left[\left(\boldsymbol{x}_i - \boldsymbol{x}_c\right)^\top \boldsymbol{v}\right]^2 \geq \max_i \left\|\boldsymbol{x}_i - \boldsymbol{x}_c\right\|_2^2.$$

Meanwhile, Algorithm 13 can reduce both the upper and lower bounds of the largest eigenvalue of the covariance matrix $\boldsymbol{C}$. For simplicity, the iteration index k has been omitted here.

Algorithm 13 Structure step

1: Discard the farthest point in the search set $\boldsymbol{x}_{\text{far}} := \arg\max_{\boldsymbol{x}_i} \left\|\boldsymbol{x}_i - \boldsymbol{x}_c\right\|_2^2$.

2: Add a new point to the search set $\boldsymbol{x}_{\text{new}} = \frac{1}{m-1} \sum_{i \neq \text{far}} \boldsymbol{x}_i$ (replace $\boldsymbol{x}_{\text{far}}$).

The above discussion presented the trial step and structure step in the SUSD-TR algorithm.

4.2.5 Numerical Results

This section reports numerical results, including solving two test problems and comparing the performance of SUSD-TR with other derivative-free optimization algorithms on a test problem set.

Example 4.1. We implemented MATLAB codes of the corresponding methods to minimize the 2-dimensional Rosenbrock function

$$f(x_1, x_2) = \left(1 - x_1\right)^2 + 100\left(x_2 - x_1^2\right)^2, \tag{4-46}$$

and the function

$$f(x_1, x_2) = \left(1 - x_1\right)^2 + \left(x_2 - x_1^2\right)^2, \tag{4-47}$$

to test and compare the SUSD algorithm [117] (executing only the line search step of SUSD-TR) and our SUSD-TR algorithm, where $(x_1, x_2)^\top$ denotes $\boldsymbol{x} \in \Re^2$. Figure 4-10 shows the iteration trajectories of SUSD and SUSD-TR for solving the two problems. Hollow circles indicate the center points in the iteration process. We observe that the SUSD algorithm exhibits instability in the search direction, with $\boldsymbol{v}_1$ failing to converge

and eventually missing the minimizer $(1, 1)^{\mathsf{T}}$. In contrast, applying SUSD-TR successfully converges to the minimizer.

For these experiments, the algorithm parameters were set as $\kappa = 1.2$, $\Delta_k = 5$, $\beta = 1$, and the five initial search points were

$$\boldsymbol{x}_1^{(0)} = \begin{pmatrix} 20 \\ 0 \end{pmatrix}, \quad \boldsymbol{x}_2^{(0)} = \begin{pmatrix} 23 \\ 4 \end{pmatrix}, \quad \boldsymbol{x}_3^{(0)} = \begin{pmatrix} 23 \\ -4 \end{pmatrix}, \quad \boldsymbol{x}_4^{(0)} = \begin{pmatrix} 17 \\ 4 \end{pmatrix}, \quad \boldsymbol{x}_5^{(0)} = \begin{pmatrix} 17 \\ -4 \end{pmatrix}.$$

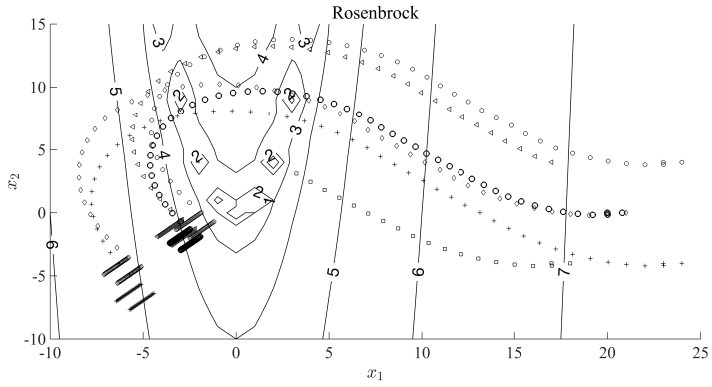
The model function in SUSD-TR was an underdetermined quadratic interpolation model.

In summary, when solving these examples, the SUSD algorithm sometimes fails to converge, with search points moving along gradient ascent directions. However, SUSD-TR effectively converges to the minimizer, with search points mostly moving along the descent direction of function values, requiring less than half the function evaluations compared with SUSD. One reason is that, in each iteration, there always exists a point moving along the descent direction of the quadratic interpolation model; when the model is accurate (at least fully linear), this direction is close to the gradient descent direction of the objective function.

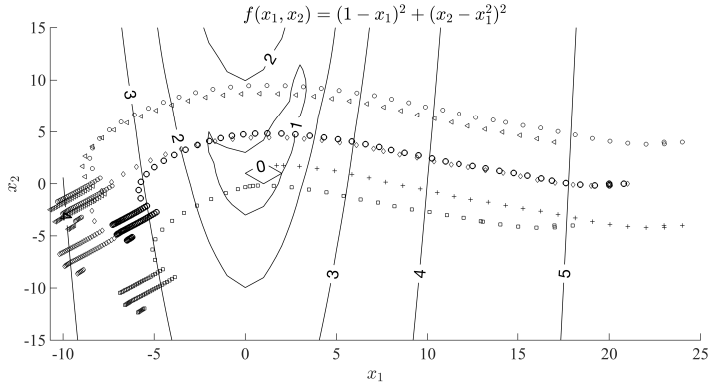
Table 4-4 Test problems for Figure 4-11 and Figure 4-12

ARGLINA	ARGLINA4	ARGLINB	ARGLINC	ARGTRIG
ARWHEAD	BDQRTIC	BDQRTICP	BDVALUE	BROWNAL
BROYDN3D	BROYDN7D	BRYBND	CHAINWOO	CHEBQUAD
CHNROSNBZ	CHPOWELLB	CHPOWELLS	CHROSEN	COSINE
CUBE	CURLY10	CURLY20	CURLY30	DIXMAANE
DIXMAANF	DIXMAANG	DIXMAANH	DIXMAANI	DIXMAANJ
DIXMAANK	DIXMAANL	DIXMAANM	DIXMAANN	DIXMAANO
DIXMAANP	DQRTIC	EDENSCH	ENGVAL1	ERRINROS
EXPSUM	EXTROSNB	EXTTET	FIROSE	FLETCBV2
FLETCBV3	FLETCHCR	FMINSRF2	FREUROTH	GENBROWN
GENHUMPS	GENROSE	INDEF	INTEGREQ	LIARWHD
LILIFUN3	LILIFUN4	MOREBV	MOREBVL	NCB20
NCB20B	NONCVXU2	NONCVXUN	NONDIA	NONDQUAR
PENALTY1	PENALTY2	PENALTY3	PENALTY3P	POWELLSG
POWER	ROSENBROCK	SBRYBND	SBRYBNDL	SCHMVETT
SCOSINE	SCOSINEL	SEROSE	SINQUAD	SPARSINE
SPARSQUR	SPHRPTS	SPMSRTLS	SROSENBR	STMOD
TOINTGSS	TOINTTRIG	TQUARTIC	TRIGSABS	-

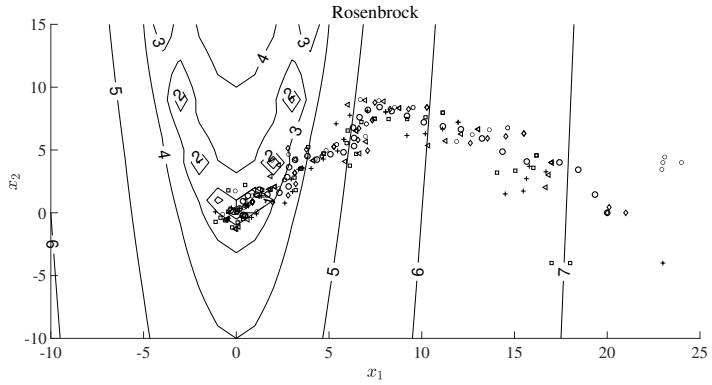
Solving the above classical examples demonstrates the advantages of our algorithm.



