Les Cahiers du GERAD ISSN: 0711–2440

Model-based methods in derivative-free nonsmooth optimization

C. Audet,
W. Hare

G–2018–34

May 2018

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Model-based methods in derivative-free nonsmooth optimization

Charles Audet [a,b]
Warren Hare [c]

[a] *GERAD HEC Montréal, Montréal (Québec), Canada, H3T 2A7*

[b] *Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec), Canada, H3C 3A7*

[c] *Mathematics, University of British Columbia, Okanagan Campus, Kelowna (British Columbia), Canada V1V 1V7*

charles.audet@gerad.ca
warren.hare@ubc.ca

May 2018
Les Cahiers du GERAD
G–2018–34

Abstract: Derivative-free optimization (DFO) is the mathematical study of the optimization algorithms that do not use derivatives. One branch of DFO focuses on model-based DFO methods, where an approximation of the objective function is used to guide the optimization algorithm. Historically, model-based DFO has often assumed that the objective function is smooth, but unavailable analytically. However, recent progress has brought model-based DFO into the realm of nonsmooth optimization. In this chapter, we survey some of the progress of model-based DFO for nonsmooth optimization. We begin with some historical context on model-based DFO. From there, we discuss methods for constructing models of smooth functions and their accuracy. This leads to modelling techniques for nonsmooth functions and a discussion on several frameworks for model-based DFO for NSO. We conclude the paper with some of our opinions on profitable research directions in model-based DFO for NSO.

Acknowledgments: Hare's research was partially funded by the Natural Sciences and Engineering Research Council (NSERC) of Canada, Discover Grant #355571-2013.

Audet's research was partially funded by the Natural Sciences and Engineering Research Council (NSERC) of Canada, Discover Grant #2015-05311.

1 Introduction

In 1969, Winfield defended a Ph.D. thesis titled "Function and Functional Optimization by Interpolation in Data Tables" [92]. The associated paper appeared 4 years later in the *Journal of the Institute of Mathematics and its Applications* [93]. In it, Winfield presented a method to optimize a function using only function values. Winfield was not the first to consider the challenge of optimization using only function values (consider, for example, the influential papers by Hooke and Jeeves [52] or Nelder and Mead [67]). However, Winfield was the first to use the function values to build a model of the objective function and then use the model to guide the search for a new incumbent solution. It is now fifty years later, and this framework is the cornerstone of *model-based derivative-free optimization*.

Let us define *derivative-free optimization* (DFO) as the mathematical study of algorithms for continuous optimization that do not use first-order information (derivatives, gradients, directional derivatives, subgradients, etc.). Note that, we said that the algorithms do not use first-order information, which is quite different from saying that the objective function is nonsmooth. Also note, we use the term *mathematical study* to emphasize that research examines concepts such as proof of convergence, accuracy of stopping conditions, and rigourous analysis of numerical tests. This allows us to remove heuristic methods from our definition. In our opinion, when the mathematical analysis of an algorithm is provided, it is no longer a heuristic.

DFO methods have been successfully applied to widely different areas: oil production optimization problems [38, 43], molecular geometry [2, 60], helicopter rotor blade design [18, 19, 85], research on water resources [1, 40, 65, 63], alloy and process design [41, 42, 80]. Many other engineering applications and extensions are given in [4, 35].

Using our definition, DFO can be broadly split into two algorithmic styles: direct search methods, and model-based methods.

Direct search DFO methods work from an incumbent solution and examine a collection of trial points to seek improvement. If improvement is found, then the incumbent solution is updated; while if no improvement is found, then a step size parameter is decreased and a new collection of trial points is examined. The aforementioned works of Hooke and Jeeves [52] and Nelder and Mead [67] are classic examples of direct search methods.[1] Direct search methods have lead to many successful algorithms and been applied in many applications. However, the focus of this chapter is model-based DFO, so we simply refer the curious reader to a few detailed surveys on the subject [4, 58, 94] and the books [8, Part II] [29, Chpts 7 & 8].

Model-based DFO methods begin by approximating the objective function with a model function, and then use the model function to help guide optimization. In [93], the model is built through quadratic interpolation, and the information was used by minimizing the model over a trust-region. This has remained one of the most popular techniques in model-based DFO (see Table 1). However, as we shall explore in this paper, other ideas have recently emerged.

To understand model-based DFO methods, a first step is to study modelling techniques. Intuitively, we require techniques that produce "good" models. This can be interpreted many ways. Winfield's quadratic interpolation approach interprets "good" in the form of a 2nd-order Taylor-like error term: (under some conditions) there exists constants κ_f, κ_g, and κ_h such that

$$\begin{aligned} & |f(y) - Q(y)| \leq \kappa_f \Delta^3 && \text{for all } y \in B_\Delta(x), \\ & \|\nabla f(y) - \nabla Q(y)\| \leq \kappa_g \Delta^2 && \text{for all } y \in B_\Delta(x) \\ \text{and} \quad & \|\nabla^2 f(y) - \nabla^2 Q(y)\| \leq \kappa_h \Delta && \text{for all } y \in B_\Delta(x), \end{aligned}$$

where f is the true objective function, Q is a model based on quadratic interpolation, and $\Delta > 0$ is a scalar parameter which the optimizer controls (we shall explore details of this in Section 3). Obviously, one condition required for the above equations to hold is that $\nabla^2 f(y)$ is well-defined. This would apparently separate model-based DFO from the field of *nonsmooth optimization*.

[1] The Nelder-Mead method as presented in [67] is a heuristic. However, subsequent variants, including Tseng's fortified Nelder-Mead method [89] or Price, Coope, and Byatt's Nelder-Mead reset method [77], have proven convergence. This places the Nelder-Mead method into the realm of DFO.

Let us define *nonsmooth optimization* (NSO) as the study of algorithms for continuous optimization problems that are not everywhere differentiable. In particular, NSO problems are typically not differentiable at their minimizers (or maximizers), which makes convergence analysis based on Taylor-like approximations problematic. Derivatives and gradients are generalized through variational analysis to construct directional derivatives, subgradients, or other similar objects. Precise definitions of variational analysis object relevant to this paper will appear as required.

In order to merge the fields of model-based DFO and NSO, it is necessary to break away from the traditional Taylor-like error bounds, and pursue novel modelling techniques and analysis. In this paper, we demonstrate that the merging of model-based DFO and NSO is not just possible, but underway. We begin with a more complete background on model-based DFO (Section 2) and then move on to some basic methods for constructing and using models of smooth functions (Section 3). We also discuss ways to analyze their accuracy. The methods for constructing models of smooth functions will allow us to discuss modelling methods and their accuracy for nonsmooth functions, which we present in Section 4. We also provide further details on several frameworks that employ such models in model-based DFO for NSO. Section 5 concludes the paper with some of our opinions on profitable research directions in model-based DFO for NSO.

2 Historical overview

As mentioned in Section 1, model-based DFO dates back to at least 1969 [92]. However, we did not mention that Winfield's paper made very little impact on the field. In fact, the paper was largely ignored by the DFO community until Han and Liu [44] cited it in 2004 as a first contribution to the field.[2]

Despite the early beginning of model-based DFO, it was not until 1994 that model-based DFO received the spark it required to take on life. That spark came in the form of a fully analyzed and implemented model-based DFO method called COBYLA (Constrained Optimization BY Linear Approximation) [71]. Powell's COBYLA method uses linear interpolation to build a linear model the objective function, then minimizes the linear model over a trust-region. Over the course of the algorithm, model accuracy is improved until asymptotically it is equivalent to a 1st-order Taylor expansion. Thus, asymptotically the method becomes equivalent to steepest descent, and convergence is ensured [71] under some standard conditions.

Shortly after Powell's COBYLA, Conn and Toint [30] developed a similar method based on quadratic interpolation and a quadratic model function.[3] The use of a quadratic model means that the trust-region minimization can now take curvature into account and is no longer a reframing of steepest descent. However, the quadratic model comes at the price of making the model construction and the trust-region minimization more difficult. Like Powell, Conn and Toint's research included a proof of convergence, along with a fully implemented algorithm that was numerically tested. Combined, the papers of Powell and Conn and Toint launched model-based DFO as a field full of possibilities.

Motivated by these ideas, more researchers started exploring the ability to minimize an objective function using model-based DFO. In many cases, the development of a model-based DFO algorithm began from a classical algorithm for smooth optimization. Researchers began by showing that they could build a model of the true objective function that enjoys a property similar to a 1st or 2nd order Taylor-expansion (see Section 3). These models can then be used in place of the 1st or 2nd order Taylor-expansion. In order to show convergence, some mechanism is added to the algorithm to ensure that the model converges to the truth as the algorithm approaches a critical point. Some examples of algorithms that fit this description are provided in Table 1.

In examining Table 1, notice that the majority of methods follow the 2nd order trust-region framework described in [8, Chpt 10] [29, Chpt 10]. We include details on this framework in Section 3.5.

[2]Interestingly, Winfield's paper also appears in the references of DFO papers from 2002-2003 [58, 72, 73], but is not actually mentioned in the text of those articles.

[3]Conn and Toint's paper argues that there are 5 classes of DFO algorithms: finite-difference methods, pattern search methods, random sampling methods, successive one-dimension minimization methods, and model-based methods. Researchers have now unified pattern search methods, random sampling methods, and successive one-dimension minimization methods, as direct search methods. While finite-difference methods fall under our broad description of model-based methods.

Table 1: Model-based DFO algorithms that are closely related to classical smooth optimization algorithms. The first two algorithms are separated out, as they are algorithmic frameworks, not complete methods. The remaining algorithms are listed in order of first appearance in literature.

Framework	References		Smooth Equivalent
MBD	[8, Ch 10][29, Ch 9]		Gradient-based line-search methods
MBTR	[8, Ch 11][29, Ch 10]		2nd order Trust-region method
Algorithm	Ref.	Year	Smooth Equivalent
SQM	[92, 93]	1969	2nd order trust-region
unnamed	[61]	1975	Newton
COBYLA	[71]	1994	1st order trust-region
unnamed	[30]	1996	2nd order trust-region
"DFO"	[26]	2001	2nd order trust-region
UOBYQA	[72]	2002	2nd order trust-region
CONDOR	[14, 15]	2004	2nd order trust-region (for parallel computing)
BOOSTERS	[68, 69]	2005	2nd order trust-region (using radial basis functions)
NEWUOA	[75]	2006	2nd order trust-region (advances UOBYQA)
ORBIT	[90, 91]	2008	2nd order trust-region (using radial basis functions)
BOBYQA	[76]	2009	2nd order trust-region (advances UOBYQA to allow bound constraints)
DFTR	[28]	2009	2trust-region (1st and 2nd order convergence analysis)
CSV2	[17]	2013	2nd order trust-region
DFPP	[51, 46]	2014	Proximal point method
DEFT-FUNNEL	[82]	2015	SQP trust-region
CONORBIT	[79]	2017	2nd order trust-region for constrained optimization (adaptation of ORBIT)

2.1 Model-based DFO for nonsmooth functions

The proof of convergence of each algorithm in Table 1 requires a smoothness assumption on the objective function. First-order methods, such as [46, 51, 71], use approximate gradients in the algorithm, and proof of convergence requires that the approximate gradients asymptotically converge to the true gradients. This immediately means the objective function must be $\mathcal{C}^1$. Furthermore, assumptions that imply the approximate gradients asymptotically converge to the true gradients typically include that the gradients are locally Lipschitz continous (i.e., $f \in \mathcal{C}^{1+}$). Second-order methods (the rest of Table 1) use approximate Hessians, and require assumptions that ensure these approximations are suitably well-behaved. In some cases, it is sufficient to assume the approximate Hessians are bounded in norm. In these cases, $f \in \mathcal{C}^{1+}$ may be enough to ensure convergence. In other cases, convergence requires the approximate Hessians to asymptotically converge to the true Hessian, which typically requires $f \in \mathcal{C}^{2+}$.

More recently, some research has begun exploring the notion of solving nonsmooth optimization problems via model-based DFO. The first published research on this (to the best of the authors' knowledge) appeared in 2008 [11]. In this work, Bagirov, Karasözen and Sezer presented a discrete gradient derivative-free method for unconstrained nonsmooth optimization problems. To understand further, it is necessary to remind the reader of the definition of the subdifferential for locally Lipschitz functions. To do so, we recall that if a function is locally Lipschitz, then it is differentible almost everywhere.

Definition 2.1 (Subdifferential) *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz function. Let $D(f)$ be the set of points where f is differentiable. The* subdifferential *of f at the point $x \in \mathbb{R}^n$ is defined*

$$\partial f(x) = \text{conv}\{v \in \mathbb{R}^n : \textit{there exists } y^i \in D(f), \textit{ such that } y^i \to x, \nabla f(y^i) \to v\},$$

where conv *denotes the convex hull.*

At a given incumbent solution x^k, Bagirov, Karasözen and Sezer's approach constructs a large number of approximate gradients at points $y^{k,i}$ near x^k. The convex hull of the approximate gradients can then be used as an approximation of the subdifferential. The resulting approximation was used within a conjugate subgradient style algorithm. A similar approach [57] was published in 2010, where Kiwiel used a large number of approximate gradients to construct an approximate subdifferential and used the approximate subdifferential in a gradient sampling style algorithm.

Bagirov, Karasözen and Sezer's algorithm was implemented and tested under the name DGM. While the numerical tests show very high accuracy, they do not discuss the number of function evaluations required in each iteration. Kiwiel's work contains no numerical testing the result.

