Methodologies and Software for Derivative-free Optimization

A. L. Custódio 1  K. Scheinberg 2  L. N. Vicente 3

March 14, 2017

1Department of Mathematics, FCT-UNL-CMA, Quinta da Torre, 2829-516 Caparica, Portugal (alcustodio@fct.unl.pt). Support for this author was provided by Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) under the project UID/MAT/00297/2013 (CMA) and the grant PTDC/MAT/116736/2010.

2Department of Industrial and Systems Engineering, Lehigh University, Harold S. Mohler Laboratory, 200 West Packer Avenue, Bethlehem, PA 18015-1582, USA (katyas@lehigh.edu). The work of this author is partially supported by NSF Grants DMS 10-16571, DMS 13-19356, AFOSR Grant FA9550-11-1-0299, and DARPA grant FA 9550-12-1-0406 negotiated by AFOSR.

3CMUC, Department of Mathematics, University of Coimbra, 3001-501 Coimbra, Portugal (lnv@mat.uc.pt). Support for this research was provided by FCT under grants PTDC/MAT/116736/2010 and UID/MAT/00324/2013.
37.1 • Introduction

Derivative-Free Optimization (DFO) methods [53] are typically considered for the minimization/maximization of functions for which the corresponding derivatives are neither available for use, nor can be directly approximated by numerical techniques. Constraints may be part of the problem definition but, similar to the objective function, it is possible that their derivatives are not available. Problems of this type are common in engineering optimization, where the value of the functions is often computed by simulation and may be subject to statistical noise or other forms of inaccuracy. In fact, expensive function evaluations would prevent approximation to derivatives and, even when computed, noise would make such approximations less reliable. In the past couple of decades, intense research has resulted in robust and efficient DFO methods, accompanied by convergence theory and numerical implementations.

The purpose of the present work is to provide an overview of the main classes of state-of-the-art DFO methods, with a focus on the underlying ideas and on the respective classes of problems to which these methods are applicable. Only short descriptions of the methods and algorithms will be given, highlighting the motivational aspects that lead to their rigorous properties. We provide references to detailed algorithmic descriptions, theoretical results, and available software packages.

This chapter is structured around different problem features, rather than around classes of DFO methods as it was the case in [53]. Such a structure is more accessible to users of DFO as it directs the reader to the appropriate DFO algorithm suited for a given problem at hand.

Little notation or terminology needs to be introduced as the contents are given at a general level. However, we point out that by global convergence one means convergence to some form of stationarity regardless of the starting point. The vector norms will be $\ell_2$ ones. The symbol $C^k$ denotes the space of real $n$-dimensional functions whose derivatives are continuous up to the order $k$. The notation $O(A)$ will mean a scalar times $A$, where the scalar does not depend on the iteration counter of the method under analysis (thus depending only on the problem or on algorithmic constants). The dependence of $A$ on the dimension $n$ of the problem will be made explicit whenever appropriate. The chapter is organized as follows. Section 37.2 covers unconstrained optimization. Bound and linearly constrained problems are addressed in Section 37.3. Section 37.4 is devoted to other types of problem constraints. Extensions to global optimization, multiobjective optimization, mixed integer problems, and some additional practical issues are briefly surveyed in Section 37.5.

37.2 • Unconstrained optimization

37.2.1 • Smooth functions

In this subsection we consider the unconstrained minimization of an objective function $f: \mathbb{R}^n \to \mathbb{R}$, at least once continuously differentiable and bounded from below (for which gradients are neither available for use, nor can be accurately approximated).

Sampling and modeling. At each iteration of a trust-region method [46], one
37.2. Unconstrained optimization

typically considers the minimization of a model \( m_k(x_k + s) = f(x_k) + s^\top g_k + \frac{1}{2} s^\top H_k s \), in a region around the current iterate \( x_k \), to obtain a trial point \( x_k + s_k \). The region is frequently defined as a ball of the type \( B(x_k; \Delta_k) = \{ x_k + s \in \mathbb{R}^n : \|s\| \leq \Delta_k \} \), where \( \Delta_k \) denotes the trust-region radius. The model \( m_k \) serves as a local approximation of the function, in particular of its curvature. The vector \( g_k \) can be set to \( \nabla f(x_k) \) in the presence of first-order derivatives (similarly for \( H_k \)), but DFO trust-region methods are based on models built from sampling and some form of interpolation [133, 114, 47].