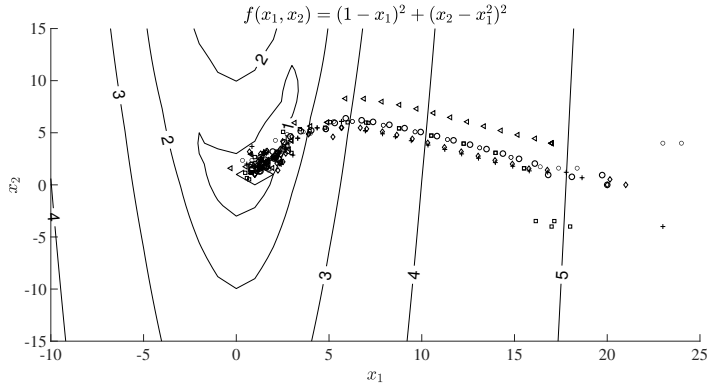
(a) Minimizing (4-46) using SUSD



(b) Minimizing (4-47) using SUSD



(c) Minimizing (4-46) using SUSD-TR



(d) Minimizing (4-47) using SUSD-TR

Figure 4-10 Solving the 2-dimensional test problems by SUSD and SUSD-TR

To further compare performance, we tested our algorithm against the derivative-free optimization algorithm based on the SUSD direction [117], as well as the representative Nelder-Mead method [51] and NEWUOA [94]. The test problems listed in Table 4-4 range in dimension from 2 to 120 and are taken from classical unconstrained optimization test function sets [177, 178, 180, 181, 185, 186]. The corresponding numerical results are shown in Figures 4-11 and 4-12.

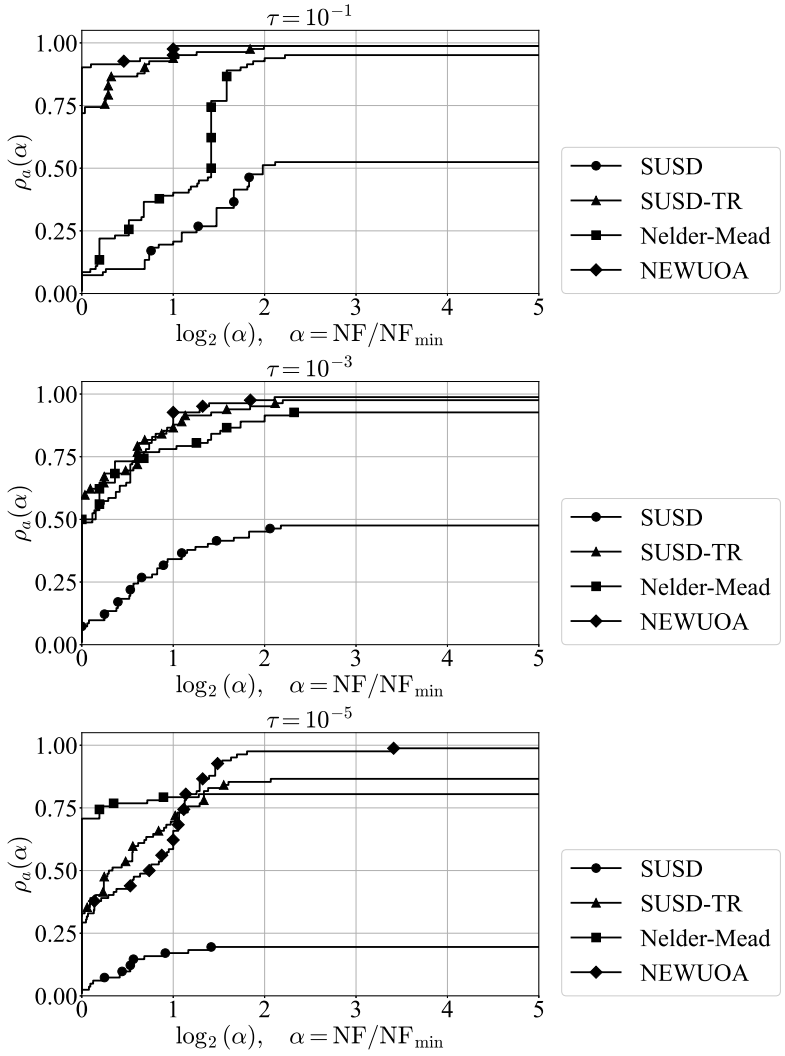


Figure 4-11 Performance Profile for solving the test problems

Our algorithm starts from $m = 2n+1$ randomly selected initial points, with parameters set as $\beta = 1$, $P = 5$, $\varepsilon = 10^{-6}$, $\kappa = 1.2$, and accuracy levels $\tau = 10^{-1}, 10^{-3}, 10^{-5}$. From Figures 4-11 and 4-12, we can observe that, for the tested problems, the SUSD-TR algorithm is more effective than the Nelder-Mead method at certain accuracy levels and can achieve performance comparable to the NEWUOA algorithm, efficiently solving these test problems. Its numerical performance is clearly superior to the method based solely on the SUSD direction (i.e., the pure line-search type). Other algorithms were run with their respective default parameters of the same scale.

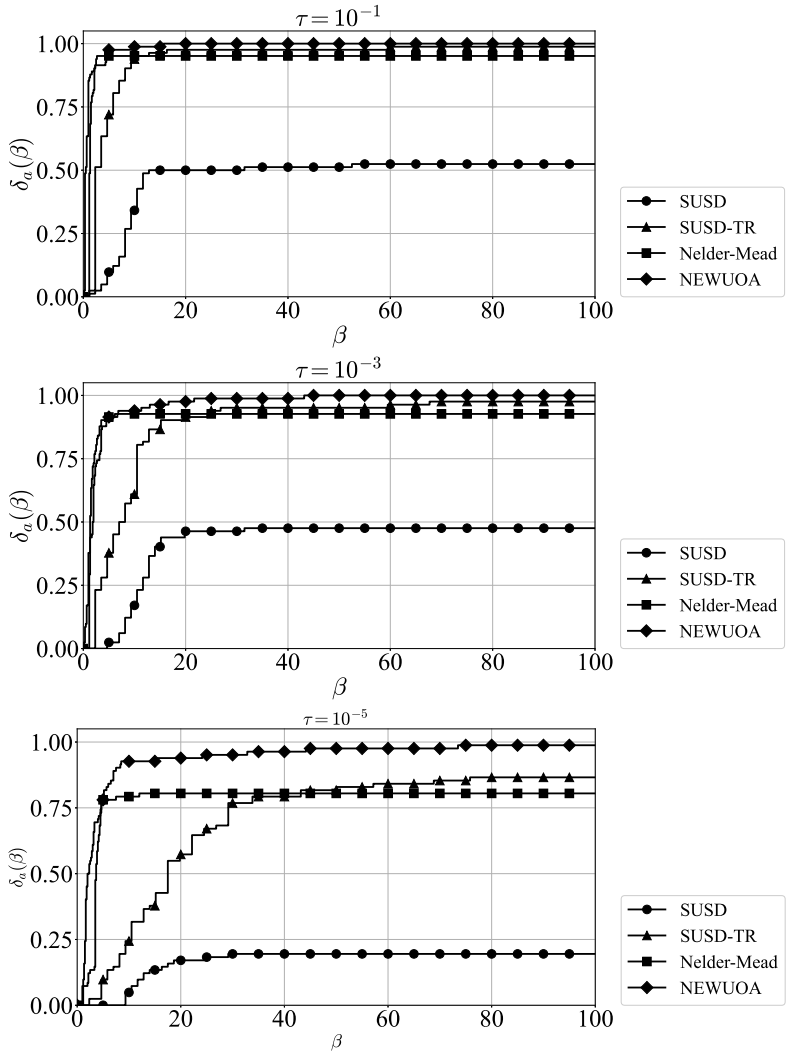


Figure 4-12 Data Profile for solving the test problems

In addition, both the SUSD-TR and SUSD algorithms can handle problems with high function evaluation costs in a parallelizable way, which is a key advantage over other methods (since in the line-search step, *m* points can be evaluated simultaneously).