In DFO, the speed of convergence is typically measured in function evaluations, not time [64, 13]. This is done because, while academic test problems are extremely fast to compute, in many real-world applications each function evaluation takes a notable amount of time. For example, in [95], the authors consider the planning of bioequivalence studies in the pharmaceutical industry. In this problem, each function evaluation launches a Monte-Carlo simulation and requires approximately 3 minutes to complete on a cluster of 6 computers. Thus, an algorithm that requires one million of function evaluations, would require almost six years to complete. In in [95], the authors limit the number of function evaluations to 100 in order to get a solution within a 5 hour time period.

With this in mind, let us reconsider the approaches of Bagirov, Karasözen and Sezer and of Kiwiel. Each approximate gradient requires $n+1$ or $2n$ function evaluations (the first being from [11], the second from [57]). So, the natural question is, how many approximate gradients are needed to provide a reasonable approximation to the subdifferential?

A heuristic answer to this was given by Burke, Lewis, and Overton [22], who suggested $2n$ gradients be sampled each iteration. Another answer was given in [9], where the authors examined the question of how many exact gradients are needed to reconstruct a polyhedral subdifferential? In $\mathbb{R}^2$, the answer is 3 times the number of edges of the polytope. In $\mathbb{R}^n$, the precise answer is still unknown, but does depend on the number of faces.

The planning of bioequivalence studies problem from [95] has only $n = 4$ optimization variables. Therefore, each iteration of the approach of [11] or [57] would require $2n^2 = 32$ function evaluations, which corresponds to more than 1.5 hour. So, only three complete iterations could be completed in the 5 hour time period. A similar problem with $n = 8$ and 6 minute function evaluations would not even complete a single iteration in 5 hours. Clearly, these methods are intractable for even small dimension problem, unless the time per function evaluation is negligible.

Happily, this is not the end of the story for model-based DFO of nonsmooth problems. Recognizing the difficulty of approximating the subdifferential of an arbitrary function, researchers have turned their attention to approximating the subdifferential of a structured function. In [47, 48], the authors considered finite-max functions, i.e., functions of the form $f(x) = \max\{F_i(x) : i = 1, 2, \ldots, N_f\}$ and each $F_i \in \mathcal{C}^2$. Finite-max functions do arise naturally in DFO problems; for example, an application in seismic retrofitting design is examined in [16].

The finite-max structure allows for a simple formula for the subdifferential:

$$f(x) = \max\{F_i(x) : i = 1, 2, \ldots, N_f\}, F_i \in \mathcal{C}^2, \text{ implies that } \partial f(x) = \operatorname{conv}\{\nabla F_i(x) : F_i(x) = f(x)\}.$$

Using this formula it is possible to approximate the subdifferential of a finite-max function using only $n+1$ function evaluations, which allows for the application of these methods in practice. The details of this construction are presented in Section 4.

Another structured function was studied in [59]. In that paper, Larson, Menickelly, and Wild, considered problems where the objective function takes the form of an $\ell - 1$ norm: $f(x) = \sum_{i=1}^{N_f} |F_i(x)|$ and each $F_i \in \mathcal{C}^2$.

In this case we have,

$$f(x) = \sum_{i=1}^{N_f} |F_i(x)|, F_i \in \mathcal{C}^2, \text{ implies that } \partial f(x) = \sum_{i=1}^{N_f} \alpha_i \nabla F_i(x),$$

where

$$\alpha_i = \begin{cases} 1 & F_i(x) > 0, \\ [-1, 1] & F_i(x) = 0, \\ -1 & F_i(x) > 0. \end{cases}$$

Once again, using the simpler formula allows the ability to approximate the subdifferential using just $n + 1$ function evaluations. The technique has recently been extended to piecewise-linear functions [56].

In [47, 48, 56, 59], the authors focused on line-search and trust-region method. However, the subdifferential approximation techniques therein can be employed with other algorithms. For example, in [12], the comirror algorithm for nonsmooth optimization is reexamined in light of the subdifferential approximation ideas in [48]. Another example is the recent work [49], where the authors examine the $\mathcal{VU}$ algorithm from [62] in the setting of DFO.

Table 2 summarizes the current model-based DFO algorithms for nonsmooth optimization.

Table 2: Model-based DFO algorithms that are designed for nonsmooth objective functions.

Algorithm	Ref.	Year	Smooth Equivalent
DGM	[11]	2008	conjugate subgradient
WASG	[47]	2012	gradient sampling
RAGS	[48]	2013	subgradient descent with line search
$\text{DFO}_{\text{comirror}}$	[12]	2015	comirror
CMS, GDMS, DMS, SMS	[59]	2016	manifold sampling
MS_4PL	[56]	preprint	manifold sampling
DFO_VU	[49]	preprint	$\mathcal{VU}$ algorithm

In all of these cases, the trick to a successful model-based DFO algorithm for nonsmooth optimization is to focus on structured functions where the subdifferential can be approximated using a reasonable number of function evaluations. Section 4 returns to this statement, and discusses model building techniques for nonsmooth functions.

2.2 Model-based DFO used within direct search

Before moving on to more technical details of model-based DFO, it is worth a small detour the realm of direct search methods. In particular, we discuss of some of the work that has considered applying model-based methods as subroutines in direct search methods.

Some direct search methods, such as generalized Pattern Search (GPS [88]) and Mesh Adaptive Direct Search (MADS [6]), are of a search-poll paradigm [20]. The poll step is a local exploration intended for theoretical and practical local convergence. The search step is a global exploration in the space of variables, whose purpose is to identify promising regions. Model-based strategies have been incorporated into the search step for a better global exploration of the space of variables, as well as into the poll step to improve the efficiency of the local exploration. Table 3 lists some direct search methods that exploit simplex gradients or even simplex Hessians. This information is used in the search step with line search strategies, or with trust-region methods, and is also used in the poll step for opportunistic termination.

The algorithms listed in the top half of the table are for unconstrained or bound-constrained optimization. The algorithms on the bottom half are for problems with nonsmooth constraints, and they construct models of the objective function and of each constraint, or penalize constraints into the objective function. As with the previous tables, the last column indicates a connection with more classical smooth algorithms. But here, we are taking about similarities rather than equivalences.

Table 3: Uses of model-based DFO techniques with a direct search framework.

Algorithm	Ref.	Year	Smooth algorithm with similarities to model-based portion
Unconstrained or bound-constrained optimization			
IMFIL	[21]	1998	Gradient-based search
GPS	[36]	2007	Gradient-based line-search methods
SNOBFIT	[53]	2008	Sequential quadratic programming
GPS & MADS	[33]	2008	Gradient-based line-search methods Newton with diagonal Hessian approximation
SID-PSM	[34]	2010	Trust-region
MADS	[24]	2013	Trust-region
Constrained optimization			
DFN	[39]	2014	Projected linesearch with penalty function
MADS	[10]	2014	Line search
QPMADS	[23]	2015	Directional derivative-based Hessian update, linear models of the constraints
PBTR	[5]	2016	Trust-region
GOSAC	[66]	2017	Cubic radial basis function
LOWESS	[86]	2018	Locally weighted regression
MADS	[3]	2018	ℓ_∞-norm trust-region

3 Building models of smooth functions

In order to study the mathematical details of model-based DFO for smooth optimization it is valuable to provide some background on model-based DFO for smooth optimization. As mentioned, the first step in model-based DFO is the construction of a model.

3.1 Linear interpolation and the simplex gradient

For a smooth function, the simplest model is that of linear interpolation. The following definition presents conditions on the points used to build such model.

Definition 3.1 (Poised for linear interpolation) *The set* $\mathbb{Y} = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$, *is* poised[4] for linear interpolation *if the* $(n+1) \times (n+1)$ *matrix* $\begin{bmatrix} \mathbf{1} & Y^\top \end{bmatrix}$ *is invertible, where* $Y = \begin{bmatrix} y^0 & y^1 & \ldots & y^n \end{bmatrix}$ *and* $\mathbf{1} \in \mathbb{R}^{n+1}$ *is the vector of all ones.*

The next definition defines the linear interpolation function.

Definition 3.2 (Linear interpolation function) *Let* $\mathbb{Y} = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$ *be poised for linear interpolation and* $f : \mathbb{R}^n \mapsto \mathbb{R}$. *Then the* linear interpolation function of f over $\mathbb{Y}$ *is*

$$L_Y(x) := \alpha_0 + \alpha^\top x$$

where (α_0, α) *is the unique solution to* $\begin{bmatrix} \mathbf{1} & Y^\top \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha \end{bmatrix} = f(\mathbb{Y})$.

The linear interpolation function provides a first-order approximation of the function f. As such, the gradient of the linear interpolation function can be used to check (approximately) descent directions and first-order stopping conditions.

Recall that a simplex in $\mathbb{R}^n$ is a bounded polytope with a non-empty interior and exactly $n+1$ vertices.[5] It is fairly easy to show that $\mathbb{Y} = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$, is poised for linear interpolation if and only if $\text{conv}(\mathbb{Y})$ is a simplex [8, Prop 9.1]. As such, the gradient of a linear interpolation model has been named the *simplex gradient*.[6]

[4] The term *poised* was introduced by Sauer and Xu [83] in 1995 and imported into the context of DFO by Conn and Toint [30] in the following year.

[5] The term *simplex* was first used by Schoute [84] back in 1902. Since then, the simplices have arisen in numerous areas of optimization, largely because they are the simplest full dimensional polytope.

[6] The *simplex gradient* was introduced by Kelley [54] to monitor the performance of a modified Nelder-Mead algorithm.

Definition 3.3 (Simplex gradient) *Let $f : \mathbb{R}^n \mapsto \mathbb{R}$, and let $\mathbb{Y} = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$ be poised for linear interpolation. The* simplex gradient of f over $\mathbb{Y}$*, denoted $\nabla_S f(\mathbb{Y})$, is the gradient of the linear interpolation function L_Y of f over $\mathbb{Y}$:*

$$\nabla_S f(\mathbb{Y}) := \nabla L_Y(x) = \alpha.$$

The quality of the linear interpolation model, and the simplex gradient, are analyzed in the following theorem.

Theorem 3.4 (Linear interpolation error bounds) *Let $x \in \mathbb{R}^n$ and $f \in \mathcal{C}^{1+}$ on $B_{\bar{\Delta}}(x)$ with constant K. Let $\mathbb{Y}$ be poised for linear interpolation with $y^0 = x$ and $\Delta = \Delta(\mathbb{Y}) \leq \bar{\Delta}$. Define*

$$\widehat{L} := \frac{1}{\Delta}\left[y^1 - y^0 \quad y^2 - y^0 \quad \ldots \quad y^n - y^0\right].$$

Then the linear interpolation function $L(y) = \alpha_0 + \alpha^\top y$ satisfies

$$\|f(y) - (\alpha_0 + \alpha^\top y)\| \leq \left(\frac{1}{2}K(1 + \sqrt{n}\|\widehat{L}^{-1}\|)\right)\Delta^2 \quad \text{for all } y \in B_\Delta(x). \tag{1}$$

Then the simplex gradient $\alpha = \nabla_S f(\mathbb{Y})$ satisfies

$$\|\nabla f(y) - \alpha\| \leq \left(\frac{1}{2}K\sqrt{n}\|\widehat{L}^{-1}\|\right)\Delta. \tag{2}$$

Furthermore,

$$\|\nabla f(y) - \alpha\| \leq \left(\frac{1}{2}K(2 + \sqrt{n}\|\widehat{L}^{-1}\|)\right)\Delta \quad \text{for all } y \in B_\Delta(x). \tag{3}$$

Proof. Proofs can be found in [29, Chpt 2], [55, Chpt 5], and [8, Chpt 9], for example. □

The complexity of computing simplex gradients has recently been studied in [31].

3.2 Fully linear and fully quadratic models

Theorem 3.4 shows that linear interpolation provides an accuracy similar to that of a first-order Taylor expansion. In [91], Wild and Shoemaker formalized this property under the name *fully linear models*, although they give credit to Conn, Scheinberg and Vicente [27] for inspiring the terminology. Indeed, the latter authors use the terms *fully linear, quadratic* and even *fully cubic* but do not provide a formal definition. They further introduced the name *fully quadratic models* for models that provide an accuracy similar to that of a second-order Taylor expansion. These terms provide a convenient language for model-based DFO, so we now take a brief interlude from constructing models to formalize these definitions. We begin with fully linear.

Definition 3.5 (Class of fully linear models) *Given $f \in \mathcal{C}^1$, $x \in \mathbb{R}^n$, and $\bar{\Delta} > 0$ we say that $\{\tilde{f}_\Delta\}_{\Delta \in (0,\bar{\Delta}]}$ is a* class of fully linear models *of f at x parameterized by Δ if there exists a pair of scalars $\kappa_f(x) > 0$ and $\kappa_g(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

$$\begin{array}{rrcll} & |f(y) - \tilde{f}_\Delta(y)| & \leq & \kappa_f(x)\Delta^2 & \text{for all } \ y \in B_\Delta(x) \\ \text{and} & \|\nabla f(y) - \tilde{g}_\Delta(y)\| & \leq & \kappa_g(x)\Delta & \text{for all } \ y \in B_\Delta(x) \end{array}$$

where $\tilde{g}_\Delta = \nabla \tilde{f}_\Delta$.

The previous definition involves classes of models at a given x. Let us now generalize the definition to classes that adapt to any $x \in \mathbb{R}^n$.