How well the model approximates the function is reflected by the ratio \( \rho_k = \frac{[f(x_k) - f(x_k + s_k)]}{[m_k(x_k) - m_k(x_k + s_k)]} \). The algorithm proceeds by accepting the trial point \( x_k + s_k \) when \( \rho_k \geq \eta_0 \) for some \( \eta_0 > 0 \). If \( \rho_k < \eta_1 \), with \( \eta_1 \geq \eta_0 \), then the quality of the model may be improved if not deemed sufficiently good, or, if the quality of the model is believed to be good, the trust-region radius is reduced since the step is then deemed to be too large. If \( x_k \) is non-stationary and \( m_k \) has good quality, the algorithm succeeds in accepting a trial point \( x_k + s_k \) as a new iterate (at which the function value is improved) in a finite number of reductions of the trust-region radius \( \Delta_k \) (see [53, Lemmas 10.6 and 10.17]).

In first-order approaches, the quality of a model is measured by its ability to provide accuracy similar to a first-order Taylor expansion:

\[
\begin{align*}
|f(y) - m_k(y)| & \leq \kappa_f \Delta^2 \\
\|\nabla f(y) - \nabla m_k(y)\| & \leq \kappa_g \Delta & \forall y \in B(x_k; \Delta),
\end{align*}
\]

where \( \kappa_f \) and \( \kappa_g \) are positive constants. Models that are \( C^1 \) (with a Lipschitz continuous gradient) and satisfy the above bounds are called fully linear [50]. It was shown in [48] that a subsequence of the iterates generated by a model-based trust-region method drives the gradient to zero, under the condition that fully linear models are available when necessary. This result was further improved in [52] for the whole sequence of iterates, including the case where \( \eta_0 = 0 \), which means that any decrease in the function value is sufficient to accept a new point.

If convergence to second-order stationarity points is desired, then fully quadratic models [50] need to be considered. In this case the models should be \( C^2 \) (with a Lipschitz continuous Hessian) and satisfy:

\[
\begin{align*}
|f(y) - m(y)| & \leq \kappa_f \Delta^3 \\
\|\nabla f(y) - \nabla m(y)\| & \leq \kappa_g \Delta^2 \\
\|\nabla^2 f(y) - \nabla^2 m(y)\| & \leq \kappa_h \Delta & \forall y \in B(x; \Delta),
\end{align*}
\]

Convergence to second-order stationary points is established in [52].

Building a (fully linear or fully quadratic) model based on a sample set raises questions related to the choice of the basis functions used in the model definition and to the geometry of the sample set. The use of polynomial models is quite attractive due to its simplicity, and in [50, 51] a first systematic approach to the subject of sampling geometry when using this class of functions was proposed (introducing the notion of \( \Lambda \)-poised sets, which is related to Lagrange polynomials and ensures fully linear or fully quadratic models). The strict need of controlling geometry or considering model-improvement steps was questioned in [70], where good numerical results were reported for an interpolation-based trust-region method (using complete quadratic models) which ignores the geometry of the sample sets. In [123] an example was given showing that geometry cannot be totally ignored and that some form of model improvement is necessary, at least when the size of the model gradient...
becomes small (a procedure known as the criticality step, which then ensures that the trust-region radius converges to zero). In [123] an interpolation-based trust-region method was proposed which resorts to geometry-improving steps only when the model gradient is small. Global convergence for this method is the result of a self-correction property inherent in the combination of trust regions and polynomial interpolation models.

Quadratic functions are particularly well suited to capture curvature [53]. In a context of expensive function evaluation, construction of a complete quadratic model, which requires \((n+1)(n+2)/2\) functions evaluations, could be unaffordable. A typical approach is to consider minimum Frobenius norm models, which are commonly built when at least \(n+1\) sampling points are available for use, allowing at least to compute a fully linear model. Some variants minimize the Frobenius norm of the model Hessian [49], since the norm of the model Hessian is connected with the accuracy of the model. Other approaches, inspired by quasi-Newton methods, use a least updating minimum Frobenius norm strategy, by minimizing the difference between the current and the previous model Hessians [117]. The minimization of the \(\ell_1\)-norm of the model Hessian has also been proposed to build accurate models from relatively small sample sets [30]. Inspired by the sparse solution recovery theory developed in compressed sensing, the underlying idea is to take advantage of the sparsity of the Hessian in cases where the sparsity structure is not known in advance. Algorithms to compute fully linear and fully quadratic models, in the context of polynomial interpolation or regression, can be found in [50, 51] (see also [53]).