Based on the above numerical tests, we conclude that compared with using the SUSD direction alone, our algorithm successfully improves problem-solving performance by combining the SUSD direction with trust-region techniques.

4.2.6 Conclusion

In this section, we proposed the SUSD-TR algorithm, which combines the process of solving a trust-region subproblem of the interpolation model function with the process of propagating points along the SUSD direction. We presented the dynamical system corresponding to the SUSD-TR algorithm and analyzed the stability of its parallel search direction. Numerical results demonstrated the advantages of our SUSD-TR method. In future work, we will consider more effective step-size strategies, better ways of combin-

ing steps, and different SUSD directions. Moreover, extending the approach to small-scale modifications of multiple points and to solving constrained problems will be important future research directions.

Chapter 5 Conclusion and Future Work

Many optimization problems arising from science, engineering, artificial intelligence, and machine learning involve situations where derivatives are unavailable or unreliable. In such problems, the objective function can only be treated as a black-box output, without providing derivative information. To address such cases, derivative-free optimization (DFO) methods are required. DFO is one of the most important, open, and challenging areas in computational science and engineering, with enormous practical potential. The goal of designing DFO methods is to achieve optimization using as few function evaluations as possible. We know that trust-region methods are a celebrated class of algorithms in nonlinear optimization. Trust-region algorithms generate new iterates by minimizing a quadratic model within a region close to the current point. In the derivative-free case, the corresponding models are usually constructed by polynomial interpolation, regression, or other approximation techniques. Such optimization methods are referred to as model-based DFO methods. Chapters 2 through 4 of this dissertation study unconstrained DFO problems.

Chapter 2 investigates how to design better approximation models and discusses in depth the relationship between approximation and optimization. Among the most effective model-based DFO methods are trust-region algorithms based on underdetermined quadratic interpolation models. During iterations, using different techniques to update the quadratic model produces different interpolation models. The least norm quadratic models proposed by Powell, Conn, and Toint [171, 211] have been internationally leading models over the last two decades. This dissertation improves upon these models in the following two ways.

First, we propose to construct a quadratic model function by minimizing the H^2 norm of the change between the new quadratic model and the old one, thereby reducing the lower bound on the number of interpolation points or equations. This method has corresponding projection properties and error bounds, and we provide an easily implementable model updating formula. Furthermore, by giving the barycenter of the weights in the least weighted H^2 norm update of quadratic models, we identify the optimal weight coefficients, accompanied by theoretical analysis and numerical results.

Second, recognizing that trust-region steps can provide information on model optimality in model-based DFO methods, we propose a new perspective to understand and analyze the Conn–Toint model, and introduce a new, easily implementable model. The article also discusses theoretical motivations for using such a model. To the best of our knowledge, this is the first work to construct underdetermined quadratic interpolation models for DFO by explicitly considering the nature of trust-region iterations.

Future work includes further exploration of the relationship between approximation and optimization. The work in this dissertation has already revealed the importance of approximation for optimization. Building upon the discussion of least H^2 norm updated quadratic models and function optimality, more DFO algorithms can be developed, particularly algorithms with stronger convergence guarantees. For example, adaptive weight coefficients in the least H^2 norm update for problems with different structures could be designed. In addition, comparisons of weight coefficients in the least weighted H^2 norm models from other perspectives may be explored. Further details and work are also needed: for instance, can we construct better models incorporating model optimality using only "function value comparison" mechanisms (instead of exact function value evaluation)? In fact, this dissertation's exploration of "optimization and approximation" aims to foster integration of these important disciplines. Notably, our latest findings indicate that DFO lies at the intersection of approximation and optimization. "Optimization for approximation" and "approximation for optimization" represent two promising avenues of research with great potential in both mathematical theory and practical applications. Another future direction is determining better choices for the number of interpolation points at each iteration.

Chapter 3 discusses DFO with transformed objective functions. We propose a DFO method for solving optimization problems with transformed objectives and provide a corresponding probing scheme. For strictly convex models with unique minimizers in the trust region, we prove the existence of model-optimality-preserving transformations beyond translations, and we give a necessary and sufficient condition for transformed function values corresponding to model-optimality-preserving transformations. We derive the corresponding quadratic model for affine transformations of the objective function and prove that some monotone positive transformations (even affine ones with positive multiplicative coefficients) are not model-optimality-preserving. We also conduct interpolation error analysis, provide the case for affine transformations, and present convergence analysis for first-order critical points. Numerical results for test and practical problems are given. As discussed in Chapter 3, there is still much to study regarding transformed-objective DFO. For example, one could attempt to apply minimal Frobenius norm quadratic model updates to practical transformed DFO problems (e.g., black-box optimization with noise-injection or privacy-preserving mechanisms). Furthermore, convergence analysis under weaker assumptions for transformed-objective problems remains an open and challenging problem. The open question of minimizing "moving-target" objectives without derivatives, as raised in this dissertation, is also interesting and challenging. We believe that transformed DFO has begun to show potential impacts in both theory and applications. In fact, it characterizes noisy optimization problems from a new perspective and derives new theoretical results. Thus, further theoretical analysis and algorithm design for such problems are valuable. Moreover,

transformed DFO is closely related to noisy black-box problems and machine learning tasks, making relevant AI applications worthy of attention.

Chapter 4 consists of two parts. The first part discusses subspace methods for solving large-scale DFO problems, and the second explores parallel DFO algorithms combining line-search and trust-region frameworks. In current DFO problems, large-scale problems remain a bottleneck, since when the problem dimension is high, the cost of constructing local polynomial models and interpolation errors can be prohibitive. We consider this the curse of dimensionality in DFO. To address this challenge, we propose a new subspace optimization method for solving large-scale black-box problems, which uses subspace techniques and quadratic models to efficiently search for minimizers. Our new method, 2D-MoSub, iteratively employs two-dimensional quadratic models in two-dimensional subspaces to find new points and perform updates, and it enjoys favorable approximation error and convergence properties. Future work includes solving even larger-scale problems (with or without derivatives). In addition, we plan to explore more techniques, including subspace methods, parallelization, and randomization, to handle large-scale problems. We also intend to design new subspace selection strategies and study large-scale constrained problems. In practice, solving large-scale problems is critical in many applications, including machine learning and other practical needs. We find that 2D-MoSub holds promise for solving large-scale problems of significant importance in optimization and numerical computation. Moreover, we will consider using high-performance computing for large-scale analysis and optimization and further explore randomized subspace methods.

In the second part of Chapter 4, we introduce a parallel method, SUSD-TR, that enhances line search using quadratic models. This method combines the SUSD direction, derived from the covariance matrix of interpolation points, with the solution of the trust-region subproblem for quadratic models at each iteration. We analyze the dynamical system of the SUSD-TR algorithm and the properties of its iterative search direction. Numerical results demonstrate the efficiency of this algorithm. Future work includes further study of parallelization methods, improved step-size strategies, better combinations of the two frameworks, and exploration of different SUSD directions. Moreover, we plan to investigate simultaneous small-scale corrections on multiple points and extend the method to constrained optimization problems. In fact, SUSD-TR can be regarded as a parallel and derivative-free version of the combination of line-search and trust-region frameworks. It seeks to leverage and combine the strengths of these two classical iterative optimization frameworks. Going forward, we will continue to study the comparison and combination of model-based DFO methods and direct search methods, as well as explore more specific applications, such as distributed source localization scenarios.