Definition 3.6 (Fully linear models) *Let $f \in \mathcal{C}^1$. We say that $\tilde{f}_\Delta$* are fully linear models of *f to mean that, given any $x \in \mathbb{R}^n$ and $\bar{\Delta} > 0$, there exists a class $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ of fully linear models of f at x parameterized by Δ.*

We say that $\tilde{f}_\Delta$ are fully linear models of *f with constants κ_f and κ_g to mean that, given any $x \in \mathbb{R}^n$ and $\bar{\Delta} > 0$, there exists a class $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ of fully linear models of f at x parameterized by Δ and the scalars $\kappa_f(x)$ and $\kappa_g(x)$ can be taken as the constants: $\kappa_f(x) = \kappa_f$ and $\kappa_g(x) = \kappa_g$ (independent of x).*

Fully linear models provide a linear level of accuracy for approximating the function. If a higher accuracy is desired, then we demand *fully quadratic models*.

Definition 3.7 (Class of fully quadratic models) *Given $f \in \mathcal{C}^2$, $x \in \mathbb{R}^n$, and $\bar{\Delta} > 0$ we say that $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a* class of fully quadratic models *of f at x parameterized by Δ if there exists scalars $\kappa_f(x) > 0$, $\kappa_g(x) > 0$, and $\kappa_h(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

$$\begin{array}{rrcll} & |f(y) - \tilde{f}_\Delta(y)| & \leq & \kappa_0(x)\Delta^3 & \text{for all } \ y \in B_\Delta(x) \\ & \|\nabla f(y) - \tilde{g}_\Delta(y)\| & \leq & \kappa_g(x)\Delta^2 & \text{for all } \ y \in B_\Delta(x) \\ \text{and} & \|\nabla^2 f(y) - \tilde{h}_\Delta(y)\| & \leq & \kappa_h(x)\Delta & \text{for all } \ y \in B_\Delta(x) \end{array}$$

where $\tilde{g}_\Delta = \nabla\tilde{f}_\Delta$ and $\tilde{H}_\Delta = \nabla^2\tilde{f}_\Delta$.

Again, we generalize to any $x \in \mathbb{R}^n$.

Definition 3.8 (Fully quadratic models) *Let $f \in \mathcal{C}^2$.*

We say that $\tilde{f}_\Delta$ are fully quadratic models of *f to mean that, given any $x \in \mathbb{R}^n$ and $\bar{\Delta} > 0$, there exists a class $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ of fully quadratic models of f at x parameterized by Δ.*

We say that $\tilde{f}_\Delta$ are fully quadratic models of *f with constants κ_f, κ_g, and κ_h to mean that, given any $x \in \mathbb{R}^n$ and $\bar{\Delta} > 0$, there exists a class $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ of fully quadratic models of f at x parameterized by Δ and the scalars $\kappa_f(x)$, $\kappa_g(x)$, and $\kappa_h(x)$ can be taken as the constants: $\kappa_f(x) = \kappa_f$, $\kappa_g(x) = \kappa_g$, and $\kappa_h(x) = \kappa_h$ (independent of x).*

Basically, fully linear models enjoy a 1st-order Taylor-like behaviour and fully quadratic models enjoy a 2nd-order Taylor-like behaviour.

The value of this terminology is that it removes the need to define the model building process within a model-based DFO algorithm. Instead, algorithms can begin with the assumption that a class of fully linear or quadratic models is available, and then prove convergence using Definition 3.5 or 3.7 as appropriate. We will return to this in Subsection 3.5, but for now we return attention to the construction of models.

3.3 The generalized simplex gradient

One weakness of the simplex gradient is its dependance on the simplex. In particular, $\mathbb{Y}$ must contain exactly well-poised $n+1$ points. Custódio et al. [33] and Regis [78] present a more general framework that may be applied when the number of points is not $n+1$. Suppose $\mathbb{Y}$ contains $m+1$ elements and consider the linear system

$$L^\top \alpha \quad = \quad \delta^{f(\mathbb{Y})} \tag{4}$$

where $L = \begin{bmatrix} y^1 - y^0 & y^2 - y^0 & \dots & y^m - y^0 \end{bmatrix} \in \mathbb{R}^{n\times m}$ and

$$\delta^{f(\mathbb{Y})} = \begin{bmatrix} f(y^1) - f(y^0) & f(y^2) - f(y^0) & \dots & f(y^m) - f(y^0) \end{bmatrix}^\top \in \mathbb{R}^n.$$

If $m = n$ and if $\mathbb{Y}$ is a simplex, then the system is *determined* and the solution α to Equation (4) is unique and is the simplex gradient from Definition 3.3. If $m < n$, then the system is *underdetermined*, so has an

infinite set of solutions. This can be resolved by seeking the solution minimal norm:

$$\text{if } m < n, \text{ then solve } \alpha \in \operatorname*{argmin}_{\alpha \in \mathbb{R}^n} \{\|\alpha\|^2 : L^\top \alpha = \delta^{f(\mathbb{Y})}\}.$$

If $m > n$, then the system is *overdetermined*, so may not have a solution. This can be resolved by seeking the least square solution

$$\text{if } m < n, \text{ then solve } \alpha \in \operatorname*{argmin}_{\alpha \in \mathbb{R}^n} \{\|L^\top \alpha - \delta^{f(\mathbb{Y})}\|^2\}.$$

In all three cases, the solution can be written as $\alpha = (L^\dagger)^\top \delta^{f(\mathbb{Y})}$, where $L^\dagger$ is the Moore-Penrose pseudoinverse of the matrix L [33, 78].

Since a the resulting generalization no longer requires $\mathbb{Y}$ to form a simplex, the term simplex gradient no longer seems appropriate.

Definition 3.9 (Generalized simplex gradient) *Let $f : \mathbb{R}^n \mapsto \mathbb{R}$, and let $\mathbb{Y} = \{y^0, y^1, \ldots, y^k\} \subset \mathbb{R}^n$ be an ordered set with $m \geq 1$. The* generalized simplex gradient of f over $\mathbb{Y}$, *denoted $\nabla_S f(\mathbb{Y})$, is given by*

$$\nabla_S f(\mathbb{Y}) := (L^\dagger)^\top \delta^{f(\mathbb{Y})},$$

where

$$L = \begin{bmatrix} y^1 - y^0 & y^2 - y^0 & \ldots & y^m - y^0 \end{bmatrix} \quad \text{and} \quad \delta^{f(\mathbb{Y})} = \begin{bmatrix} f(y^1) - f(y^0) \\ f(y^2) - f(y^0) \\ \vdots \\ f(y^m) - f(y^0) \end{bmatrix}.$$

The above definition leads to a generalization of Equation (2).

Theorem 3.10 (Generalized simplex gradient error bounds) *Let $x \in \mathbb{R}^n$ and $f \in \mathcal{C}^{1+}$ on $B_{\bar{\Delta}}(x)$ with constant K. Let $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\}$ with $y^0 = x$ and $\Delta = \Delta(\mathbb{Y}) \leq \bar{\Delta}$. Suppose $\mathbb{Y}$ is* poised *in the sense that $Y = [y^0 \; y^1 \; \ldots \; y^m]$ is full-rank. Let $\widehat{U}(\mathbb{Y})\widehat{\Sigma}(\mathbb{Y})\widehat{V}(\mathbb{Y})^\top$ be the* reduced *singular-value decomposition*[7] *of $\widehat{L}^\top$ and define $\tilde{V}(\mathbb{Y}) = \begin{cases} I_n & \text{if } m \geq n \\ \widehat{V}(\mathbb{Y}) & \text{if } m < n. \end{cases}$*

Then, the generalized simplex gradient $\nabla_S f(\mathbb{Y})$ satisfies

$$\|\tilde{V}(\mathbb{Y})^\top \left[\nabla f(y^0) - \nabla_S f(\mathbb{Y})\right]\| \leq \left(\frac{1}{2} K \sqrt{m} \|\widehat{\Sigma}(\mathbb{Y})^{-1}\|\right) \Delta. \tag{5}$$

Consequently,

$$\|\tilde{V}(\mathbb{Y})^\top \left[\nabla f(y) - \nabla_S f(\mathbb{Y})\right]\| \leq \left(\frac{1}{2} K (2 + \sqrt{m} \|\widehat{\Sigma}(\mathbb{Y})^{-1}\|)\right) \Delta \quad \text{for all } y \in B_\Delta(x). \tag{6}$$

Proof. Proof of Equation (5) can be found in [78, Cor 1].[8] Proof of Equation (6) is a trivial adaptation of the proof of Equation (9.6) in [8, Thm 9.5]. □

Comparing Theorem 3.10 to Theorem 3.4 we consider three cases.

i) ($m = n$) If $m = n$, then we are in the determined case. The statement, "$Y = [y^0 \; y^1 \; \ldots \; y^m]$ is full-rank" is exactly equivalent to $\mathbb{Y}$ is poised for linear interpolation [8, Prop 9.1]. The matrix $\tilde{V}(\mathbb{Y})^\top$ is the identity, and $\|\widehat{L}(\mathbb{Y})^{-1}\| = \|\widehat{\Sigma}(\mathbb{Y})^{-1}\|$. Thus, the gradient error in Theorem 3.4 is reproduced by Theorem 3.10.

[7] The reduced SVD of the matrix $\widehat{L}^\top \in \mathbb{R}^{m \times n}$ of rank r is such that $\widehat{U}(\mathbb{Y}) \in \mathbb{R}^{m \times r}$ is an orthonormal basis for the column space, $\widehat{\Sigma}(\mathbb{Y}) \in \mathbb{R}^{r \times r}$ is a nonsingular diagonal matrix with positive entries, and $\widehat{V}(\mathbb{Y})^\top \in \mathbb{R}^{n \times r}$ is an orthonormal basis for the row space.

[8] Note that this result is presented in [33, Thm 2.1] and in [32, Thm 4.1.1] without proof.

ii) $(m < n)$ If $m < n$, then we are in the underdetermined case. The statement "Y is full-rank" now means that $\text{rank}(Y) = m + 1$. In this case $\tilde{V}(\mathbb{Y})^\top = \widehat{V}(\mathbb{Y})^\top$ and the gradient error in Theorem 3.4 is not reproduced.

Indeed, consider an example with $n = 2$ and $m = 1$ on $f(x) = \beta(x_1 + x_2)$. Define the sample set $\mathbb{Y} = \{\Delta e_1, \Delta e_2\}$, where e_i are the coordinate vectors and $\Delta > 0$. In this case, $L = [\Delta(e_1 - e_2)] = [\Delta \quad -\Delta]^\top$ and $\delta^{f(\mathbb{Y})} = [0]$. Therefore, the generalized simplex gradient is $\nabla_S f(\mathbb{Y}) = 0$, regardless of Δ. As $\nabla f(x) = [\beta \quad \beta]^\top$, $\|\nabla f(y^0) - \nabla_S f(\mathbb{Y})\| = \sqrt{2}\beta$, regardless of Δ. So the gradient approximation will never become accurate. To understand the error bound, note that the reduced singular-value decomposition of $\widehat{L}^\top = [1 \quad -1]$ is

$$\widehat{U} = [1], \quad \widehat{\Sigma} = [\sqrt{2}], \quad \widehat{V}^\top = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \end{bmatrix}.$$

Now examine

$$\left\| \tilde{V}(\mathbb{Y})^\top \left[\nabla f(y^0) - \nabla_S f(\mathbb{Y}) \right] \right\| = \left\| \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} \beta \\ \beta \end{bmatrix} \right\| = 0$$

The effect of $\tilde{V}(\mathbb{Y})^\top$ in the error bound is to project the vectors onto the subspace spanned by the matrix $\widehat{L}^\top$. Thus, the error bound is saying, the approximate gradient is sufficient to provide approximate directional derivatives in directions parallel to the $\text{span}(\widehat{L}^\top)$. In directions not parallel to $\text{span}(\widehat{L}^\top)$, the directional derivatives created using $\nabla_S f(\mathbb{Y})$ are uncontrolled.

iii) $(m > n)$ If $m > n$, then we are in the overdetermined case. The matrix $\tilde{V}(\mathbb{Y})^\top$ is the identity, so we have a direct bound on the error in the gradient approximation, similar but not the same as Theorem 3.4. In particular, notice the error bound contains a $\sqrt{m}$, instead of the $\sqrt{n}$ in Theorem 3.4. This implies that as the number of sample points increases the error may actually get worse.

Note, in case (i) and (iii) above, we carefully state that we achieve a bound on the error in the gradient approximation. This is not quite equivalent to creating a fully linear model. Fully linear models also require the models to approximate function values and a error bound on the quality of those approximations. Conveniently however, a good gradient approximation along with the exact function value is sufficient to create a fully linear model.

Proposition 3.11 *Suppose $x \in \mathbb{R}^n$, $\bar{\Delta} > 0$, $f \in \mathcal{C}^{1+}$ with constant K on $B_{\bar{\Delta}}(x)$. Suppose that the model functions $\tilde{f}_\Delta \in \mathcal{C}^{1+}$ with constant $\tilde{K}$ on $B_{\bar{\Delta}}(x)$. Suppose there exists $\kappa_g > 0$ such that the gradients $\{\nabla \tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ satisfy*

$$\|\nabla f(y) - \nabla \tilde{f}_\Delta(y)\| \le \kappa_g(x)\Delta \qquad \text{for all} \quad y \in B_\Delta(x).$$

Define $\hat{f}_\Delta = \tilde{f}_\Delta - \tilde{f}_\Delta(x) + f(x)$. Then $\{\hat{f}_\Delta\}$ is a class of fully linear models of f at x.

In particular, if $\tilde{f}_\Delta(x) = f(x)$ for every $\Delta \in (0, \bar{\Delta}]$, then $\{\tilde{f}_\Delta\}$ is a class of fully linear models.