An alternative to polynomial bases are radial basis functions (RBFs) [39, 115]. An RBF is defined by the composition of an univariate function and a function measuring the distance to a sample point. Thus, it is constant on a sphere and has a structure different from polynomials (more nonlinear; potentially more nonconvex). Models based on RBFs typically involve a linear polynomial tail and can be made fully linear. The use of RBFs in model-based trust-region methods was analyzed in [132].

Currently, several solvers implementing interpolation-based trust-region methods are available to the community. Quadratic polynomial models are in the heart of DFO [2] and NEWUOA [118] computational codes. In the first case, when the size of the sampling set is not large enough to build a complete quadratic interpolation model, minimum Frobenius norm models are computed. In contrast, NEWUOA [118] uses the least updating minimum Frobenius norm strategy, described above. Good numerical results on unconstrained problems were also reported for the BC-DFO code [76], an interpolation based trust-region method developed for bound constrained optimization (see Section 37.3 below). Models based on RBFs are implemented in ORBIT [131].

**Sampling using simplex sets.** In turn, direct-search methods use function values from sampling only to make algorithmic decisions, without explicit or implicit modeling of the function. However, the geometry of the sample sets continues to play a crucial role in the algorithmic design and convergence properties.

One possibility is to sample at the vertices of a simplex set, which are in number \(n+1\), exactly as many points as required to build a fully linear model. The goal of each iteration in the well known Nelder-Mead algorithm [111] is to improve the worst vertex of a simplex, and for this purpose a number of operations are performed (reflection, expansion, outside contraction, inside contraction, and shrink).

The various simplex operations allow the method to follow the curvature of the
37.2. Unconstrained optimization

function which explains its good performance in many problems.

However, all simplex operations but shrinks can deteriorate the simplex geometry (an evident example are the expansions), thus it becomes difficult to establish convergence for the original algorithm. In fact, a $C^2$ strictly convex function has been constructed in [110] for $n = 2$ showing that the algorithm [111] fails to converge to the minimizer (by generating an infinite sequence of inside contractions). Convergence can be established for $n = 1$ (see [96] or Exercise 7 of Chapter 8 of [53]) and for $n = 2$ for functions where the Hessian is always positive definite and when no simplex expansions are allowed [95]. Modified variants have been proposed, yielding global convergence in $\mathbb{R}^n$, by including strategies like monitoring the simplex geometry and then possibly attempting a poll-type step (see below) together with using a sufficient decrease condition for accepting new points [127] (see the survey in [53]).

Numerical implementations of variants of the Nelder-Mead method can be found in [8] or in the Matrix Computation Toolbox [9] (see the function \texttt{NMSMAX}).

Sampling using positive spanning sets. Direct-search methods can also be of directional type, where the function is evaluated along directions in positive spanning sets [61]. (A positive spanning set (PSS) is a set of vectors that spans $\mathbb{R}^n$ with nonnegative coefficients.)

Typically, these methods evaluate the objective function at points of the form $x_k + \alpha_k d$, $d \in D_k$, where $x_k$ represents the current iterate, $\alpha_k$ the current step size parameter, and $D_k$ denotes a PSS. This procedure (called polling) is attempted with the goal of decreasing the current best function value. When only simple decrease is required, polling is successful if $f(x_k + \alpha_k d) < f(x_k)$, for some $d \in D_k$. Similarly to trust-region methods, several authors proposed the use of sufficient decrease strategies [105, 93], where success requires $f(x_k + \alpha_k d) < f(x_k) - \rho(\alpha_k)$, for some $d \in D_k$, and where $\rho(\cdot)$ represents a forcing function (namely a non-negative, non-decreasing function satisfying $\rho(t)/t \to 0$ when $t \to 0$). When no improvement is found, $\alpha_k$ is decreased. When polling is successful, $\alpha_k$ is kept constant or increased.

A property of a PSS essential for the minimization of a smooth function is that at least one of its vectors is a descent direction, regardless where the negative gradient is [55, 93]. Thus, unless the current iterate is already a first-order stationary point, the algorithm will succeed in finding a better point in a finite number of reductions of the step size. As in model-based trust-region methods, where the trust-region radius is guaranteed to converge to zero, in direct search a subsequence of step sizes will also converge to zero. In fact, imposing sufficient decrease promotes unsuccessful iterations with consequent reductions of the step size, and using the boundedness from below of the function, one can easily ensure convergence to zero for a subsequence of step sizes [93]. When only simple decrease is required, one has to implicitly keep a distance of the order of the step size among all iterates, and the typical way to achieve it is by generating PSSs such that all trial points lie in underlying integer lattices [64, 125, 22].