References

[1] 袁亚湘, 孙文瑜. 最优化理论与方法 [M]. 科学出版社, 1997.

[2] 袁亚湘. 非线性规划数值方法 [M]. 上海科学技术出版社, 1993.

[3] Nocedal J, Wright S J. Numerical Optimization [M]. Berlin: Springer, 2006.

[4] Bertsekas D P. Nonlinear Programming [M]. Belmont, MA, USA: Athena Scientific, 1999.

[5] 戴彧虹, 刘新为. 线性与非线性规划算法与理论 [J]. 运筹学学报, 2014, 18(1): 69-92.

[6] 郭田德, 韩丛英, 唐思琦. 组合优化机器学习方法 [M]. 科学出版社, 2019.

[7] 刘浩洋, 户将, 李勇锋, 等. 最优化：建模、算法与理论 [M]. 高等教育出版社, 2020.

[8] 刘歆, 刘亚锋. 凸优化 [M]. 科学出版社, 2024.

[9] Ferris M C, Pang J S. Engineering and economic applications of complementarity problems [J]. SIAM Review, 1997, 39(4): 669-713.

[10] Biswas P, Lian T C, Wang T C, et al. Semidefinite programming based algorithms for sensor network localization [J]. ACM Transactions on Sensor Networks, 2006, 2(2): 188-220.

[11] Liu X, Wang X, Wen Z, et al. On the convergence of the self-consistent field iteration in kohn–sham density functional theory [J]. SIAM Journal on Matrix Analysis and Applications, 2014, 35(2): 546-558.

[12] Bottou L, Curtis F E, Nocedal J. Optimization methods for large-scale machine learning [J]. SIAM review, 2018, 60(2): 223-311.

[13] Li Z, Zhang S, Wang Y, et al. Alignment of molecular networks by integer quadratic programming [J]. Bioinformatics, 2007, 23(13): 1631-1639.

[14] Wen Z, Yin W. A feasible method for optimization with orthogonality constraints [J]. Mathematical Programming, 2013, 142(1): 397-434.

[15] 刘歆. 强关联多电子体系的优化模型与算法 [J]. 计算数学, 2023, 45(2): 141-159.

[16] Li T, Xie J, Lu S, et al. Duopoly game of callable products in airline revenue management [J]. European Journal of Operational Research, 2016, 254(3): 925-934.

[17] Yin J, Li Q. A semismooth Newton method for support vector classification and regression [J]. Computational Optimization and Applications, 2019, 73(2): 477-508.

[18] Xia Y, Yuan Y X. A new linearization method for quadratic assignment problems [J]. Optimisation Methods and Software, 2006, 21(5): 805-818.

[19] Xu D, Du D. The k-level facility location game [J]. Operations Research Letters, 2006, 34 (4): 421-426.

[20] Conn A R, Scheinberg K, Vicente L N. Introduction to Derivative-free Optimization [M]. Philadelphia: SIAM, 2009.

[21] Audet C, Hare W. Derivative-free and Blackbox Optimization [M]. Heidelberg: Springer, 2017.

[22] Audet C, Orban D. Finding optimal algorithmic parameters using derivative-free optimization [J]. SIAM Journal on Optimization, 2006, 17(3): 642-664.

[23] Aly A, Guadagni G, Dugan J B. Derivative-free optimization of neural networks using local search [C]//2019 IEEE 10th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON). Piscataway: IEEE, 2019: 293-299.

[24] Higham N J. Accuracy and Stability of Numerical Algorithms [M]. Philadelphia: SIAM, 2002.

[25] Levina T, Levin Y, McGill J, et al. Dynamic pricing with online learning and strategic consumers: An application of the aggregating algorithm [J]. Operations Research, 2009, 57(2): 327-341.

[26] Li S, Xie P, Zhou Z, et al. Simulation of interaction of folded waveguide space traveling wave tubes with derivative-free mixedinteger based NEWUOA algorithm [C]//2021 7th International Conference on Computer and Communications. Piscataway: IEEE, 2021: 1215-1219.

[27] Booker A, Frank P, Dennis J E, et al. Managing surrogate objectives to optimize a helicopter rotor design-further experiments [C]//7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. 1998: 4717.

[28] Booker A J, Dennis J E, Frank P D, et al. Optimization using surrogate objectives on a helicopter test example [M]//Computational Methods for Optimal Design and Control. Boston: Springer, 1998: 49-58.

[29] Serafini D B. A Framework for Managing Models in Nonlinear Optimization of Computationally Expensive Functions [M]. Houston: Rice University, 1999.

[30] Audet C, Dennis J E. A pattern search filter method for nonlinear programming without derivatives [J]. SIAM Journal on Optimization, 2004, 14(4): 980-1010.

[31] Marsden A L. Aerodynamic Noise Control by Optimal Shape Design [M]. Stanford: Stanford University, 2005.

[32] Marsden A L, Wang M, Dennis J E, et al. Optimal aeroacoustic shape design using the surrogate management framework [J]. Optimization and Engineering, 2004, 5(2): 235-262.

[33] Duvigneau R, Visonneau M. Hydrodynamic design using a derivative-free method [J]. Structural and Multidisciplinary Optimization, 2004, 28: 195-205.

[34] Green J E J. Faster and more accurate testing [J]. Advanced Materials and Processes, 1987, 131: 72, 75-76, 79.

[35] Gu T, Li W, Zhao A, et al. BBGP-sDFO: Batch Bayesian and Gaussian process enhanced subspace derivative free optimization for high-dimensional analog circuit synthesis [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2024, 43(2): 417-430.

[36] Martins J R, Lambe A B. Multidisciplinary design optimization: A survey of architectures [J]. AIAA Journal, 2013, 51(9): 2049-2075.

[37] Ragonneau T M. Model-based derivative-free optimization methods and software [D]. Hong Kong Polytechnic University, 2023.

[38] Ghanbari H, Scheinberg K. Black-box optimization in machine learning with trust region based derivative free algorithm [R]. 2017.

[39] Wilson Z T, Sahinidis N V. The ALAMO approach to machine learning [J]. Computers & Chemical Engineering, 2017, 106: 785-795.

[40] Ughi G, Abrol V, Tanner J. An empirical study of derivative-free-optimization algorithms for targeted black-box attacks in deep neural networks [J]. Optimization and Engineering, 2022, 23(3): 1319-1346.

[41] Tett S F B, Gregory J M, Freychet N, et al. Does model calibration reduce uncertainty in climate projections? [J]. Journal of Climate, 2022, 35(8): 2585 - 2602.

[42] 丁晓东. 基于插值模型的无导数优化方法及其应用 [D]. 北京: 中国科学院数学与系统科学研究院, 2010.

[43] Meza J C, Martinez M L. Direct search methods for the molecular conformation problem [J]. Journal of Computational Chemistry, 1994, 15(6): 627-632.

[44] Alberto P, Nogueira F, Rocha H, et al. Pattern search methods for user-provided points: Application to molecular geometry problems [J]. SIAM Journal on Optimization, 2004, 14 (4): 1216-1236.

[45] Davis P. Looking beyond the black box in optimization [J]. SIAM News, 2016, 49(8).

[46] 张在坤. 无导数优化方法的研究 [D]. 北京: 中国科学院数学与系统科学研究院, 2012.