Proof. We need to show the existence of $\kappa_f(x)$ such that

$$|f(y) - \hat{f}_\Delta(y)| \le \kappa_f(x)\Delta^2 \qquad \text{for all} \quad y \in B_\Delta(x).$$

Since that $\hat{f}_\Delta(x) = f(x)$, we have that

$$\begin{aligned} |f(y) - \hat{f}_\Delta(y)| \quad \le \quad & |f(y) - f(x) - \nabla f(x)^\top (y - x)| \\ & + |\nabla f(x)^\top (y - x) - \nabla \hat{f}_\Delta(x)^\top (y - x)| \\ & + |\hat{f}_\Delta(x) + \nabla \hat{f}_\Delta(x)^\top (y - x) - \hat{f}_\Delta(y)|. \end{aligned} \tag{7}$$

Applying $f \in \mathcal{C}^{1+}$ and $\tilde{f}_\Delta \in \mathcal{C}^{1+}$, we note that $\hat{f}_\Delta \in \mathcal{C}^{1+}$, and therefore the first-order Taylor approximation error bound holds [8, Lem 9.4]. In particular,

$$\begin{array}{rcccll} |f(y) - f(x) - \nabla f(x)^\top (y - x)| & \le & \frac{1}{2}K\|y - x\|^2 & \le & \frac{1}{2}K\Delta^2 & \text{for all} \quad y \in B_\Delta(x), \text{ and} \\ |\hat{f}_\Delta(x) + \nabla \hat{f}_\Delta(x)^\top (y - x) - \hat{f}_\Delta(y)| & \le & \frac{1}{2}\tilde{K}\|y - x\|^2 & \le & \frac{1}{2}\tilde{K}\Delta^2 & \text{for all} \quad y \in B_\Delta(x). \end{array}$$

Examining the middle term of inequality (7) and using our gradient error bound, we find that

$$\begin{array}{rcll} |\nabla f(x)^\top (y-x) - \nabla \hat{f}_\Delta(x)^\top (y-x)| & \leq & \|\nabla f(x) - \nabla \hat{f}_\Delta(x)\| \|y-x\| & \\ & \leq & \kappa_g(x)\Delta \|y-x\| & \text{for all } \ y \in B_\Delta(x) \\ & \leq & \kappa_g(x)\Delta^2 & \text{for all } \ y \in B_\Delta(x). \end{array}$$

Summing these approximations, we see that $\kappa_f(x) = K + \tilde{K} + \kappa_g(x)$ satisfies the desired inequality. □

3.4 Quadratic models

The framework of generalized simplex gradients, combined with Proposition 3.11, provides a straight-forward way to generate fully linear models. However, the error bound in Theorem 3.10 includes a "$\sqrt{m}$" term, which suggests that using more points will not necessarily improve model quality. Moreover, when the number of sample points becomes sufficiently large, it may become possible to build quadratic models.

For quadratic models, we consider model functions of the form

$$Q(x) = \alpha_0 + \alpha^\top x + \frac{1}{2} x^\top H x, \quad \text{with } H = H^\top.$$

As such, we seek $\alpha_0 \in \mathbb{R}$, $\alpha \in \mathbb{R}^n$, and $H \in \mathbb{R}^{n \times n}$. Enforcing $H = H^\top$ implies we need $1 + n + n(n+1)/2 = \frac{1}{2}(n+1)(n+2)$ well-poised sample points. We next formalize the definition of being poised for quadratic interpolation.

Definition 3.12 (Poised for quadratic interpolation) *The set* $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\} \subset \mathbb{R}^n$ *with* $m = \frac{1}{2}(n+1)(n+2) - 1$, *is* poised for quadratic interpolation *if the system*

$$\alpha_0 + \alpha^\top y^i + \frac{1}{2}(y^i)^\top H y^i = 0, \quad i = 0, 1, 2, \ldots m$$

has a unique (trivial) solution for α_0, α, *and* $H = H^\top$.

Being poised for linear or for quadratic interpolation has a nice geometrical interpretation. A linear system of equations $\alpha_0 + \alpha^\top y^i = f(y^i)$, $i \in \{0, 1, \ldots, n\}$ has a unique solution unless the points are collinear. That is, the set $\mathbb{Y} = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$ is **not** poised for linear interpolation if and only if the points all lie on the level surface of a single nontrivial linear function. Similarly, a linear system of equations $\alpha_0 + \alpha^\top y^i + \frac{1}{2}(y^i)^\top H y^i = f(y^i)$, $i \in \{0, 1, \ldots, \frac{1}{2}(n+1)(n+2) - 1\}$ has a unique solution unless the points all lie on the level surface of a single nontrivial quadratic function. That is, the set $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\} \subset \mathbb{R}^n$, $m = \frac{1}{2}(n+1)(n+2) - 1$, is **not** poised for quadratic interpolation if and only if the points all lie on the level surface of a single nontrivial quadratic function. Figure 1 illustrates the notion of being poised for linear or for quadratic interpolation in the case where $n = 2$. The figure illustrates that it can be difficult to visually confirm whether or not a set is poised for quadratic interpolation. Fortunately, the algebraic test is still straight-forward.

Given a set that is poised for quadratic interpolation, we define the quadratic interpolation function.

Definition 3.13 (Quadratic interpolation function) *Let* $f : \mathbb{R}^n \mapsto \mathbb{R}$, *and let* $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\} \subset \mathbb{R}^n$ *with* $m = \frac{1}{2}(n+1)(n+2) - 1$, *be poised for quadratic interpolation. Then the* quadratic interpolation function *of* f *over* $\mathbb{Y}$ *is*

$$Q_Y(x) := \alpha_0 + \alpha^\top x + \frac{1}{2} x^\top H x$$

where $(\alpha_0, \alpha, H = H^T)$ *is the unique solution to*

$$\alpha_0 + \alpha^\top y^i + \frac{1}{2}(y^i)^\top H y^i = f(y^i), \quad i = 0, 1, 2, \ldots m.$$

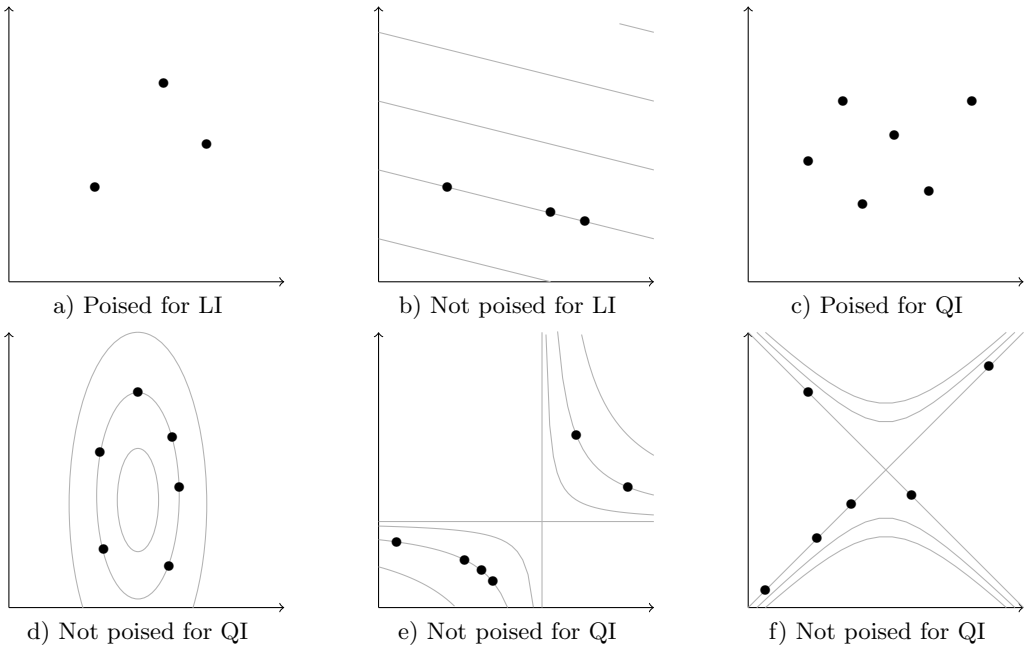

Figure 1: Poised and non-poised sets for interpolation (Figure inspired from [8]).

In [27], Conn, Scheinberg and Vicente show that quadratic interpolation resulted in a class of fully quadratic models.

Theorem 3.14 (Quadratic interpolation is fully quadratic) *Let $x \in \mathbb{R}^n$ and $f \in \mathcal{C}^{2+}$ on $B_{\bar{\Delta}}(x)$ with constant K. Let $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\} \subset \mathbb{R}^n$ be poised for quadratic interpolation with $y^0 = x$ and $\Delta = \Delta(\mathbb{Y}) \leq \bar{\Delta}$.*

For $0 < \delta \leq 1$ define $\mathbb{Y}_\delta = \{y^0, y^0 + \delta(y^1 - y^0), \ldots, y^0 + \delta(y^m - y^0)\}$. Then $\mathbb{Y}_\delta$ is poised for quadratic interpolation with $y^0 = x$ and $\Delta(\mathbb{Y}_\delta) = \delta\Delta < \bar{\Delta}$.

Let Q_δ be the quadratic interpolation function of f over $\mathbb{Y}_\delta$. Then $\{Q_\delta\}_{\delta \in (0,1]}$ defines a class of fully quadratic models at x parametrized by δ.

Proof. Details can be found in [27]. □

In Subsection 3.3 we saw one method for dealing with sample sets with greater than $n+1$ points. Comparing Theorem 3.10 to Theorem 3.14 makes it clear that if the number of sample points reaches $\frac{1}{2}(n+1)(n+2)$, then quadratic interpolation should become the preferred option. However, if $n+1 < m < \frac{1}{2}(n+1)(n+2)$ is closer to $\frac{1}{2}(n+1)(n+2)$ than $n+1$, then it still might be reasonable to try and squeeze some curvature information out of the data. To do this we turn to minimum Frobenius norm models.

Consider the situation where $\mathbb{Y} = \{y^0, y^1, \ldots, y^m\}$ and $n+1 < m+1 < (n+1)(n+2)/2$. There are not enough sample points to build a unique quadratic model of f. Indeed, there can be infinitely many quadratic functions parameterized by $\alpha_0, \alpha, H = H^\top$ that satisfy

$$\alpha_0 + \alpha^\top y^i + \frac{1}{2}(y^i)^\top H y^i = f(y^i) \quad \text{for } i = 0, 1, 2, \ldots m.$$

Among all these quadratic functions, we select the one whose quadratic coefficients of H are as small as possible in the Frobenius norm sense.[9] As such, we solve the following quadratic optimization problem

$$\begin{aligned} \min_{\alpha_0,\alpha,H} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} h_{i,j}^2 \\ & \alpha_0 + \alpha^\top y^i + \tfrac{1}{2}(y^i)^\top H y^i = f(y^i) \quad \text{for } i = 0, 1, 2, \dots m \\ & H = H^\top. \end{aligned} \tag{8}$$

Thus, we have the idea of minimum Frobenius norm models. The following definition extends the notion of a set of points being poised for Frobenius norm modelling.

Definition 3.15 (Poised for minimum Frobenius norm modelling) *The set* $\mathbb{Y} = \{y^0, y^1, \dots, y^m\} \subset \mathbb{R}^n$ *with* $n < m < \frac{1}{2}(n+1)(n+2) - 1$, *is* poised for minimum Frobenius norm modelling *if Problem* (8) *has a unique solution* (α_0, α, H).

Given a set of poised points, we next introduce the minimum Frobenius norm model.

Definition 3.16 (Minimum Frobenius norm model function) *Let* $f : \mathbb{R}^n \mapsto \mathbb{R}$, *and* $\mathbb{Y} = \{y^0, y^1, \dots, y^m\} \subset \mathbb{R}^n$, $n < m < \frac{1}{2}(n+1)(n+2) - 1$, *be poised for minimum Frobenius norm modelling. Then the* minimum Frobenius norm model function of f over $\mathbb{Y}$ *is*

$$M_Y(x) := \alpha_0 + \alpha^\top x + \frac{1}{2} x^\top H x$$

where (α_0, α, H) *is the unique solution to Problem* (8).

In these models, the number of points exceeds the required number of points for linear interpolation, but is not sufficient for quadratic interpolation. Therefore, the error bounds for minimum Frobenius norm models are not as strong as for quadratic interpolation, but at least as strong as those for linear interpolation models.

Theorem 3.17 (Minimum Frobenius norm modelling is fully linear) *Let* $x \in \mathbb{R}^n$ *and* $f \in \mathcal{C}^{1+}$ *on* $B_{\bar{\Delta}}(x)$ *with constant* K. *Let* $\mathbb{Y} = \{y^0, y^1, \dots, y^m\} \subset \mathbb{R}^n$ *be poised for minimum Frobenius norm modelling with* $y^0 = x$ *and* $\Delta = \Delta(\mathbb{Y}) \leq \bar{\Delta}$.

For $0 < \delta \leq 1$ *define* $\mathbb{Y}_\delta = \{y^0, y^0 + \delta(y^1 - y^0), \dots, y^0 + \delta(y^m - y^0)\}$. *Then* $\mathbb{Y}_\delta$ *is poised for minimum Frobenius norm modelling with* $y^0 = x$ *and* $\Delta(\mathbb{Y}_\delta) = \delta\Delta < \bar{\Delta}$.

Let M_δ *be the minimum Frobenius norm model function of* f *over* $\mathbb{Y}_\delta$. *Then* $\{M_\delta\}_{\delta \in (0,1]}$ *defines a class of fully linear models at* x *parametrized by* δ.

Proof. A proof of this result is found in [34]. □

On a final note, the particular case in which $m = n$ is such that $\mathbb{Y}$ forms a simplex. Quadratic interpolation with minimum Frobenius norm produces the same solution as linear interpolation because the optimal solution will set H to the zero matrix.