Using simple decrease and a finite number of PSSs through the iterations, it was proved in [125] that the gradient is driven to zero for a subsequence of the iterates. Such an algorithmic framework was improved, generalized, and analyzed in [22] and coined generalized pattern search (see also [13]). It was shown in [93] that an infinite number of PSSs can be used when sufficient decrease is imposed (an approach known as generating set search), as long as they are uniformly non-degenerate (meaning that their cosine measure [93] is bounded away from zero).
Polling can be opportunistic (when moving to the first point \( x_k + \alpha_k d \) yielding the desired decrease) or complete (when the best of the points \( x_k + \alpha_k d \) is taken in \( D_k \) and then compared with \( x_k \)). Doing complete polling leads to the convergence of the whole sequence of gradients to zero [93] (under the additional condition that the step size converges to zero, which occurs naturally when imposing sufficient decrease or if the step size is never increased). When polling is not complete, for instance when the first poll direction leading to descent is taken, the order of the poll directions has some influence in the numerical performance of the method (see [24, 60]).

Nowadays, several implementations of direct-search methods of directional type are available, such as DFL [1], HOPSPACK [5], NOMAD [10], and SID-PSM [12]. Even if most of these solvers offer additional features, polling is common to all of them.

37.2.2 • Non-smooth functions

In the presence of non-smoothness, the cone of descent directions can be arbitrarily narrow (see the example provided in [93, Page 441]). Thus, the use of a finite number of PSSs may not guarantee the existence of a descent direction among the poll vectors, and can cause stagnation of the optimization process. This fact was the main motivation for considering more general sets of directions [54] (see also [18], where the motivation arose from a practical context).

To rigorously avoid stagnation and guarantee some form of convergence (as defined below), poll vectors must therefore be asymptotically dense in the unit sphere. When simple decrease is used, all the generated trial points are required to belong to integer lattices and Mesh Adaptive Direct Search (MADS) [24] offers a framework to do so while using infinitely many directions (and taking them from PSSs if desired). If sufficient decrease is imposed, then the computation of new points is free of rules, and the set of poll directions could be simply randomly generated in the unit sphere [130] (an approach here denoted by RdDS).

In the absence of smoothness, convergence can be established by proving the non-negativity of some form of generalized directional derivatives at a limit point of the sequence of iterates and along all normalized directions. To do so, the authors in [22, 24] proposed the use of Clarke [44] analysis for locally Lipschitz continuous functions. As a consequence of using asymptotically dense sets of directions, a hierarchy of convergence results was derived in [24], depending on the level of non-smoothness present in the function. More recently, using Rockafellar generalized directional derivatives [121], the convergence results were extended to discontinuous functions [130]. Second-order results can be found in [14].

Simplex gradients [90] have been suggested as a possibility to define directions of potential descent. A simplex gradient can be regarded as the gradient of a particular linear interpolation model, requiring the evaluation of the function in a simplex (and its quality as an approximation to the gradient in the continuous differentiable case is analyzed in [90, 53]). Simplex gradients are also a possibility to approximate a direction in the Clarke subdifferential [44], defined for Lipschitz continuous functions as the set \( \partial f(x) = \{ \zeta \in \mathbb{R}^n : f^\circ(x; d) \geq \zeta^T d \text{ for all } d \in \mathbb{R}^n \} \), where \( f^\circ(x; d) \) represents the Clarke generalized directional derivative at \( x \) along \( d \) (and its quality as an approximation to such generalized gradients was analyzed in [56]).

In practice, non-smooth functions are frequently non-smooth compositions of
smooth functions. Lower-$C^k$ functions [122], for instance, are characterized by being locally given as a maximum of $C^k$ functions. Convex functions are lower-$C^2$ [122]. Trivially, $f = \max\{f_1, \ldots, f_m\}$ is a lower-$C^k$ function, provided that each $f_i$ is $C^k$. In [85] (see references therein) minmax problems of this form have been addressed, when the $f_i$’s are $C^1$ functions, by considering simplex gradients as approximations to generalized gradients in a line search approach. The general lower-$C^2$ case was considered in [32], adapting ideas from convex non-smooth optimization.

Another possibility for optimizing a non-smooth function without derivatives is by approximating it by a family of smoothing functions (see [74, 92, 113]). The smoothing functions typically depend on a parameter, which must be then driven asymptotically, and may require prior knowledge of the non-smooth structure of the function. Regarding numerical implementations, NOMAD [10, 97] is a reference for non-smooth unconstrained DFO using direct search. In this solver, two different instances are available to build the asymptotically dense sets of directions in the unit sphere fulfilling the integer lattice requirements, namely the probabilistic LT-MADS [24] and the deterministic ORTHOMADS [16].