[47] Wright M H. Direct search methods: Once scorned, now respectable [J]. Pitman Research Notes in Mathematics Series, 1996: 191-208.

[48] Lewis R M, Torczon V J, Trosset M W. Direct search methods: Then and now [J]. Journal of Computational and Applied Mathematics, 2000, 124(1-2): 191-207.

[49] Kolda T, Lewis R, Torczon V J. Optimization by direct search: New perspectives on some classical and modern methods [J]. SIAM Review, 2003, 45: 385-482.

[50] Hooke R, Jeeves T A. "Direct search" solution of numerical and statistical problems [J]. Journal of the ACM, 1961, 8(2): 212-229.

[51] Nelder J A, Mead R. A simplex method for function minimization [J]. The Computer Journal, 1965, 7(4): 308-313.

[52] Tseng P. Fortified-descent simplicial search method: A general approach [J]. SIAM Journal on Optimization, 1999, 10: 269-288.

[53] Fermi E, Metropolis N. Numerical solution of a minimization problem, Los Alamos unclassified report ls-492 [R]. Los Alamos, USA: Alamos National Laboratory, 1952.

[54] Box G E P. Evolutionary operation: A method for increasing industrial productivity [J]. Applied Statistics, 1957, 6(2): 81-101.

[55] Torczon V J. Multi-directional search: A direct search algorithm for parallel machines [D]. Houston, TX, USA: Rice University, 1989.

[56] Dennis J E, Torczon V J. Direct search methods on parallel machines [J]. SIAM Journal on Optimization, 1991, 1(4): 448-474.

[57] Torczon V J. On the convergence of the multidirectional search algorithm [J]. SIAM Journal on Optimization, 1991, 1(1): 123-145.

[58] Lewis R M, Torczon V J. Rank ordering and positive bases in pattern search algorithms: ICASE Technical report TR 96-71 [R]. Hampton, USA: NASA Langley Research Center, 1996.

[59] Torczon V J. On the convergence of pattern search algorithms [J]. SIAM Journal on optimization, 1997, 7(1): 1-25.

[60] Lewis R M, Torczon V J. Pattern search algorithms for bound constrained minimization [J]. SIAM Journal on Optimization, 1999, 9(4): 1082-1099.

[61] Lewis R M, Torczon V J. Pattern search methods for linearly constrained minimization [J]. SIAM Journal on Optimization, 2000, 10(3): 917-941.

[62] Audet C, Dennis J E. Analysis of generalized pattern searches [J]. SIAM Journal on Optimization, 2000, 13.

[63] Hough P D, Kolda T G, Torczon V J. Asynchronous parallel pattern search for nonlinear optimization [J]. SIAM Journal on Scientific Computing, 2002, 23(1): 134-156.

[64] Kolda T G. Revisiting asynchronous parallel pattern search for nonlinear optimization [J]. SIAM Journal on Optimization, 2006, 16(2): 563-586.

[65] Abramson M A, Audet C, Dennis J E. Nonlinear programming by mesh adaptive direct searches [J]. SIAG/OPT Views-and-News, 2006, 17: 2-11.

[66] Audet C, Le Digabel S, Tribes C. NOMAD user guide [R]. Les Cahiers du GERAD, 2009.

[67] 邓乃扬. 计算方法丛书: 无约束最优化计算方法 [M]. 科学出版社, 1982.

[68] McKinnon K I M. Convergence of the Nelder-Mead simplex method to a nonstationary point [J]. SIAM Journal on Optimization, 1998, 9(1): 148-158.

[69] Kelley C T. Detection and remediation of stagnation in the Nelder–Mead algorithm using a sufficient decrease condition [J]. SIAM Journal on Optimization, 1999, 10(1): 43-55.

[70] Nazareth L, Tseng P. Gilding the lily: A variant of the Nelder-Mead algorithm based on golden-section search [J]. Computational Optimization and Applications, 2002, 22(1): 133-144.

[71] Price C J, Coope I D, Byatt D. A convergent variant of the Nelder-Mead algorithm [J]. Journal of Optimization Theory and Applications, 2002, 113(1): 5-19.

[72] Rosenbrock H H. An automatic method for finding the greatest or least value of a function [J]. The Computer Journal, 1960, 3(3): 175-184.

[73] Swann W H. Direct search methods [M]//Murray W. Numerical Methods for Unconstrained Optimization. London: Academic Press, 1972: 13-28.

[74] Smith C S. The automatic computation of maximum likelihood estimates [R]. Scientific Department, National Coal Board, 1962.

[75] Powell M J D. An efficient method for finding the minimum of a function of several variables without calculating derivatives [J]. The Computer Journal, 1964, 7(2): 155-162.

[76] Stewart III G W. A modification of Davidon's minimization method to accept difference approximations of derivatives [J]. Journal of the ACM (JACM), 1967, 14(1): 72-83.

[77] Gill P E, Murray W. Quasi-Newton methods for unconstrained optimization [J]. IMA Journal of Applied Mathematics, 1972, 9(1): 91-108.

[78] Gill P E, Murray W, Saunders M A, et al. Computing forward-difference intervals for numerical optimization [J]. SIAM Journal on Scientific and Statistical Computing, 1983, 4(2): 310-321.

[79] Nesterov Y, Spokoiny V. Random gradient-free minimization of convex functions [J]. Foundations of Computational Mathematics, 2017, 17(2): 527-566.

[80] Duchi J C, Jordan M I, Wainwright M J, et al. Optimal rates for zero-order convex optimization: The power of two function evaluations [J]. IEEE Transactions on Information Theory, 2015, 61(5): 2788-2806.

[81] Scheinberg K. Finite difference gradient approximation: To randomize or not? [J]. INFORMS Journal on Computing, 2022, 34(5): 2384-2388.

[82] Zhigljavsky A A. Theory of Global Random Search [M]. Heidelberg: Springer Science & Business Media, 2012.

[83] Berahas A S, Cao L, Choromanski K, et al. A theoretical and empirical comparison of gradient approximations in derivative-free optimization [J]. Foundations of Computational Mathematics, 2022, 22(2): 507-560.

[84] Diniz-Ehrhardt M A, Martínez J M, Raydán M. A derivative-free nonmonotone line-search technique for unconstrained optimization [J]. Journal of Computational and Applied Mathematics, 2008, 219(2): 383-397.

[85] Zangwill W I. Minimizing a function without calculating derivatives [J]. The Computer Journal, 1967, 10(3): 293-296.

[86] Gilmore P, Kelley C T. An implicit filtering algorithm for optimization of functions with many local minima [J]. SIAM Journal on Optimization, 1995, 5(2): 269-285.

[87] Kelley C T. A brief introduction to implicit filtering [R]. North Carolina State University. Center for Research in Scientific Computation, 2002.

[88] Kelley C T. Implicit filtering [M]. Philadelphia: SIAM, 2011.

[89] Greenstadt J. A quasi-Newton method with no derivatives [J]. Mathematics of Computation, 1972, 26(117): 145-166.

[90] Greenstadt J. Revision of a derivative-free quasi-Newton method [J]. Mathematics of Computation, 1978, 32(141): 201-221.

[91] Winfield D. Function and functional optimization by interpolation in data tables [D]. Harvard University, 1969.

[92] Powell M J D. On trust region methods for unconstrained minimization without derivatives [J]. Mathematical Programming, 2003, 97: 605-623.

[93] Powell M J D. Least Frobenius norm updating of quadratic models that satisfy interpolation conditions [J]. Mathematical Programming, 2004, 100: 183-215.

[94] Powell M J D. The NEWUOA software for unconstrained optimization without derivatives [M]//Large-scale Nonlinear Optimization. Boston: Springer, 2006: 255-297.