3.5 Using fully linear or quadratic models in a DFO algorithm

Now that we have established the ability to construct fully linear and fully quadratic models, let us examine how they can be used within derivative-free optimization algorithms. As mentioned, the value of the fully linear and fully quadratic terminology is that it decouples the algorithm from the model building process.

[9] This Frobenius norm approach was suggested by Conn, Scheinberg and Toint [25] in 1998, by Powell [74] for the UOBYQA algorithm, and by Custódio, Rocha and Vicente [34] for SID-PSM.

This is perhaps best illustrated via an example. Therefore, we present the model-based trust-region framework from [8, Chpt 11] (which is minor adaption of the framework in [29, Chpt 10]). As mentioned, many of the algorithms in Table 1 fit into this framework.

Algorithm 3.1 Model-based trust-region

Given $f \in \mathcal{C}^1$, starting point x^0 and fully linear models $\tilde{f}_\Delta$ of f

0. Initialize:

$\Delta^0 \in (0, \infty)$	initial trust-region radius
$\tilde{f}^0$	initial model (a fully linear model of f on $B_{\Delta^0}(x^0)$)
μ	model accuracy parameter
$\eta \in (0, 1)$	sufficient decrease test parameter
$\gamma \in (0, 1)$	trust-region update parameter
$\epsilon_{\texttt{stop}} \in [0, \infty)$	stopping tolerance
$k \leftarrow 0$	iteration counter

1. Model building and accuracy check:
   - if $\Delta^k > \mu \|\nabla \tilde{f}^k(x^k)\|$ or $\tilde{f}^k$ is not a fully linear model of f on $B_{\Delta^k}(x^k)$
     - decrease $\Delta^{k+1} = \gamma \Delta^k$, set $x^{k+1} = x^k$
     - use $\tilde{f}_\Delta$ to create a fully linear model $\tilde{f}^{k+1}$ of f on $B_{\Delta^{k+1}}(x^k)$
     - increment $k \leftarrow k + 1$ and go to 1
2. Stopping criterion:
   - if $\Delta^k < \epsilon_{\texttt{stop}}$ and $\|\nabla \tilde{f}^k(x^k)\| < \epsilon_{\texttt{stop}}$
     - declare algorithm success; stop
3. Trust-region subproblem:
   - solve (or approximate) $x^k_{\texttt{tmp}} \in \operatorname{argmin}_x \{\tilde{f}^k(x) \ : \ x \in B_{\Delta^k}(x^k)\}$
4. Candidate test and trust-region update:
   - evaluate $f(x^k_{\texttt{tmp}})$ and compute the ratio $\rho^k = \dfrac{f(x^k) - f(x^k_{\texttt{tmp}})}{\tilde{f}^k(x^k) - \tilde{f}^k(x^k_{\texttt{tmp}})}$
   - if $\rho^k > \eta$ (sufficient decrease exists), declare iterate success and
     - let x^{k+1} be any point such that $f(x^{k+1}) \leq f(x^k_{\texttt{tmp}})$
     - increase next trust-region $\Delta^{k+1} = \gamma^{-1} \Delta^k$
     - create $\tilde{f}^{k+1}$ using $\tilde{f}^k$ and x^{k+1}
   - otherwise $\rho^k \leq \eta$, declare iterate failure and
     - set $x^{k+1} = x^k$, decrease trust-region $\Delta^{k+1} = \gamma \Delta^k$ and keep $\tilde{f}^{k+1} = \tilde{f}^k$
   - increment $k \leftarrow k + 1$ and go to 1

In examining the model-based trust-region framework note that there is no mention of how the fully linear models are constructed. Convergence is established using the error bounds in Definition 3.5. This in done through two procedures imbedded within the algorithm.

In Step 1, the algorithm checks if $\Delta^k \leq \mu \|\nabla \tilde{f}^k(x^k)\|$. If this verification fails, then the trust-region radius Δ^k is decreased and a new model is constructed. As a result, the algorithm links the model accuracy with the first-order critical conditions: Higher accuracy is demanded as a potential critical point is approached.

In Step 2, the algorithm checks if $\Delta^k < \epsilon_{\texttt{stop}}$ and $\|\nabla \tilde{f}^k(x^k)\| < \epsilon_{\texttt{stop}}$. Thus, the algorithm should only stop when both model accuracy and the approximated gradient are sufficiently small. If both conditions are true, then applying the fully linear models ensure that

$$\|\nabla f(x^k)\| \leq \|\nabla f(x^k) - \nabla \tilde{f}^k(x^k)\| + \|\nabla \tilde{f}^k(x^k)\| \leq \kappa_g \Delta^k + \Delta^k \leq (\kappa_g + 1)\epsilon_{\texttt{stop}},$$

and an upper bound on the true gradient is obtained.

Using these two precautions, other smooth optimization algorithms can also be adapted to work with fully linear models. Gradient-based line-search methods [8, Ch 10] [29, Ch 9] [71], Newton's method [61], the proximal point method [51], and SQP [82] have been adapted to work for DFO. Of these, [8, Ch 10], [29, Ch 9], and [51] designed the algorithm to be independent of the modelling technique. As such, these algorithms are better termed frameworks, as changing the modelling technique may drastically alter the behaviour of the algorithm [46].

As future researchers explore model-based algorithms for DFO, we recommend using language that decouples the algorithm and the model building technique.

4 Accuracy and models of nonsmooth functions

4.1 Accuracy of Nonsmooth Functions

While Definitions 3.6 and 3.8 (fully linear and fully quadratic) provide a useful terminology discussing models of smooth functions, they depend on the true gradients and true Hessians to provide the marker for comparison. As such, if the objective function f is nonsmooth, then it is impossible to construct a class of fully linear approximations. We therefore introduce new terminology that allows us to appraise the quality of a model. We will begin with *order N function accuracy.*

Definition 4.1 (Order N function accuracy) *Given f, $x \in \mathrm{dom}(f)$, and $\bar{\Delta} > 0$ we say that $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of models of f at x parameterized by Δ that* provides order N function accuracy at x *if there exist a scalar $\kappa_f(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

$$|f(x) - \tilde{f}_\Delta(x)| \leq \kappa_f(x)\Delta^N.$$

We say that $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of models of f at x parameterized by Δ that provides order N function accuracy near x *if there exist a scalar $\kappa_f(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

$$|f(y) - \tilde{f}_\Delta(y)| \leq \kappa_f(x)\Delta^N \qquad \text{for all } \ y \in B_\Delta(x).$$

Clearly, if $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of fully linear models, then it is also a class of models that provides order 2 function accuracy near x. Similarly, if $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of fully quadratic models, then it is also a class of models that provides order 3 function accuracy near x.

Let us now introduce the definition of subgradient accuracy of a Lipschitz function. The definition relies on the notion of the Clarke subdifferential introduced in Definition 2.1. The elements of the subdifferential are called subgradients.

Definition 4.2 (Order N subgradient accuracy) *Given $f \in \mathcal{C}^{0+}$, $x \in \mathbb{R}^n$, and $\bar{\Delta} > 0$ we say that $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of models of f at x parameterized by Δ that* provides order N subgradient accuracy at x *if there exist a scalar $\kappa_g(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

i) given any $v \in \partial f(x)$ there exists $\tilde{v} \in \partial\tilde{f}_\Delta(x)$ with $\|v - \tilde{v}\| \leq \kappa_g(x)\Delta^N$, and
ii) given any $\tilde{v} \in \partial\tilde{f}_\Delta(x)$ there exists $v \in \partial f(x)$ with $\|v - \tilde{v}\| \leq \kappa_g(x)\Delta^N$.

We say that $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of models of f at x parameterized by Δ that provides order N subgradient accuracy near x *if there exist a scalar $\kappa_g(x) > 0$ such that, given any $\Delta \in (0, \bar{\Delta}]$ the model $\tilde{f}_\Delta$ satisfies*

i) for all $y \in B_\Delta(x)$ given any $v \in \partial f(y)$ there exists $\tilde{v} \in \partial\tilde{f}_\Delta(y)$ with $\|v - \tilde{v}\| \leq \kappa_g(x)\Delta^N$, and
ii) for all $y \in B_\Delta(x)$ given any $\tilde{v} \in \partial\tilde{f}_\Delta(y)$ there exists $v \in \partial f(y)$ with $\|v - \tilde{v}\| \leq \kappa_g(x)\Delta^N$.

If $f \in \mathcal{C}^{1+}$ and $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of fully linear models, then it is also a class of models that provides order 1 subgradient accuracy near x. Similarly, if $f \in \mathcal{C}^{1+}$ and $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of fully quadratic models, then it is also a class of models that provides order 2 subgradient accuracy near x.

Surprisingly, if $f \in \mathcal{C}^{1+}$ and $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$, $\tilde{f}_\Delta \in \mathcal{C}^{1+}$, is a class of models that provides order 1 subgradient accuracy and order 2 function accuracy *at* x, then $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ is a class of fully linear models.

Theorem 4.3 (Fully linear versus order N accuracy) *Suppose $x \in \mathbb{R}^n$, $\bar{\Delta} > 0$, $f \in \mathcal{C}^{1+}$ with constant K on $B_{\bar{\Delta}}(x)$. Suppose that the model functions $\tilde{f}_\Delta \in \mathcal{C}^{1+}$ with constant $\tilde{K}$ on $B_{\bar{\Delta}}(x)$. Then the following are equivalent*

i) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *is a class of fully linear models,*

ii) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *provides order* 1 *subgradient accuracy and order* 2 *function accuracy near* x*,*

iii) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *provides order* 1 *subgradient accuracy and order* 2 *function accuracy at* x*.*

Proof. Clearly (i) and (ii) are equivalent and imply (iii).

Suppose (iii) is true. Note that $f \in \mathcal{C}^{1+}$ and $\tilde{f}_\Delta \in \mathcal{C}^{1+}$ imply that $\partial f = \{\nabla f\}$ and $\partial \tilde{f}_\Delta = \{\nabla \tilde{f}_\Delta\}$. Thus, order 1 subgradient accuracy at x is equivalent to

$$\|\nabla f(x) - \nabla \tilde{f}_\Delta(x)\| \leq \kappa_g \Delta,$$

where $\kappa_g = \kappa_g(x)$ is the parameter of subgradient accuracy. Considering any $y \in B_{\bar{\Delta}}(x)$, we find that

$$\begin{array}{rcl} \|\nabla f(y) - \nabla \tilde{f}_\Delta(y)\| & \leq & \|\nabla f(y) - \nabla f(x)\| + \|\nabla f(x) - \nabla \tilde{f}_\Delta(x)\| + \|\nabla \tilde{f}_\Delta(x) - \nabla \tilde{f}_\Delta(y)\| \\ & \leq & K\|y - x\| + \kappa_g \Delta + \tilde{K}\|y - x\| \\ & \leq & (\kappa_g + K + \tilde{K})\Delta. \end{array}$$

Thus, the models provide order 1 subgradient accuracy *near* x using parameter $\kappa_g^* = \kappa_g + K + \tilde{K}$.

Next define $\hat{f}_\Delta = \tilde{f}_\Delta - \tilde{f}_\Delta(x) + f(x)$. By Proposition 3.11 $\{\hat{f}_\Delta\}$ is a class of fully linear models of f at x, so (ii) holds for $\hat{f}_\Delta$. Let κ_f be the parameter of function accuracy $\tilde{f}_\Delta$ *at* x and $\hat{\kappa}_f$ be the parameter of function accuracy $\hat{f}_\Delta$ *near* x. Considering any $y \in B_{\bar{\Delta}}(x)$, we find that

$$\begin{array}{rcl} |f(y) - \tilde{f}_\Delta(y)| & \leq & |f(y) - \hat{f}_\Delta(y)| + |\hat{f}_\Delta(y) - \tilde{f}_\Delta(y)| \\ & \leq & \hat{\kappa}_f \Delta^2 + |\tilde{f}_\Delta(y) - \tilde{f}_\Delta(x) + f(x) - \tilde{f}_\Delta(y)| \\ & = & \hat{\kappa}_f \Delta^2 + |\tilde{f}_\Delta(x) - f(x)| \\ & \leq & \hat{\kappa}_f \Delta^2 + \kappa_f \Delta^2. \end{array}$$

Thus, the models provide order 2 function accuracy *near* x using parameter $\kappa_f^* = \hat{\kappa}_f + \kappa_f$. □

The notion of order N subgradient accuracy can be extended to higher (generalized) derivatives in the obvious manner. This suggests the following *conjecture* (since we leave this for future researchers, we do not formally define the notion of order N Hessian accuracy).

Conjecture 4.4 (Fully quadratic versus N order accuracy) *Suppose* $x \in \mathbb{R}^n$*,* $\bar{\Delta} > 0$*,* $f \in \mathcal{C}^{2+}$ *with constant* K *on* $B_{\bar{\Delta}}(x)$*. Suppose that the model functions* $\tilde{f}_\Delta \in \mathcal{C}^{2+}$ *with constant* $\tilde{K}$ *on* $B_{\bar{\Delta}}(x)$*. Then the following are equivalent*

i) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *is a class of fully quadratic models,*

ii) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *provides order* 1 *Hessian accuracy, order* 2 *gradient accuracy, and order* 3 *function accuracy near* x*,*

iii) $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ *provides order* 1 *Hessian accuracy, order* 2 *gradient accuracy, and order* 3 *function accuracy at* x*.*

In light of Theorem 4.3, it would appear that the minimum standard for a model-based DFO method to be expected to work might be that the class of models $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ provides order 1 subgradient accuracy and order 2 function accuracy near x^k. In fact, if the only target is first-order optimality, then it suffices to have access to a class of models that provides order 1 subgradient accuracy at x^k.