37.2.3 · Noisy functions

Simplex gradients are also used as search directions for the optimization of noisy functions. In implicit filtering [36], a (not too refined) line search is performed along a negative simplex gradient. A quasi-Newton scheme is then used for curvature approximation. Such ingredients equip the method in [36] to noisy problems in the hope that it can escape from spurious minimizers. A detailed description of the algorithm and corresponding convergence results can be found in the recent book [91]. A numerical implementation, called IFFCO, is available at [6].

In the presence of noise it is natural to consider least-squares regression techniques (see Chapter 4 in [53]) and use them in trust-region methods. However, when the level of noise is large, this type of models may over-fit the available data. In [89], assuming the knowledge of an upper bound for the level of noise present in function evaluations, it was suggested to relax the interpolation conditions using the corresponding bound. In [34] it was suggested instead to incorporate the knowledge about the noise level from each function evaluation in a weighted regression. When the level of noise is sufficiently small relatively to the trust radius, trust-regions methods based on weighted regression models retain global convergence to stationary points [34].

If the noise present in the function evaluation has a stochastic nature, then a simple possible approach would be to replicate function evaluations performed at each point, conferring accuracy to the estimation of the real corresponding function value. This procedure as been followed to adapt simplex type methods [20] and interpolation-based trust-region methods [62] to noisy optimization. Recently, in the context of direct search using PSSs [43], replication techniques were also applied to smooth and non-smooth functions computed by Monte Carlo simulation. Statistical based approaches, namely by using hypotheses tests, have also been suggested for providing confidence to the decision of accepting a new point, when using direct search in the presence of stochastic noise [126, 124].
37.2.4 Worst case complexity and global rates

The analysis of global convergence of algorithms can be complemented or refined by deriving worst case complexity (WCC) bounds for the number of iterations or function evaluations, an information which may be valuable in many practical instances. Derivative-free or zero-order methods have also been recently analyzed with the purpose of establishing their WCC bounds. As in gradient-based methods (see [112, 78, 40]), it was shown in [129] a WCC bound of $O(\epsilon^{-2})$ for the number of iterations of direct-search methods (using PSSs and imposing sufficient decrease), when applied to a smooth, possibly non-convex function. Such type of bound translates into a sublinear global rate of $1/\sqrt{k}$ for the decay of the norm of the gradient. Note that these rates are called global since they are obtained independently of the starting point. In DFO it becomes also important to measure the effort in terms of the number of function evaluations: the corresponding WCC bound for direct search is $O(n^2\epsilon^{-2})$. DFO trust-region methods achieve similar bounds and rates [73]. The authors in [41] have derived a better WCC bound of $O(n^2\epsilon^{-3/2})$ for their adaptive cubic overestimation algorithm but using finite differences to approximate derivatives.

In the non-smooth case, using smoothing techniques, it was established a WCC bound of $O\left((-\log(\epsilon))\epsilon^{-3}\right)$ iterations (and $O\left(n^3(-\log(\epsilon))\epsilon^{-3}\right)$ function evaluations) for the zero-order methods in [73, 74, 113], where the threshold $\epsilon$ refers now to the gradient of a smoothed version of the original function and the size of the smoothing parameter. Composite DFO trust-region methods [73] can achieve $O(\epsilon^{-2})$ when the non-smooth part of the composite function is known.

In [112, Section 2.1.5] it is also shown that the gradient method achieves an improved WCC bound of $O(\epsilon^{-1})$ if the function is convex and the solutions set is nonempty. Correspondingly, the global decaying rate for the gradient is improved to $1/k$. Due to convexity, the rate $1/k$ holds also for the error in function values. For derivative-free optimization, direct search [67] attains the $O(\epsilon^{-1})$ bound ($O(n^2\epsilon^{-1})$ for function evaluations) and the global rate of $1/k$ in the convex (smooth) case. As in the gradient method, direct search achieves an r-linear rate of convergence in the strongly convex case [67]. The analysis can be substantially simplified when direct search does not allow an increase in the step size (see [94]).

The factor of $n^2$ has been proved to be approximately optimal, in a certain sense, in the WCC bounds for the number of function evaluations attained by direct search (see [68]).