[95] Conn A, Scheinberg K, Vicente L N. Geometry of sample sets in derivative free optimization. part ii: polynomial regression and underdetermined interpolation [J]. IMA Journal of Numerical Analysis, 2008, 28: 721-748.

[96] Conn A, Scheinberg K, Vicente L N. Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points [J]. SIAM Journal on Optimization, 2009, 20: 387-415.

[97] Xie P, Yuan Y. Least H^2 norm updating of quadratic interpolation models for derivative-free trust-region algorithms [R]. IMA Journal of Numerical Analysis, 2025: drae106.

[98] Xie P, Yuan Y. A new two-dimensional model-based subspace method for large-scale unconstrained derivative-free optimization: 2D-MoSub [R]. 2023.

[99] Xie P. Sufficient conditions for error distance reduction in the ℓ^2-norm trust region between minimizers of local nonconvex multivariate quadratic approximates [R]. Journal of Computational and Applied Mathematics, 2025, 453: 116146.

[100] Björkman M, Holmström K. Global optimization of costly nonconvex functions using radial basis functions [J]. Optimization and Engineering, 2000, 1: 373-397.

[101] Wild S M, Regis R G, Shoemaker C A. ORBIT: Optimization by radial basis function interpolation in trust-regions [J]. SIAM Journal on Scientific Computing, 2008, 30(6): 3197-3219.

[102] Xie P, Yuan Y. A derivative-free optimization algorithm combining line-search and trust-region techniques [J]. Chinese Annals of Mathematics, Series B, 2023, 44(5): 719-734.

[103] Bandeira A S, Scheinberg K, Vicente L N. Convergence of trust-region methods based on probabilistic models [J]. SIAM Journal on Optimization, 2014, 24(3): 1238-1264.

[104] Gratton S, Royer C W, Vicente L N, et al. Complexity and global rates of trust-region methods based on probabilistic models [J]. IMA Journal of Numerical Analysis, 2017, 38(3): 1579-1597.

[105] Van Laarhoven P J M, Aarts E H L. Simulated Annealing: Theory and Applications: volume 37 [M]. Dordrecht: Springer Netherlands, 1987.

[106] Goldberg D E. Genetic algorithms in search, optimization and machine learning [M]. Boston: Addison−Wesley Longman, 1989.

[107] Goldberg D E, Holland J H. Genetic algorithms and machine learning [J]. Machine Learning, 1988, 3(2-3): 95-99.

[108] Holland J H. Adaptation in natural and artificial systems: Introductory analysis with applications to biology, control, and artificial intelligence [M]. Cambridge, MA: MIT Press, 1992.

[109] Kivkpatrick S. Optimization by simulated annealing: Quantitative studies [J]. Journal of Statistical Physics, 1984, 34(5/6): 975-986.

[110] Kirkpatrick S, Gelatt C, Vecchi M. Optimization by simulated annealing [J]. Science (New York, N.Y.), 1983, 220: 671-80.

[111] Cartis C, Gould N I M, Toint Ph L. On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization [J]. SIAM Journal on Optimization, 2012, 22(1): 66-86.

[112] Sarker R, Mohammadian M, Yao X. Evolutionary Optimization: volume 48 [M]. Kluwer Academic Pub, 2002.

[113] Hansen N, Ostermeier A. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation [C]//Proceedings of IEEE International Conference on Evolutionary Computation. IEEE, 1996: 312-317.

[114] Hansen N, Ostermeier A. Completely derandomized self-adaptation in evolution strategies [J]. Evolutionary Computation, 2001, 9(2): 159-195.

[115] Hansen N. The CMA evolution strategy: A tutorial [R]. 2016.

[116] Wu W, Zhang F. A speeding-up and slowing-down strategy for distributed source seeking with robustness analysis [J]. IEEE Transactions on Control of Network Systems, 2015, 3(3): 231-240.

[117] Al-Abri S, Lin T X, Tao M, et al. A derivative-free optimization method with application to functions with exploding and vanishing gradients [J]. IEEE Control Systems Letters, 2020, 5(2): 587-592.

[118] Zhang H, Conn A R, Scheinberg K. A derivative-free algorithm for least-squares minimization [J]. SIAM Journal on Optimization, 2010, 20(6): 3555-3576.

[119] Cartis C, Roberts L. A derivative-free Gauss–Newton method [J]. Mathematical Programming Computation, 2019, 11(4): 631-674.

[120] Grapiglia G N, Yuan J, Yuan Y. A derivative-free trust-region algorithm for composite nonsmooth optimization [J]. Computational and Applied Mathematics, 2016, 35(2): 475-499.

[121] Liu S, Wang L, Xiao N, et al. An inexact preconditioned zeroth-order proximal method for composite optimization [Z]. 2024.

[122] Xie P. A derivative-free trust-region method for optimization on the ellipsoid [J]. Journal of Physics: Conference Series, 2023, 2620(1): 012007.

[123] Tang Y, Li N. Distributed zero-order algorithms for nonconvex multi-agent optimization [C]//2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton). 2019: 781-786.

[124] Pelikan M, Goldberg D E, Cantú-Paz E. BOA: The Bayesian optimization algorithm [C]// Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99: volume 1. Burlington: Morgan Kaufmann Publishers, 1999: 525-532.

[125] Gratton S, Royer C W, Vicente L N, et al. Direct search based on probabilistic descent [J]. SIAM Journal on Optimization, 2015, 25(3): 1515-1541.

[126] Ghadimi S, Lan G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming [J]. SIAM Journal on Optimization, 2013, 23(4): 2341-2368.

[127] Larson J, Wild S M. A batch, derivative-free algorithm for finding multiple local minima [J]. Optimization and Engineering, 2016, 17(1): 205-228.

[128] Larson J, Menickelly M, Wild S M. Derivative-free optimization methods [J]. Acta Numerica, 2019, 28: 287-404.

[129] 张在坤. 无导数优化 [M]//中国学科发展战略：数学优化. 科学出版社, 2021: 84-92.

[130] Rios L M, Sahinidis N V. Derivative-free optimization: a review of algorithms and comparison of software implementations [J]. Journal of Global Optimization, 2013, 56(3): 1247-1293.

[131] Hansen N, Müller S D, Koumoutsakos P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES) [J]. Evolutionary Computation, 2003, 11(1): 1-18.

[132] Scheinberg K. Manual for Fortran software package DFO version 2.0 [R]. Tech. rep., 2003.

[133] Kelley C T. Users guide for IMFIL version 1.0 [R]. 2011.

[134] Ge D, Liu T, Liu J, et al. SOLNP+: A derivative-free solver for constrained nonlinear optimization [R]. 2022.

[135] Powell M J D. A tolerant algorithm for linearly constrained optimization calculations [J]. Mathematical Programming, 1989, 45: 547-566.

[136] Powell M J D. A direct search optimization method that models the objective and constraint functions by linear interpolation [M]//Advances in Optimization and Numerical Analysis. Dordrecht: Springer, 1994: 51-67.

[137] Powell M J D. UOBYQA: unconstrained optimization by quadratic approximation [J]. Mathematical Programming, 2002, 92(3): 555-582.

[138] Powell M J D. The BOBYQA algorithm for bound constrained optimization without derivatives [J]. Cambridge NA Report NA2009/06, University of Cambridge, Cambridge, 2009: 26-46.

[139] Powell M J D. On fast trust region methods for quadratic models with linear constraints [J]. Mathematical Programming Computation, 2015, 7(3): 237-267.

[140] Cartis C, Fiala J, Marteau B, et al. Improving the flexibility and robustness of model-based derivative-free optimization solvers [J]. ACM Transactions on Mathematical Software, 2019, 45(3): 1-41.