Algorithm 4.1 Model-based steepest-descent

Given $f \in \mathcal{C}^1$, starting point $x^0 \in \mathbb{R}^n$ and a class of models $\{\tilde{f}_\Delta\}_{\Delta\in(0,\bar{\Delta}]}$ that provides order 1 subgradient accuracy at given points

0. Initialize:
   - $\Delta^0 \in (0,\infty)$ initial model accuracy parameter
   - $\mu^0 \in (0,\infty)$ initial target accuracy parameter
   - $\eta \in (0,1)$ an Armijo parameter
   - $\varepsilon_d \in (0,1)$ minimum decrease angle parameter
   - $\epsilon_{\texttt{stop}} \in [0,\infty)$ stopping tolerance
   - $k \leftarrow 0$ iteration counter
1. Model and descent direction:
   access the class of models to select $\tilde{f}_\Delta^k$ that provides order 1 subgradient accuracy at x^k
   select $\tilde{g}^k \in \partial\tilde{f}_\Delta^k(x^k)$ to (approximately) solve $\operatorname{argmin}\{\|g\| : g \in \partial\tilde{f}_\Delta^k(x^k)\}$
2. Model accuracy checks:
   a) if $\Delta^k < \epsilon_{\texttt{stop}}$ and $\|\tilde{g}^k\| < \epsilon_{\texttt{stop}}$
      declare algorithm success and stop
   b) if $\Delta^k > \mu^k\|\tilde{g}^k\|$
      declare the model inaccurate
      decrease $\Delta^{k+1} \le \frac{1}{2}\Delta^k$, set $\mu^{k+1} = \mu^k, x^{k+1} = x^k$, $k \leftarrow k+1$, go to 1
   c) if $\Delta^k \le \mu^k\|\tilde{g}^k\|$
      declare the model accurate and proceed to 3
3. Line search:
   set $d^k = -\frac{\tilde{g}^k}{\|\tilde{g}^k\|}$ (descent)
   perform a line-search in the direction d^k to seek t^k with
   $f(x^k + t^k d^k) < f(x^k) + \eta t^k (d^k)^\top \tilde{g}^k$ (Armijo)
   if t^k is found declare line-search success
   otherwise declare line-search failure
4. Update:
   if line-search success
      let x^{k+1} be any point such that $f(x^{k+1}) \le f(x^k + t^k d^k)$
      set $\Delta^{k+1} = \Delta^k, \mu^{k+1} = \mu^k$
   otherwise (line-search failure)
      set $x^{k+1} = x^k, \Delta^{k+1} = \Delta^k$, and decrease $\mu^{k+1} \le \frac{1}{2}\mu^k$
   increment $k \leftarrow k+1$ and go to 1

As the name suggests, the Model-based steepest-descent framework is essentially the nonsmooth steepest-descent algorithm with the true subdifferential replaced with a model-based approximate subdifferential. Convergence of the Model-based steepest-descent framework can be established using fairly standard DFO techniques. To begin, consider the stopping test (Step 2a).

Lemma 4.5 *Suppose the stopping test in Step (2a) is triggered. Then x^k satisfies*

$$dist(0, \partial f(x^k)) \le (1+\kappa_g)\epsilon_{\texttt{stop}},$$

where κ_g is the constant for subgradient accuracy.

Conversely, if $dist(0,\partial f(x^k)) < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\}$ and $\Delta^k < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\}$, then the algorithm will stop.

Proof. First suppose the stopping test is triggered. Order 1 subgradient accuracy implies that for any $\tilde{g}^k \in \partial\tilde{f}_{\Delta^k}$ there exists $g^k \in \partial f(x^k)$ with $\|\tilde{g}^k - g^k\| \le \kappa_g\Delta^k$. So, if $\Delta^k < \epsilon_{\texttt{stop}}$ and $\|\tilde{g}^k\| < \epsilon_{\texttt{stop}}$, using this g^k, we note that

$$\begin{aligned} \operatorname{dist}(0,\partial f(x^k)) &\le \|0 - g^k\| \\ &\le \|0 - \tilde{g}^k\| + \|\tilde{g}^k - g^k\| \\ &\le \epsilon_{\texttt{stop}} + \kappa_g\Delta^k \le (1+\kappa_g)\epsilon_{\texttt{stop}}. \end{aligned}$$

Now suppose $\text{dist}(0, \partial f(x^k)) < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\}$ and $\Delta^k < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\}$. Let $g^k \in \partial f(x^k)$ with $\|g^k\| < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\}$. Applying order 1 subgradient accuracy, we know that there exists $\tilde{g}^k \in \partial \tilde{f}_{\Delta^k}$ with $\|\tilde{g}^k - g^k\| < \kappa_g \Delta^k$. Which yields

$$\|\tilde{g}^k\| < \kappa_g \Delta^k + \|g^k\| < \kappa_g \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\} + \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\} \leq \epsilon_{\texttt{stop}}.$$

Since $\Delta^k < \frac{1}{2}\min\{\epsilon_{\texttt{stop}}/\kappa_g, \epsilon_{\texttt{stop}}\} \leq \epsilon_{\texttt{stop}}$, the stopping conditions are triggered. □

Lemma 4.5 demonstrates the importance of the two parts of defining order 1 subgradient accuracy. Part ii of Definition 4.2 is used to show that if the algorithm stops, then an approximate critical point is found. Part i of Definition 4.2 is used to show that if the algorithm reaches an approximate critical point, then the stopping conditions will be triggered (once Δ^k is sufficiently small).

Next, we consider the situation where Algorithm 4.1 is run without stopping conditions (i.e., Step 2a is omitted).

Theorem 4.6 *Suppose $f \in \mathcal{C}^{0+}$ with constant K and bounded below. Suppose Algorithm 4.1 is run without stopping conditions and let x^k be the resulting infinite sequence. Suppose the step sizes t^k are bounded below by a constant $\bar{t} > 0$. Then either*

i) $f(x^k) \searrow -\infty$, or
ii) $\|d^k\| \to 0$ and every cluster point $\hat{x}$ of $\{x^k\}$ satisfies $0 \in \partial f(\hat{x})$.

Proof. A comparison of Algorithm 4.1 to [48, Alg. AGS] shows that Algorithm 4.1 generalizes [48, Alg. RAGS] to include any modelling technique that provides order 1 subgradient accuracy (whereas [48, Alg. RAGS] employed specific modelling technique in the algorithm). All the proofs in [48] except [48, Lem 1] use only order 1 subgradient accuracy assumption, while [48, Lem 1] proves the applied modelling technique provides order 1 subgradient accuracy. As such, this result is an easy adaptation of [48]. □

In presenting the Model-based steepest-descent framework, we note that, like steepest descent, it is unlikely to be competitive with a more advanced algorithm.[10] Instead, our goal in presenting Algorithm 4.1 is to demonstrate how a model-based DFO algorithm can be decoupled from the models. Any modelling technique that satisfies order 1 subgradient accuracy at x^k can be used within the Model-based steepest-descent framework to create a convergent method.

Let us return now to Table 2. As we have just demonstrated, the RAGS algorithm in [48] is easily adapted to allow for any model that provides order 1 subgradient accuracy at x^k. As the RAGS algorithm is an advancement to the WASG algorithm in [47], it is reasonable to say that WASG is also adaptable to any model that provides order 1 subgradient accuracy at x^k. Considering the rest of Table 2:

- The DGM algorithm of [11] applies a model that converges in a limiting sense only;[11]
- The $\text{DFO}_{\text{comirror}}$ of [12] applies a model that provides order 1 subgradient accuracy and order 2 function accuracy at x^k [12, Cor 3.3] [45, Thm 2.1 & 3.1];
- The CMS, GDMS, DMS, SMS algorithms of [59] apply models that provides order 1 subgradient accuracy and order 2 function accuracy at x^k [59, Lem 2] [45, Thm 2.1 & 3.1];
- The MS_4PL of [56] applies a model that to provides order 1 subgradient accuracy ([56, Lem 1] shows the first required inequality, the second required inequality can be obtained using similar techniques);
- The DFO_VU algorithm of [49] applies a model that provides order 1 subgradient accuracy and order 2 function accuracy at x^k [45, Thm 2.1 & 3.1].

[10] This said, in model-based DFO the relative effectiveness of algorithms is often less clear. For smooth optimization, Newton's method will almost always dramatically outperform steepest descent. In model-based DFO, the inaccuracies in the models and the difficulty in constructing models with good quality Hessian approximations may make steepest descent a competitive alternative [46].

[11] In [11], the model accuracy is forced to converge to 0 by pre-selecting the model accuacy level δ_k such that $\delta_k \to 0$.

In each case, it is reasonable to conjecture that there is nothing special about the models used, except the types and quality of accuracy they provide. It would be valuable to reexamine the methods above and see if a model-flexible method of each method could be created.

4.2 Constructing models of nonsmooth functions

Constructing models that provide order N function accuracy at x can be accomplished easily. Indeed, one only needs to ensure that the model value is correct at the point of interest. Constructing models that provide order N subgradient accuracy at x is more difficult.

As mentioned in Section 2, some researchers have explored constructing models that provide some subgradient accuracy for broad classes of functions [11, 57]. While theoretically interesting, the number of function calls required to create the models is too high for practical use.[12] As such, researchers have turned to studying constructing models for more structured functions [47, 48, 59, 45, 56].

Let us consider the situation where the objective function f is the composition of two functions $f(x) = g(F(x))$. Suppose that function values (for f) are computed in two steps: first the vector-valued function $F(x) = [F_1(x), F_2(x), \ldots, F_m(x)]$ is evaluated through a blackbox, second the single-valued analytically available function g is evaluated at $F(x)$. Using this framework, if we can construction model functions for each F_i, then the analytic structure of g allows an obvious method to construct a model function of f. In particular, if $\tilde{F}_{i,\Delta}$ are model functions of F_i, then we can use $\tilde{f}_\Delta(x) = g([\tilde{F}_{i,\Delta}(x), \tilde{F}_{i,\Delta}(x), \ldots, \tilde{F}_{i,\Delta}(x)])$ as our model function of f.

We next define model accuracy for vector valued functions.

Definition 4.7 (Order N accuracy for vector-valued functions) *Given*

$$F : \mathbb{R}^n \to \mathbb{R}^m \ : \ x \mapsto [F_1(x), F_2(x), ..., F_m(x)],$$

$x \in \text{dom}(F)$, *and* $\bar{\Delta} > 0$, *we say that* $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *is a class of models of* F *parameterized by* Δ *that* provides order N function (subgradient) accuracy *at* x *(near* x*) if*

$$\tilde{F}_\Delta : \mathbb{R}^n \to \mathbb{R}^m \ : \ x \mapsto [\tilde{F}_{1,\Delta}(x), \tilde{F}_{2,\Delta}(x), ...\tilde{F}_{m,\Delta}(x)],$$

and each $\tilde{F}_{i,\Delta}(x)$ *is a class of models of* F_i *that provides order* N *function (subgradient) accuracy at* x *(near* x*).*

In this case, we use $\kappa_{F_i}(x)$ *(*$\kappa_{G_i}(x)$*) as the accuracy parameter for subfunction* F_i, *and* $\kappa_F(x) = \max\{\kappa_{F_i}(x)\}$ *(*$\kappa_G(x) = \max\{\kappa_{G_i}(x)\}$*) in order to provide a single accuracy parameter suitable for all subfunctions.*

The next result provides bounds on the function accuracy of a model constructed by composing g with models of the vector-valued function F. A similar result can be found in [45, Thm 2.1].

Theorem 4.8 (Compositions of order N function accuracy) *Suppose*

$$f : \mathbb{R}^n \to \mathbb{R} \ : \ x \mapsto g(F(x)),$$

where $g : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\}$ *and* $F : \mathbb{R}^n \to \mathbb{R}^m$. *Suppose* $x \in \mathbb{R}^n$ *with* $F(x) \in \text{int}(\text{dom}(g))$ *and* g *is locally Lipschitz on its domain. Let* $\bar{\Delta} > 0$ *and* $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *be a class of models of* F *parameterized by* Δ.

Define

$$\tilde{f}_\Delta = g \circ \tilde{F}_\Delta.$$

i) If $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *provides order* N, $N \geq 1$, *function accuracy at* x, *then there exists* $\bar{\Delta}_f > 0$ *such that* $\{\tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}_f]\}$ *provides order* N *function accuracy at* x.

[12] The techniques of [11, 57] do not provide error bounds, but instead showed asymptotic convergence. Thus, independent of the number of function calls, the models do not satisfy order N accuracy conditions.

ii) If $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *provides order* N*,* $N \geq 1$*, function accuracy near* x*, then there exists* $\bar{\Delta}_f > 0$ *such that* $\{\tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}_f]\}$ *provides order* N *function accuracy near* x*.*

Proof. (i) Suppose $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ provides order N function accuracy at x. Let L be a local Lipschitz constant of g at x and $\kappa_F(x)$ be the parameter of order N function accuracy for F. Since $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ provides order N function accuracy at x, there exists $\bar{\Delta}_f \in (0, \bar{\Delta}]$ such that $\tilde{F}_\Delta(x) \in \text{int}(\text{dom}(g))$ for all $\Delta \in (0, \bar{\Delta}_f]$.

Given any $\delta \in (0, \bar{\Delta}_f]$ we find that

$$\begin{aligned} |f(x) - \tilde{f}_\Delta(x)|^2 &= |g(F(x)) - g(\tilde{F}_\Delta(x))|^2 \\ &\leq L^2 \|F(x) - \tilde{F}_\Delta(x)\|^2 \\ &= L^2 \textstyle\sum_{i=1}^m |F_i(x) - \tilde{F}_{i,\Delta}(x)|^2 \\ &\leq L^2 \textstyle\sum_{i=1}^m (\kappa_F(x))^2 \Delta^{2N} \\ &\leq L^2 m (\kappa_F(x))^2 \Delta^{2N}. \end{aligned}$$

Which shows that $|f(x) - \tilde{f}_\Delta(x)| \leq L\sqrt{m}\kappa_F(x)\Delta^N$, so $\{\tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}_f]\}$ provides order N function accuracy at x.