37.2.5 Models and descent of probabilistic type

The development of probabilistic models in [30] for DFO and the benefits of randomization for deterministic first-order optimization, led to the consideration of trust-region methods where the accuracy of the models is given with some positive probability [31]. It has been shown that provided the models are fully linear with a certain probability, conditioned to the prior iteration history, the gradient of the objective function converges to zero with probability one. In this trust-region framework, if $p_k \geq p_0 > 0$ and the trust-region radius is sufficiently small relatively to the size of the model gradient $g_k$, then the step is taken and the trust-region radius is possibly increased. Otherwise the step is rejected and the trust-region radius is decreased. It is shown in [31] that global convergence to second-order stationary points is also attainable almost surely.
Not surprisingly, one can define descent in a probabilistic way similarly as for fully linear models. A set of directions is probabilistically descent if at least one of them makes an acute angle with the negative gradient with a certain probability. Direct search based on probabilistic descent has been proved globally convergent with probability one [77]. Polling based on a reduced number of randomly generated directions (which can go down to two) satisfies the theoretical requirements [77] and can provide numerical results that compare favorable to the traditional use of PSSs.

It has been proved in [77] that both probabilistic approaches (for trust regions and direct search) enjoy, with overwhelmingly high probability, a gradient decaying rate of $1/\sqrt{k}$ or, equivalently, that the number of iterations taken to reach a gradient of size $\epsilon$ is $\mathcal{O}(\epsilon^{-2})$. Interestingly, the WCC bound in terms of function evaluations for direct search based on probabilistic descent is reduced to $\mathcal{O}(mn\epsilon^{-2})$, where $m$ is the number of random poll directions [77].

Recently, in [42], it was proposed and analyzed a trust-region model-based algorithm for solving unconstrained stochastic optimization problems, using random models obtained from stochastic observations of the objective function or its gradient.

### 37.3 Bound and linearly constrained optimization

We now turn our attention to linearly constrained optimization problems in which $f(x)$ is minimized subject to $b \leq Ax \leq c$, where $A$ is an $m \times n$ matrix and $b$ and $c$ are $m$-dimensional vectors. The inequalities are understood componentwise. In particular, if $A$ is the identity matrix, then we have a bound constrained optimization problem. Again, we consider the derivative-free context, where it is not possible to evaluate derivatives of $f$.

**Sampling along directions.** In a feasible method, where all iterates satisfy the constraints, the geometry of the boundary near the current iterate should be taken into account when computing search directions (to allow for sufficiently long feasible displacements). In direct search this can be accomplished by computing sets of positive generators for tangent cones of nearby points, and then using them for polling. (A set of positive generators of a convex cone is a set of vectors that spans the cone with nonnegative coefficients.) If there are only bounds on the variables, such a scheme is ensured simply by considering all the coordinate directions [98]. For general non-degenerate linear constraints, there are schemes to compute such positive generators [100] (for the degenerate case see [17]). If the objective function is continuously differentiable, the resulting direct-search methods are globally convergent to first-order stationary points [100] (see also [93]), in other words, to points where the gradient is in the polar of the tangent cone, implying that the directional derivative is nonnegative for all directions in the tangent cone. Implementations are given in HOPSPACK [5] and PSwarm [11].

If the objective function is non-smooth, one has to use polling directions asymptotically dense in the unit sphere (for which there are two main techniques, either MADS [24] or RdDS [130]). We have seen that in unconstrained optimization global convergence is attained by proving that the Clarke generalized derivative is nonnegative at a limit point for all directions in $\mathbb{R}^n$ — which, in the presence of bounds/linear constraints, trivially includes all the directions of the tangent cone at the limit point. One can also think of hybrid strategies, combining positive generators and dense generation (see the algorithm CS-DFN [69] for bound constrained
optimization where the coordinate directions are enriched by densely generated ones when judged efficient).

**Sampling and modeling.** Active-set type approaches have also been considered in the context of trust-region methods for derivative-free bound constrained optimization. One difficulty is that the set of interpolation points may get aligned at one or more active bounds and deteriorate the quality of the interpolation set. In [76] an active-set strategy is considered by pursuing minimization in the subspace of the free (non-active) variables, circumventing such a difficulty and saving function evaluations from optimization in lower dimensional subspaces. The respective code is called BC-DFO [76].

In other strategies, all the constraints are included in the trust-region subproblem. This type of trust-region methods was implemented in the codes BOBYQA [119] (a generalization of NEWUOA [118] for bound constrained optimization) and DFO [2] (which also considers feasible regions defined by continuously differentiable functions for which gradients can be computed). Recently, extensions to linearly constrained problems have been provided in the codes LINCOA [120] and LCOBYQA [83].