[141] Cartis C, Roberts L. Scalable subspace methods for derivative-free nonlinear least-squares optimization [J]. Mathematical Programming, 2023, 199(1-2): 461-524.

[142] Ragonneau T M, Zhang Z. PDFO: a cross-platform package for Powell's derivative-free optimization solvers [R]. 2023.

[143] Ragonneau T M. Model-based derivative-free optimization methods and software [D]. Hong Kong: Department of Applied Mathematics, The Hong Kong Polytechnic University, 2022.

[144] Xie P. NEWUOA-Matlab-Version-2.0 [R/OL]. 2022. https://github.com/PengchengXieLSEC/NEWUOA-Matlab-Version-2.0/releases/tag/2.0.

[145] Xie P. BOBYQA-Matlab-Version-1.0 [R/OL]. 2023. https://github.com/PengchengXieLSEC/BOBYQA-Matlab-Version-1.0/releases/tag/Version-1.0.

[146] Zhang Z. PRIMA: Reference implementation for Powell's methods with modernization and amelioration [R]. 2023.

[147] Spendley W, Hext G, Himsworth F. Sequential application of simplex designs in optimization and evolutionary operation [J]. Technometrics, 1962, 4: 441-461.

[148] Winfield D. Function minimization by interpolation in a data table [J]. IMA Journal of Applied Mathematics, 1973, 12.

[149] Tröltzsch A, Gratton S, Toint Ph L. A model-based trust-region algorithm for DFO and its adaptation to handle noisy functions and gradients [C]//The 21st International Symposium on Mathematical Programming. 2012.

[150] Xie P, Yuan Y. Derivative-free optimization with transformed objective functions (DFOTO) and the algorithm based on the least Frobenius norm updating quadratic model [J]. Journal of the Operations Research Society of China, 2025, 13: 327-363.

[151] Cartis C, Gould N I M, Toint Ph L. On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization [J]. SIAM Journal on Optimization, 2012, 22(1): 66-86.

[152] Conn A R, Gould N I, Toint Ph. L. Trust Region Methods [M]. Philadelphia: SIAM, 2000.

[153] Bandeira A S, Scheinberg K, Vicente L N. Computation of sparse low degree interpolating polynomials and their application to derivative-free optimization [J]. Mathematical Programming, 2012, 134(1): 223-257.

[154] Zhang Z. Sobolev seminorm of quadratic functions with applications to derivative-free optimization [J]. Mathematical Programming, 2014, 146(1-2): 77-96.

[155] Berghen F V, Bersini H. CONDOR, a new parallel, constrained extension of Powell's UOBYQA algorithm: Experimental results and comparison with the DFO algorithm [J]. Journal of Computational and Applied Mathematics, 2005, 181(1): 157-175.

[156] Conn A R, Scheinberg K, Toint Ph L. A derivative free optimization algorithm in practice [C]//7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. Reston: AIAA, 1998: 4718.

[157] Wild S M. MNH: A derivative-free optimization algorithm using minimal norm Hessians [C]//The Tenth Copper Mountain Conference on Iterative Methods. 2008.

[158] Oeuvray R, Bierlaire M. Boosters: A derivative-free algorithm based on radial basis functions [J]. International Journal of Modelling and Simulation, 2009, 29(1): 26-36.

[159] Marazzi M, Nocedal J. Wedge trust region methods for derivative free optimization [J]. Mathematical Programming, 2002, 91(2): 289-305.

[160] Conn A R, Scheinberg K, Toint Ph L. On the convergence of derivative-free methods for unconstrained optimization [M]//Approximation Theory and Optimization: Tributes to M.J.D. Powell. 1997: 83-108.

[161] Steihaug T. The conjugate gradient method and trust regions in large scale optimization [J]. SIAM Journal on Numerical Analysis, 1983, 20(3): 626-637.

[162] Toint Ph L. Towards an efficient sparsity exploiting newton method for minimization [M]// Sparse matrices and their uses. Academic Press, 1981: 57-88.

[163] Yuan Y. On the truncated conjugate gradient method [J]. Mathematical Programming, 2000, 87: 561-573.

[164] Dolan E D, Moré J J. Benchmarking optimization software with performance profiles [J]. Mathematical Programming, 2002, 91(2): 201-213.

[165] Moré J J, Wild S M. Benchmarking derivative-free optimization algorithms [J]. SIAM Journal on Optimization, 2009, 20(1): 172-191.

[166] Yuan Y. Subspace techniques for nonlinear optimization [M]//Jeltsch R, Li D Q, Sloan I H. Some Topics in Industrial and Applied Mathematics. Beijing: Higher Education Press, 2007: 206-218.

[167] Yuan Y. Subspace methods for large scale nonlinear equations and nonlinear least squares [J]. Optimization and Engineering, 2009, 10(2): 207-218.

[168] Yuan Y. A review on subspace methods for nonlinear optimization [C]//Proceedings of the International Congress of Mathematics. 2014: 807-827.

[169] Berglund E, Khirirat S, Wang X. Zeroth-order randomized subspace newton methods [C]// 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2022: 6002-6006.

[170] Powell M J D. Beyond symmetric Broyden for updating quadratic models in minimization without derivatives [J]. Mathematical Programming, 2013, 138(1): 475-500.

[171] Conn A R, Toint Ph L. An algorithm using quadratic interpolation for unconstrained derivative free optimization [M]//Di Pillo G, Giannessi F. Nonlinear Optimization and Applications. Boston: Springer, 1996: 27-47.

[172] Conn A R, Scheinberg K, Toint Ph L. A derivative free optimization algorithm in practice [C]//Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. AIAA, 1998: 129-139.

[173] Evans L C. Partial differential equations [M]. Providence, R.I.: American Mathematical Society, 2010.

[174] Powell M J D. On updating the inverse of a KKT matrix [J]. Numerical Linear Algebra and Optimization, ed. Yaxiang Yuan, Science Press (Beijing), 2004: 56-78.

[175] Conn A R, Scheinberg K, Vicente L N. Geometry of interpolation sets in derivative free optimization [J]. Mathematical Programming, 2008, 111(1): 141-172.

[176] Powell M J D. On the Lagrange functions of quadratic models that are defined by interpolation [J]. Optimization Methods and Software, 2001, 16(1-4): 289-309.

[177] Gould N I M, Orban D, Toint Ph L. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited [J]. ACM Transactions on Mathematical Software, 2003, 29 (4): 373-394.

[178] Moré J J, Garbow B S, Hillstrom K E. Testing unconstrained optimization software [J]. ACM Transactions on Mathematical Software, 1981, 7(1): 17-41.

[179] Conn A R, Gould N I M, Toint Ph L. Testing a class of methods for solving minimization problems with simple bounds on the variables [J]. Mathematics of Computation, 1988, 50 (182): 399-430.

[180] Lukšan L, Matonoha C, Vlcek J. Modified CUTE problems for sparse unconstrained optimization [R]. 2010.

[181] Li Y J, Li D H. Truncated regularized Newton method for convex minimizations [J]. Computational Optimization and Applications, 2009, 43: 119-131.

[182] Jarre F. EXPSUM Dataset [R/OL]. 2015. http://www.opt.uni-duesseldorf.de/~jarre/dot/f_cx.m.

[183] Conn A, Gould N, Lescrenier M, et al. Performance of a multifrontal scheme for partially separable optimization [M]//Advances in Optimization and Numerical Analysis. Dordrecht: Springer, 1994: 79-96.

[184] Toint Ph L. Some numerical results using a sparse matrix updating formula in unconstrained optimization [J]. Mathematics of Computation, 1978, 32(143): 839-851.

[185] Andrei N. An unconstrained optimization test functions collection [J]. Advanced Modeling and Optimization, 2008, 10(1): 147-161.