(ii) Suppose $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ provides order N function accuracy near x. As $F(x) \in \text{int}(\text{dom}(g))$ and F is continuous, there exists $\delta > 0$ such that $F(y) \in \text{int}(\text{dom}(g))$ for all $y \in \text{cl}(B_\delta(x))$. Since $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta})\}$ provides order N function accuracy at x, this implies the existence of $\bar{\Delta}_f \in (0, \min\{\delta, \bar{\Delta}\}]$ such that $\tilde{F}_\Delta(y) \in \text{int}(\text{dom}(g))$ for all $\Delta \in (0, \bar{\Delta}_f]$, $y \in \text{cl}(B_{\bar{\Delta}_f}(x))$.

By the continuity of F, $\{z \in \mathbb{R}^m : z = F(y), y \in \text{cl}(B_{\bar{\Delta}_f}(x))\} \subseteq \text{int}(\text{dom}(g))$ is compact. As such, there exists a Lipschitz constant for g relative to this set [81, Thm 9.2]. Let L be this Lipschitz constant and $\kappa_F(x)$ be the parameter of order N function accuracy for F.

Given any $\Delta \in (0, \bar{\Delta}_f]$ and $y \in B_\Delta(\bar{x})$, following an analogous series of inequalities to the above, we find that

$$|f(y) - \tilde{f}_\Delta(y)|^2 \leq L^2 m (\kappa_F(x))^2 \Delta^{2N}.$$

Which shows that $\{\tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}_f]\}$ provides order N function accuracy near x. □

It is not difficult to grasp the importance of $F(x) \in \text{int}(\text{dom}(g))$ in Theorem 4.8, as without it we can select models that result in $g(\tilde{F}_\Delta(x)) = \infty$ [45, Ex 2.3].

We next examine subgradient accuracy of a model constructed by composing g with models of the vector-valued function F. Theorem 4.9 shows that given three key pieces the composition models retain subgradient accuracy properties. These key pieces are: the chain rule $\partial f(x) = \nabla F(x)^\top \partial g(F(x))$ holds, the subdifferential is bounded, and the models give exact function values at the point of interest.

Theorem 4.9 (Compositions of order N subgradient accuracy) *Suppose*

$$f : \mathbb{R}^n \to \mathbb{R} \ : \ x \mapsto g(F(x)),$$

where $g : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\}$ *and* $F : \mathbb{R}^n \to \mathbb{R}^m$*. Suppose a nondegeneracy condition holds for* g *and* F *so that*

$$\partial f(x) = \nabla F(x)^\top \partial g(F(x))$$

and $\partial g(F(x))$ *is bounded.*

Let $\bar{\Delta} > 0$ *and* $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *be a class of models of* F *parameterized by* Δ *satisfying* $F(x) = \tilde{F}_\Delta(x)$*. Define*

$$\tilde{f}_\Delta = g \circ \tilde{F}_\Delta.$$

and suppose a nondegeneracy condition holds for g *and* $\tilde{F}_\Delta$ *so that*

$$\partial \tilde{f}_\Delta(x) = \nabla \tilde{F}_\Delta(x)^\top \partial g(\tilde{F}_\Delta(x)).$$

If $\{\tilde{F}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *provides order* N, $N \geq 1$, *subgradient accuracy at* x, *then* $\{\tilde{f}_\Delta : \Delta \in (0, \bar{\Delta}]\}$ *provides order* N *subgradient accuracy at* x.

Proof. Apply $F(x) = \tilde{F}_\Delta(x)$ and the chain rule to see that

$$\partial \tilde{f}_\Delta(x) = \nabla \tilde{F}_\Delta(x)^\top \partial g(F(x)).$$

Note $\partial g(F(x))$ is bounded and define $M = \sup\{\|w\| : w \in \partial g(F(x))\}$. Let $\kappa_G(x)$ be the parameter of order N subgradient accuracy for F.

Selecting any $v \in \partial f(x)$, there exist $w \in \partial g(F(x))$ such that $v = \nabla F(x)^\top w$. Define $\tilde{v} = \nabla \tilde{F}_\Delta(x)^\top w \in \partial \tilde{f}_\Delta(x)$, and notice that

$$\begin{array}{rcl}
\|v - \tilde{v}\|^2 & = & \|\nabla F(x)^\top w - \nabla \tilde{F}_\Delta(x)^\top w\|^2 \\
& = & \left\| \begin{bmatrix} \nabla F_1(x)^\top w - \nabla \tilde{F}_{1,\Delta}(x)^\top w \\ \nabla F_2(x)^\top w - \nabla \tilde{F}_{2,\Delta}(x)^\top w \\ \vdots \\ \nabla F_m(x)^\top w - \nabla \tilde{F}_{m,\Delta}(x)^\top w \end{bmatrix} \right\|^2 \\
& = & \sum_{i=1}^m (F_m(x)^\top w - \nabla \tilde{F}_{m,\Delta}(x)^\top w)^2 \\
& \leq & \sum_{i=1}^m \|F_m(x) - \nabla \tilde{F}_{m,\Delta}(x)\|^2 \|w\|^2 \\
& \leq & \sum_{i=1}^m (\kappa_G(x))^2 \Delta^{2N} M^2 \leq m(\kappa_G(x))^2 M^2 \Delta^{2N}.
\end{array}$$

This yields $\|v - \tilde{v}\| \leq \sqrt{m}\kappa_G(x) M \Delta^N$, which gives the first inequality required for order N subgradient accuracy.

Conversely, selecting any $\tilde{v} \in \partial \tilde{f}_\Delta(x)$, there exist $w \in \partial g(F(x))$ such that $v = \nabla \tilde{F}_\Delta(x)^\top w$. Define $v = \nabla F(x)^\top w \in \partial f(x)$. Repeating the sequence of inequalities above, yields the second inequality required for order N subgradient accuracy. □

The important of the chain rule in Theorem 4.9 is obvious. Examples of what can go wrong if the subdifferential is not bounded or the models do not give exact function values at the point of interest can be found in [45, Ex 3.2 & 3.3].

5 Conclusions and open directions

DFO studies algorithm that do not use derivative information, because explicit derivative information is unavailable. This might be because the derivatives are not explicitly known, or simply because they do not exist. Model-based methods build approximations and use them in an algorithm that explores the space of variables. The ways that the models are constructed and the way that the algorithm exploits them define various model-based methods.

Many model-based methods target unconstrained optimization problems, and assume that the objective function is once or twice differentiable. These assumptions motivate the use of models, and allows the methods to be supported by a convergence analysis relating the accuracy of the model, its gradient, or possibly its Hessian. Models for such problems include interpolation, (generalized) simplex-gradient models, and other fully linear and fully quadratic models.

More recently, model-based DFO methods have targeted problems for which there is no reason to believe that the functions are differentiable. Different tools need to be used to construct models, and different analytical nonsmooth tools need to be applied to study their behaviour. We have presented some of this research.

We believe that research in model-based DFO for nonsmooth optimization is only at its beginning. Open research directions in model-based DFO for nonsmooth optimization include the following.

i) Expanding the list of algorithms, and identifying which work best for specific classes of problems. We believe that an important strategy when developing new algorithms, it to present it in a flexible way so that it is generic on which model is used. To this end we introduce the term "order N accuracy" (Definitions 4.1 and 4.2) to describe model quality without providing details on model construction.

ii) Expanding modelling techniques, in particular for nonsmooth functions. Herein we present some results for when the objective function is the composition of a nonsmooth and a smooth function. However, other possible nonsmooth structures should be considered. For example, Poissant [70] recently studied a situation in which only incomplete monotonic information is known. Another obvious structure is splitting problems $f = F + G$, where F is smooth and G is some form of nonsmooth regularizer (e.g., as found in the LASSO problem [87]). While this could be rephrased as a composition of a nonsmooth and a smooth function, it may be valuable to consider such popular structures directly.

iii) Determine how the knowledge of structure within a problem (as mentioned in ii) can best be exploited by an optimization method. A step towards this might be achieved by a better understanding of calculus rules for models, similar to the rules suggested for smooth functions in [78].

iv) Determine how model-based DFO methods can be applied to problems with nonlinear constraints. There has been some progress in the development of direct search algorithms for inequality constrained blackbox optimization [7], but model-based DFO method have largely remained in the realm of unconstrained or box-constrained problems. One step in model-based optimization has been made by Amaioua et al. [3]. Another step include be the development of a definition for "order N accuracy" of the normal cone (a starting place might be [50, 37]). Such a definition would allow for modelling techniques and model-based DFO algorithms for constrained optimization to be researched independently.

v) Further research on interweaving model-based and direct search DFO methods. More research should be conducted in combining such methods and in comparing strengths and weaknesses of each.

References

[1] S. Alarie, C. Audet, V. Garnier, S. Le Digabel, and L.-A. Leclaire. Snow water equivalent estimation using blackbox optimization. Pacific Journal of Optimization, 9(1):1–21, 2013.

[2] P. Alberto, F. Nogueira, H. Rocha, and L.N. Vicente. Pattern search methods for user-provided points: Application to molecular geometry problems. SIAM Journal on Optimization, 14(4):1216–1236, 2004.

[3] N. Amaioua, C. Audet, A.R. Conn, and S. Le Digabel. Efficient solution of quadratically constrained quadratic subproblems within the mesh adaptive direct search algorithm. European Journal of Operational Research, 268(1):13–24, 2018.

[4] C. Audet. A survey on direct search methods for blackbox optimization and their applications. In P.M. Pardalos and T.M. Rassias, editors, Mathematics without boundaries: Surveys in interdisciplinary research, chapter 2, pages 31–56. Springer, 2014.

[5] C. Audet, A.R. Conn, S. Le Digabel, and M. Peyrega. A progressive barrier derivative-free trust-region algorithm for constrained optimization. Technical Report G-2016-49, Les cahiers du GERAD, 2016.

[6] C. Audet and J.E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. SIAM Journal on Optimization, 17(1):188–217, 2006.

[7] C. Audet and J.E. Dennis, Jr. A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization, 20(1):445–472, 2009.

[8] C. Audet and W. Hare. Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, 2017.

[9] C. Audet and W. Hare. Algorithmic construction of the subdifferential from directional derivatives. Set Valued and Variational Analysis, to appear, 2018.

[10] C. Audet, A. Ianni, S. Le Digabel, and C. Tribes. Reducing the Number of Function Evaluations in Mesh Adaptive Direct Search Algorithms. SIAM Journal on Optimization, 24(2):621–642, 2014.

[11] A.M. Bagirov, B. Karasözen, and M. Sezer. Discrete gradient method: Derivative-free method for nonsmooth optimization. Journal of Optimization Theory and Applications, 137(2):317–334, May 2008.

[12] H. Bauschke, W. Hare, and W. Moursi. A derivative-free comirror algorithm for convex optimization. Optim. Method. Softw., 30(4):706–726, 2015.

[13] V. Beiranvand, W. Hare, and Y. Lucet. Benchmarking of single-objective optimization algorithms. Eng. Optim., to appear, 2018.

[14] F.V. Berghen. CONDOR: A Constrained, Non-Linear, Derivative-Free Parallel Optimizer for Continuous, High Computing Load, Noisy Objective Functions. PhD thesis, Université Libre de Bruxelles, Belgium, 2004.

[15] F.V. Berghen and H. Bersini. CONDOR, a new parallel, constrained extension of Powell's UOBYQA algorithm: Experimental results and comparison with the DFO algorithm. jcomam, 181:157–175, 2005.

[16] K. Bigdeli, W. Hare, J. Nutini, and S. Tesfamariam. Optimizing damper connectors for adjacent buildings. Optimization and Engineering, 17(1):47–75, 2016.

[17] S.C. Billups, J. Larson, and P. Graf. Derivative-free optimization of expensive functions with computational error using weighted regression. SIAM Journal on Optimization, 23(1):27–53, 2013.

[18] A.J. Booker, J.E. Dennis, Jr., P.D. Frank, D.W. Moore, and D.B. Serafini. Managing surrogate objectives to optimize a helicopter rotor design – further experiments. AIAA Paper 1998–4717, Presented at the 8th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, 1998.

[19] A.J. Booker, J.E. Dennis, Jr., P.D. Frank, D.B. Serafini, and V. Torczon. Optimization using surrogate objectives on a helicopter test example. In J. Borggaard, J. Burns, E. Cliff, and S. Schreck, editors, Optimal Design and Control, Progress in Systems and Control Theory, pages 49–58, Cambridge, Massachusetts, 1998. Birkhäuser.

[20] A.J. Booker, J.E. Dennis, Jr., P.D. Frank, D.B. Serafini, V. Torczon, and M.W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. Structural and Multidisciplinary Optimization, 17(1):1–13, 1999.

[21] D.M. Bortz and C.T. Kelley. The simplex gradient and noisy optimization problems. In Jeff Borggaard, John Burns, Eugene Cliff, and Scott Schreck, editors, Optimal Design and Control, Progress in Systems and Control Theory, pages 77–90, Cambridge, Massachusetts, 1998. Birkhäuser.

[22] J.V. Burke, A.S. Lewis, and M.L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM J. Optim., 15(3):751–779, 2005.