### 37.4 Nonlinearly constrained optimization

Consider now the more general constrained problem

\[
\begin{align*}
\min \ f(x) \\
\text{s.t.} \quad x \in \Omega = \Omega_r \cap \Omega_{nr}.
\end{align*}
\]

The feasible region of this problem is defined by relaxable and/or unrelaxable constraints. The non-relaxable constraints correspond to \( \Omega_{nr} \subseteq \mathbb{R}^n \). Such constraints have to be satisfied at all iterations in an algorithmic framework for which the objective function is evaluated. Often they are bounds or linear constraints, as considered above, but they can also include hidden constraints (constraints which are not part of the problem specification/formulation and their manifestation comes in the form of some indication that the objective function could not be evaluated). In contrast, relaxable constraints, corresponding to \( \Omega_r \subseteq \mathbb{R}^n \), need only to be satisfied approximately or asymptotically, and are often defined by algebraic inequality constraints.

Most of the globally convergent derivative-free approaches for handling nonlinearly constrained problems have been of direct search or line search type, and we summarize such activity next.

**Unrelaxable constraints.** Feasible methods may be the only option when all the constraints are unrelaxable (\( \Omega_r = \mathbb{R}^n \)). In addition they generate a sequence of feasible points, thus allowing the iterative process to be terminated prematurely with a guarantee of feasibility for the best point tested so far. This is an important feature in engineering design problems because the engineer does not want to spend a large amount of computing time and have nothing useful (i.e., feasible) to show for it. One way of designing feasible methods is by means of the barrier function (coined extreme barrier in [24])

\[
f_{\Omega_{nr}}(x) = \begin{cases} 
  f(x) & \text{if } x \in \Omega_{nr}, \\
  +\infty & \text{otherwise}. 
\end{cases}
\]
It is not necessary to evaluate \( f \) at infeasible points where the value of the extreme barrier function can be set directly to \( +\infty \). Hidden constraints are fundamentally different because it is not known a priori if the point is feasible. Direct-search methods take action solely based on function value comparisons and are thus appropriate to use in conjunction with an extreme barrier function. In the context of direct-search methods of directional type for non-smooth functions, we have seen that there are two known ways of designing globally convergent algorithms (MADS [24] and RdDS [130]). In each case, one must use sets of directions whose union (after normalization if needed) is asymptotically dense in the unit sphere of \( \mathbb{R}^n \). The resulting approaches are then globally convergent to points where the Clarke directional derivative is nonnegative along all directions in the (now unknown) tangent cone. An alternative to extreme barrier when designing feasible methods is the use of projections onto the feasible set, although this might require the knowledge of the derivatives of the constraints and be expensive or unpractical in many instances (see [106] for such an approach).

**Relaxable constraints.** In the case where there are no unrelaxable constraints (rather than those of the type \( b \leq Ax \leq c \)), one can use a penalty term by adding to the objective function a measure of constraint violation multiplied by a penalty parameter, thus allowing starting points that are infeasible with respect to the relaxable constraints. In this vein, an approach based on an augmented Lagrangian method was suggested (see [99]), considering the solution of a sequence of subproblems where the augmented Lagrangian function takes into account only the nonlinear constraints and is minimized subject to the remaining ones (of the type \( b \leq Ax \leq c \)). Each problem can then be approximately solved using an appropriate DFO method such as a (directional) direct-search method. This application of augmented Lagrangian methods yields global convergence results to first-order stationary points of the same type of those obtained under the presence of derivatives. In [65] a more general augmented Lagrangian setting is studied, where the problem constraints imposed in the subproblems are not necessarily of linear type.

In turn, algorithms for inequality constrained problems, based on smooth and non-smooth penalty functions were developed and analyzed in [101, 103, 69], imposing sufficient decrease and handling bound/linear constraints separately, proving that a subset of the set of limit points of the sequence of iterates satisfy the first-order necessary conditions of the original problem. Numerical implementations can be found in the DFL library [1].

Filter methods from derivative-based optimization [72] have also been used in the context of relaxable constraints in DFO. In a simplified way, these methods treat a constrained problem as a bi-objective unconstrained one, considering as goals the objective function and a measure of the constraints violation, but giving priority to the latter one. Typically a restoration procedure is considered to compute nearly feasible points. A first step along this direction in DFO was suggested in [23], for direct-search methods using a finite number of PSSs. The filter approach in [63] (where an envelope around the filter is used as a measure of sufficient decrease) guarantees global convergence to a first-order stationary point. Inexact restoration methods from derivative-based optimization [107] have also been applied to DFO, again algorithms alternating between restoration and minimization steps. In [108] an algorithm is proposed for problems with ‘thin’ constraints, based on relaxing feasibility and performing a subproblem restoration procedure. Inexact restoration has been applied in [38] to optimization problems where derivatives of
the constraints are available for use, thus allowing derivative-based methods in the
restoration phase.