[186] Li G. The secant/finite difference algorithm for solving sparse nonlinear systems of equations [J]. SIAM Journal on Numerical Analysis, 1988, 25: 1181-1196.

[187] Robinson S M. Quadratic interpolation is risky [J]. SIAM Journal on Numerical Analysis, 1979, 16(3): 377-379.

[188] Xie P, Yuan Y. A derivative-free method using a new underdetermined quadratic interpolation model [J]. SIAM Journal on Optimization, 2025, 35(2): 1110-1133.

[189] Conn A R, Scheinberg K, Toint Ph L. Recent progress in unconstrained nonlinear optimization without derivatives [J]. Mathematical Programming, 1997, 79(1): 397-414.

[190] The MathWorks Inc. MATLAB (R2023b) [M]. Natick, Massachusetts, 2023.

[191] Lagarias J C, Reeds J A, Wright M H, et al. Convergence properties of the Nelder-Mead simplex method in low dimensions [J]. SIAM Journal on Optimization, 1998, 9(1): 112-147.

[192] Higham N J. Optimization by direct search in matrix computations [J]. SIAM Journal on Matrix Analysis and Applications, 1993, 14(2): 317-333.

[193] Kelley C T. Iterative Methods for Optimization [M]. Philadelphia: SIAM, 1999.

[194] Higham N J. The matrix computation toolbox [R]. 2002.

[195] Dinur I, Nissim K. Revealing information while preserving privacy [C]//Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. 2003: 202-210.

[196] Dwork C, Nissim K. Privacy-preserving datamining on vertically partitioned databases [C]// Annual International Cryptology Conference. Springer, 2004: 528-544.

[197] Dwork C, McSherry F, Nissim K, et al. Calibrating noise to sensitivity in private data analysis [C]//Halevi S, Rabin T. Theory of Cryptography: Third Theory of Cryptography Conference. Berlin: Springer, 2006: 265-284.

[198] Nissim K, Raskhodnikova S, Smith A. Smooth sensitivity and sampling in private data analysis [C]//Proceedings of the 39th Annual ACM Symposium on Theory of Computing. 2007: 75-84.

[199] Kasiviswanathan S P, Lee H K, Nissim K, et al. What can we learn privately? [J]. SIAM Journal on Computing, 2011, 40(3): 793-826.

[200] McSherry F, Talwar K. Mechanism design via differential privacy [C]//48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07). IEEE, 2007: 94-103.

[201] Wang Y, Hale M, Egerstedt M, et al. Differentially private objective functions in distributed cloud-based optimization [C]//2016 IEEE 55th Conference on Decision and Control (CDC). IEEE, 2016: 3688-3694.

[202] Liu J, Huang X, Liu J K. Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryption [J]. Future Generation Computer Systems, 2015, 52: 67-76.

[203] Kusner M, Gardner J, Garnett R, et al. Differentially private Bayesian optimization [C]// International Conference on Machine Learning. PMLR, 2015: 918-927.

[204] Deng G, Ferris M C. Adaptation of the UOBYQA algorithm for noisy functions [C]// Proceedings of the 2006 Winter Simulation Conference. Piscataway: IEEE, 2006: 312-319.

[205] Jamieson K G, Nowak R, Recht B. Query complexity of derivative-free optimization [C]// Pereira F, Burges C, Bottou L, et al. Advances in Neural Information Processing Systems: volume 25. New York: Curran Associates, Inc., 2012.

[206] Wilson J D, Wintucky E G, Vaden K R, et al. Advances in space traveling-wave tubes for NASA missions [J]. Proceedings of the IEEE, 2007, 95(10): 1958-1967.

[207] Levush B. The design and manufacture of vacuum electronic amplifiers: Progress and challenges [C]//2019 International Vacuum Electronics Conference (IVEC). IEEE, 2019: 1-5.

[208] Gould N I M, Orban D, Toint Ph L. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization [J]. Computational Optimization and Applications, 2015, 60: 545-557.

[209] Nocedal J, Yuan Y. Combining trust region and line search techniques [C]//In: Advances in Nonlinear Programming. Boston: Springer, 1998: 153-175.

[210] Khalil H K. Nonlinear Systems [M]. Upper Saddle River: Prentice Hall, 2002.

[211] Powell M J D. Least Frobenius norm updating of quadratic models that satisfy interpolation conditions [J]. Mathematical Programming, 2004, 100(1): 183-215.

Pengcheng Xie's Academic Publications During the Ph.D. Study

Pengcheng Xie's publications related to Chapter 2:

(1) **P. Xie**, Ya-xiang Yuan, "A new derivative-free method using an improved underdetermined quadratic interpolation model", *SIAM Journal on Optimization* (https://epubs.siam.org/doi/full/10.1137/23M1582023)

(2) **P. Xie**, Ya-xiang Yuan, "Least H^2 norm updating quadratic interpolation model function for derivative-free trust-region algorithms", *IMA Journal of Numerical Analysis*, drae106
https://doi.org/10.1093/imanum/drae106

(3) **P. Xie**, "A derivative-free trust-region method for optimization on the ellipsoid", *Journal of Physics: Conference Series*, 2620, 012007, 2023.
https://doi.org/10.1088/1742-6596/2620/1/012007

(4) Yangyi Ye, Lin Li, **P. Xie**, Haijun Yu, "An improved adaptive orthogonal basis deflation method for multiple solutions with applications to nonlinear elliptic equations in varying geometry", *Journal of Computational Mathematics*, 2025.
https://doi.org/10.4208/jcm.2505-m2024-0276

(5) Lin Li, Yuheng Zhou, **P. Xie (corresponding author)**, Huiyuan Li, "A spectral Levenberg – Marquardt-Deflation method for multiple solutions of semilinear elliptic systems", *Journal of Computational and Applied Mathematics*, Volume 475, 116998.
https://doi.org/10.1016/j.cam.2025.116998

(6) Shuoran Li, **P. Xie (corresponding author)**, et al., "Simulation of interaction of folded waveguide space traveling wave tubes with derivative-free mixedinteger based NEWUOA algorithm", in *2021 7th International Conference on Computer and Communications (ICCC)*, pp. 1215–1219, 2021.
https://doi.org/10.1109/ICCC54389.2021.9674410

(7) Qi Zhang and **P. Xie (corresponding author)**, On the relationship between Λ-poisedness in derivative-free optimization and outliers in local outlier factor. 2024.
https://arxiv.org/abs/2407.17529

Pengcheng Xie's publications related to Chapter 3:

(1) **P. Xie**, Ya-xiang Yuan, "Derivative-free optimization with transformed objective functions and the algorithm based on the least Frobenius norm updating quadratic model", *Journal of the Operations Research Society of China*, 13, 327–363 (2025).
https://doi.org/10.1007/s40305-023-00532-x.

(2) **P. Xie**, "Sufficient conditions for distance reduction between the minimizers

of non-convex quadratic functions in the trust region", *Journal of Computational and Applied Mathematics*, 453: 116146. 2025.
https://doi.org/10.1016/j.cam.2024.116146

Pengcheng Xie's publications related to Chapter 4:

(1) **P. Xie**, Ya-xiang Yuan, "A new two-dimensional model-based subspace method for large-scale unconstrained derivative-free optimization: 2D-MoSub", *Optimization Methods and Software*, accepted
https://doi.org/10.48550/arXiv.2309.14855

(2) **P. Xie**, Ya-xiang Yuan, "A derivative-free optimization algorithm combining line-search and trust-region techniques", *Chinese Annals of Mathematics, Series B*, vol. 44, no. 5, pp. 693–708, 2023.
https://doi.org/10.1007/s11401-023-0040-y