[23] Á. Bűrmen, J. Olenšek, and T. Tuma. Mesh Adaptive Direct Search with Second Directional Derivative-based Hessian Update. Computational Optimization and Applications, 62(3):693–715, December 2015.

[24] A.R. Conn and S. Le Digabel. Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. Optimization Methods and Software, 28(1):139–158, 2013.

[25] A.R. Conn, K. Scheinberg, and Ph.L. Toint. A derivative free optimization algorithm in practice. In Proceedings the of 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, 1998.

[26] A.R. Conn, K. Scheinberg, and Ph.L. Toint. DFO (Derivative Free Optimization). https://projects.coin-or.org/Dfo, 2001.

[27] A.R. Conn, K. Scheinberg, and L.N. Vicente. Geometry of interpolation sets in derivative free optimization. Mathematical Programming, 111(1–2):141–172, 2008.

[28] A.R Conn, K. Scheinberg, and L.N. Vicente. Global convergence of general derivative-free trust-region algorithms to first and second order critical points. SIAM Journal on Optimization, 20(1):387–415, 2009.

[29] A.R. Conn, K. Scheinberg, and L.N. Vicente. Introduction to Derivative-Free Optimization. MOS-SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[30] A.R. Conn and Ph.L. Toint. An algorithm using quadratic interpolation for unconstrained derivative free optimization. In G. Di Pillo and F. Gianessi, editors, Nonlinear Optimization and Applications, pages 27–47. Plenum Publishing, New York, 1996.

[31] I. Coope and R. Tappenden. Efficient calculation of regular simplex gradients. Technical Report https://arxiv.org/abs/1710.01427v1, Department of Mathematics and Statistics, University of Canterbury, 2018.

[32] A.L. Custódio. Aplicações de Derivadas Simplécticas em Métodos de Procura Directa. PhD thesis, Universidade Nova de Lisboa, Portugal, 2008.

[33] A.L. Custódio, J.E. Dennis, Jr., and L.N. Vicente. Using simplex gradients of nonsmooth functions in direct search methods. IMA Journal of Numerical Analysis, 28(4):770–784, 2008.

[34] A.L. Custódio, H. Rocha, and L.N. Vicente. Incorporating minimum Frobenius norm models in direct search. Computational Optimization and Applications, 46(2):265–278, 2010.

[35] A.L. Custódio, K. Scheinberg, and L.N. Vicente. Methodologies and software for derivative-free optimization. In T. Terlaky, M.F. Anjos, and S. Ahmed, editors, Advances and Trends in Optimization with Engineering Applications, MOS-SIAM Book Series on Optimization, chapter 37. SIAM, Philadelphia, 2017.

[36] A.L. Custódio and L.N. Vicente. Using sampling and simplex derivatives in pattern search methods. SIAM Journal on Optimization, 18(2):537–555, 2007.

[37] C. Davis and W. Hare. Exploiting known structures to approximate normal cones. Math. Oper. Res., 38(4):665–681, 2013.

[38] D. Echeverrá Ciaurri, O.J. Isebor, and L.J. Durlofsky. Application of derivative-free methodologies to generally constrained oil production optimization problems. Procedia Computer Science, 1(1):1301–1310, 2010.

[39] G. Fasano, G. Liuzzi, S. Lucidi, and F. Rinaldi. A Linesearch-Based Derivative-Free Approach for Nonsmooth Constrained Optimization. SIAM Journal on Optimization, 24(3):959–992, 2014.

[40] K.R. Fowler, J.P. Reese, C.E. Kees, J.E. Dennis Jr., C.T. Kelley, C.T. Miller, C. Audet, A.J. Booker, G. Couture, R.W. Darwin, M.W. Farthing, D.E. Finkel, J.M. Gablonsky, G. Gray, and T.G. Kolda. Comparison of derivative-free optimization methods for groundwater supply and hydraulic capture community problems. Advances in Water Resources, 31(5):743–757, 2008.

[41] A.E. Gheribi, C. Audet, S. Le Digabel, E. Bélisle, C.W. Bale, and A.D. Pelton. Calculating optimal conditions for alloy and process design using thermodynamic and properties databases, the FactSage software and the Mesh Adaptive Direct Search algorithm. CALPHAD: Computer Coupling of Phase Diagrams and Thermochemistry, 36:135–143, 2012.

[42] A.E. Gheribi, C. Robelin, S. Le Digabel, C. Audet, and A.D. Pelton. Calculating all local minima on liquidus surfaces using the FactSage software and databases and the Mesh Adaptive Direct Search algorithm. The Journal of Chemical Thermodynamics, 43(9):1323–1330, 2011.

[43] C.M. Giuliani and E. Camponogara. Derivative-free methods applied to daily production optimization of gas-lifted oil fields. Computers & Chemical Engineering, 75:60–64, 2015.

[44] L. Han and G. Liu. On the convergence of the UOBYQA method. J. Appl. Math. Comput., 16(1-2):125–142, 2004.

[45] W. Hare. Compositions of convex functions and fully linear models. Optimization Letters, 11(7):1217–1227, Oct 2017.

[46] W. Hare and M. Jaberipour. Adaptive interpolation strategies in derivative-free optimization: a case study. Pac. J. Optim., to appear, 2018.

[47] W. Hare and M. Macklem. Derivative-free optimization methods for finite minimax problems. Optimization Methods and Software, 28(2):300–312, 2013.

[48] W. Hare and J. Nutini. A derivative-free approximate gradient sampling algorithm for finite minimax problems. Computational Optimization and Applications, 56(1):1–38, 2013.

[49] W. Hare, C. Planiden, and C. Sagastizábal. A derivative-free VU-algorithm for convex finite-max problems. submitted, 2018.

[50] W. L. Hare and A. S. Lewis. Estimating tangent and normal cones without calculus. Math. Oper. Res., 30(4):785–799, 2005.

[51] W.L. Hare and Y. Lucet. Derivative-free optimization via proximal point methods. Journal of Optimization Theory and Applications, 160(1):204–220, 2014.

[52] R. Hooke and T.A. Jeeves. "Direct Search" Solution of Numerical and Statistical Problems. Journal of the Association for Computing Machinery, 8(2):212–229, 1961.

[53] W. Huyer and A. Neumaier. SNOBFIT – Stable Noisy Optimization by Branch and Fit. ACM Trans. Math. Softw., 35(2):9:1–9:25, 2008.

[54] C.T. Kelley. Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition. SIAM Journal on Optimization, 10:43–55, 1999.

[55] C.T. Kelley. Implicit Filtering. Society for Industrial and Applied Mathematics, Philadephia, PA, 2011.

[56] K. Khan, J. Larson, and S. Wild. Manifold sampling for optimization of nonconvex functions that are piecewise linear compositions of smooth components. Technical report, Argonne National Labs, 2018. http://www.mcs.anl.gov/publication/manifold-sampling-optimization-nonconvex-functions-are-piecewise-linear-compositions.

[57] K.C. Kiwiel. A nonderivative version of the gradient sampling algorithm for nonsmooth nonconvex optimization. SIAM Journal on Optimization, 20(4):1983–1994, 2010.

[58] T.G. Kolda, R.M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. SIAM Review, 45(3):385–482, 2003.

[59] J. Larson, M. Menickelly, and S.M. Wild. Manifold sampling for ℓ_1 nonconvex optimization. SIAM Journal on Optimization, 26(4):2540–2563, 2016.

[60] J.C. Meza and M.L. Martinez. On the use of direct search methods for the molecular conformation problem. Journal of Computational Chemistry, 15:627–632, 1994.

[61] R. Mifflin. A superlinearly convergent algorithm for minimization without evaluating derivatives. Mathematical Programming, 9(1):100–117, 1975.

[62] R. Mifflin and C. Sagastizábal. A $\mathcal{VU}$-algorithm for convex minimization. Math. Program., 104(2-3, Ser. B):583–608, 2005.

[63] M. Minville, D. Cartier, C. Guay, L.-A. Leclaire, C. Audet, S. Le Digabel, and J. Merleau. Improving process representation in conceptual hydrological model calibration using climate simulations. Water Resources Research, 50:5044–5073, 2014.

[64] J.J. Moré and S.M. Wild. Benchmarking derivative-free optimization algorithms. SIAM Journal on Optimization, 20(1):172–191, 2009.

[65] P. Mugunthan, C.A. Shoemaker, and R.G. Regis. Comparison of function approximation, heuristic and derivative-based methods for automatic calibration of computationally expensive groundwater bioremediation models. Water Resources Research, 41, 2005.

[66] J. Müller and J. Woodbury. GOSAC: global optimization with surrogate approximation of constraints. Journal of Global Optimization, 69(1):117–136, 2017.

[67] J.A. Nelder and R. Mead. A simplex method for function minimization. The Computer Journal, 7(4):308–313, 1965.

[68] R. Oeuvray. Trust-Region Methods Based on Radial Basis Functions with Application to Biomedical Imaging. PhD thesis, Institut de Mathématiques, École Polytechnique Fédérale de Lausanne, Switzerland, 2005.

[69] R. Oeuvray and M. Bierlaire. A new derivative-free algorithm for the medical image registration problem. International Journal of Modelling and Simulation, 2007.

[70] C. Poissant. Exploitation d'une structure monotone en recherche directe pour l'optimisation de boîtes grises. Master's thesis, École Polytechnique de Montréal, 2018.

[71] M.J.D. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In Advances in optimization and numerical analysis (Oaxaca, 1992), volume 275 of Math. Appl., pages 51–67. Kluwer Acad. Publ., Dordrecht, 1994.

[72] M.J.D. Powell. UOBYQA: Unconstrained optimization by quadratic approximation. Mathematical Programming, 92(3):555–582, 2002.

[73] M.J.D. Powell. On trust region methods for unconstrained minimization without derivatives. Mathematical Programming, 97(3):605–623, 2003.

[74] M.J.D. Powell. Least Frobenius norm updating of quadratic models that satisfy interpolation conditions. Mathematical Programming, 100(1):183–215, 2004.

[75] M.J.D. Powell. The NEWUOA software for unconstrained optimization without derivatives. In P. Pardalos, G. Pillo, and M. Roma, editors, Large-Scale Nonlinear Optimization, volume 83 of Nonconvex Optimization and Its Applications, pages 255–297. Springer, 2006.

[76] M.J.D. Powell. The BOBYQA algorithm for bound constrained optimization without derivatives. Technical report, Department of Applied Mathematics and Theoretical Physics, Cambridge University, UK, 2009.

[77] C.J. Price, I.D. Coope, and D. Byatt. A convergent variant of the Nelder Mead algorithm. Journal of Optimization Theory and Applications, 113(1):5–19, 2002.

[78] R.G. Regis. The calculus of simplex gradients. Optimization Letters, 9(5):845–865, 2015.

[79] R.G. Regis and S.M. Wild. CONORBIT: constrained optimization by radial basis function interpolation in trust regions. Optim. Methods Softw., 32(3):552–580, 2017.

[80] E. Renaud, C. Robelin, A.E. Gheribi, and P. Chartrand. Thermodynamic evaluation and optimization of the Li, Na, K, Mg, Ca, Sr, F, Cl reciprocal system. Journal Of Chemical Thermodynamics, 43(8):1286–1298, 2011.

[81] R. T. Rockafellar and R. J.-B. Wets. Variational Analysis, volume 317 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer-Verlag, Berlin, 1998.

[82] Ph.R. Sampaio and Ph.L. Toint. A derivative-free trust-funnel method for equality-constrained nonlinear optimization. Computational Optimization and Applications, 61(1):25–49, 2015.

[83] Th. Sauer and Y. Xu. On multivariate Lagrange interpolation. Mathematics of Computation, 64:1147–1170, 1995.

[84] P.H. Schoute. Mehrdimensionale Geometrie, volume 1. Cornell University Library, 1902.

[85] D.B. Serafini. A Framework for Managing Models in Nonlinear Optimization of Computationally Expensive Functions. PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1998.

[86] B. Talgorn, C. Audet, M. Kokkolaras, and S. Le Digabel. Locally weighted regression models for surrogate-assisted design optimization. Optimization and Engineering, 19(1):213–238, 2018.

[87] Robert Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), 58(1):267–288, 1996.

[88] V. Torczon. On the convergence of pattern search algorithms. SIAM Journal on Optimization, 7(1):1–25, 1997.

[89] P. Tseng. Fortified-descent simplicial search method: A general approach. SIAM Journal on Optimization, 10(1):269–288, 1999.

[90] S.M. Wild, R.G. Regis, and C.A. Shoemaker. ORBIT: optimization by radial basis function interpolation in trust-regions. SIAM Journal on Scientific Computing, 30(6):3197–3219, 2008.

[91] S.M. Wild and C.A. Shoemaker. Global convergence of radial basis function trust region derivative-free algorithms. SIAM J. Optim., 21(3):761–781, 2011.

[92] D. Winfield. Function and Functional Optimization by Interpolation in Data Tables. PhD thesis, Harvard University, USA, 1969.

[93] D. Winfield. Function minimization by interpolation in a data table. Journal of the Institute of Mathematics and its Applications, 12:339–347, 1973.

[94] M.H. Wright. Direct search methods: Once scorned, now respectable. In D.F. Griffiths and G.A. Watson, editors, Numerical Analysis 1995 (Proceedings of the 1995 Dundee Biennial Conference in Numerical Analysis), volume 344 of Pitman Research Notes in Mathematics, pages 191–208, Boca Raton, Florida, 1996. CRC Press.

[95] J. Xu, C. Audet, C.E. DiLiberti, W.W. Hauck, T.H. Montague, A.F. Parr, D. Potvin, and D.J. Schuirmann. Optimal adaptive sequential designs for crossover bioequivalence studies. Pharmaceutical Statistics, 15(1):15–27, 2016.