**Relaxable and unrelaxable constraints.** The first general approach to consider
both relaxable and unrelaxable constraints is called progressive barrier [25]. It allows
the handling of both types of constraints, by combining MADS for unrelaxable
constraints with non-dominance filter-type concepts for the relaxable constraints
(see the consequent developments in [27]). An alternative to progressive barrier
has been proposed in [79], handling the relaxable constraints by means of a merit
function instead of a filter, and using RdDS for the unrelaxable ones. The merit
function and the corresponding penalty parameter are only used in the evaluation
of an already computed step. An interesting feature of these two approaches is that
constraints can be considered relaxable until they become feasible whereupon they
can be transferred to the set of unrelaxable constraints. Both of them exhibit global
convergence properties.

**Model-based trust-region methods.** On the model-based trust-region side of
optimization without derivatives, nonlinear constraints have been considered mostly
in implementations and in a relaxable mode.

Two longstanding software approaches are COBYLA [116] (where all the func-
tions are modeled linearly by interpolation), see also [37], and DFO [49] (where all
the functions are modeled quadratically by interpolation).

Another development avenue has been along composite-step based SQP [46,
Section 15.4]. Here one models the objective function by quadratic functions and
the constraints by linear ones. The first approach has been proposed in [45] and [33],
using, respectively, filters and merit functions for step evaluation.

More recently, a trust-funnel method (where the iterates can be thought as
flowing towards a critical point through a funnel centered on the feasible set; see [75])
was proposed in [109] for the particular equality constrained case. Another approach
(and implementation code NOWPAC) has been proposed in [29] for equalities and
inequalities and inexact function evaluations.

### 37.5 General extensions

In real life applications, it is often the case that the user can supply a starting point
for the optimization process and that some (local) improvement over the provided
initialization may already fulfill the original goals. Nevertheless, there are situations
where global minimizers are requested and/or good initial guesses are unknown. Ex-
tensions of DFO to global optimization try to cope with such additional difficulties.
One possibility is to partition the feasible region into subdomains, which are locally
explored by a DFO procedure in an attempt to identify the most promising ones.
DIRECT [88] and MCS [87] follow this approach being the latter enhanced by local
optimization based on quadratic polynomial interpolation (see the corresponding
codes in [71] and [7]). An alternative is to multistart different instances of a DFO
algorithm from distinct feasible points. Recently, in the context of direct search, it
was proposed to merge the different starting instances when sufficiently close to each
other [57] (see the corresponding code GLODS [4]). Heuristics have been tailored
to global optimization without derivatives, and an example providing interesting
numerical results are evolution strategies like CMA-ES [84] (for which a modified
version is capable of globally converging to stationary points [66]).
DFO algorithms can be equipped with a search step for the purpose of improving their local or global performance (such type of steps are called *magical* in [46]). The paper [35] proposed a search-poll framework for direct search, where a search step is attempted before the poll step. A similar idea can be applied to model-based trust-region algorithms [80]. The search step is optional and does not interfere in the global convergence properties of the underlying methods. Surrogate models (see [104, Section 3.2] and [53, Section 12]) can be built and optimized in a search step such as in [59] for quadratics or in [19] for RBFs. Other possibilities for its use include the application of global optimization heuristics [128, 21]. See the various solvers [10, 11, 12].

Parallelizing DFO methods is desirable in the presence of expensive function evaluations. The poll step of direct search offers a natural parallelization by distributing the poll directions among processors [86]. Asynchronous versions of this procedure [82] are relevant in the presence of considerably different function evaluation times. Several codes [5, 10, 11] offer parallel modes. Subspace decomposition in DFO is also attractive for parallelization and surrogate building [26, 81].

The extension of DFO methods to problems involving integer or categorical variables has also been considered. The methodologies alternate between a local search in the continuous space and some finite exploration of discrete sets for the integer variables. Such discrete sets or structures could be fixed in advance [15] or be adaptively defined [102]. Implementations are available in NOMAD [10] and DFL library [1], respectively.

Multiobjective optimization has also been the subject of DFO. A common approach to compute Pareto fronts consists of aggregating all the functions into a single parameterized one, and it has been done in DFO (see [28] and references therein). In [58] the concept of Pareto dominance was used to generalize direct search to multiobjective DFO without aggregation. Implementations are available in the codes NOMAD [10] and DMS [3], respectively.
Bibliography

[1] DFL. http://www.dis.uniroma1.it/~lucidi/DFL.


