Research paper

Struct Multidisc Optim
DOI 10.1007/s00158-003-0320-9

A constrained, globalized, and bounded Nelder–Mead method for engineering optimization

M.A. Luersen, R. Le Riche, F. Guyon

Abstract One of the fundamental difficulties in engineering design is the multiplicity of local solutions. This has triggered much effort in the development of global search algorithms. Globality, however, often has a prohibitively high numerical cost for real problems. A fixed cost local search, which sequentially becomes global, is developed in this work. Globalization is achieved by probabilistic restarts. A spacial probability of starting a local search is built based on past searches. An improved Nelder–Mead algorithm is the local optimizer. It accounts for variable bounds and nonlinear inequality constraints. It is additionally made more robust by reinitializing degenerated simplexes. The resulting method, called the Globalized Bounded Nelder–Mead (GBNM) algorithm, is particularly adapted to tackling multimodal, discontinuous, constrained optimization problems, for which it is uncertain that a global optimization can be afforded. Numerical experiments are given on two analytical test functions and two composite laminate design problems. The GBNM method compares favorably with an evolutionary algorithm, both in terms of numerical cost and accuracy.

Key words global constrained optimization, Nelder–Mead method, composite laminated plates

Received: 24 May 2002
Revised version: 3 May 2003
Published online: 7 October 2003

M.A. Luersen[1, ✉], R. Le Riche[2] and F. Guyon[3]

[1] CNRS UMR 6138, Lab. de Mécanique de Rouen, France, and Mechanical Department, CEFET-PR, Brazil
e-mail: Marco.Luersen@insa-rouen.fr
[2] CNRS URA 1884 / SMS, Ecole des Mines de Saint Etienne, France
e-mail: leriche@emse.fr
[3] Lab. de Bio-statistiques Bio-mathématiques, Univ. Paris 7, France
e-mail: guyon@urbb.jussieu.fr

1 Introduction

Complex engineering optimization problems are characterized by calculation-intensive system simulations, difficulties in estimating sensitivities (when they exist), the existence of design constraints, and a multiplicity of local solutions.

Acknowledging the last point, much research has been devoted to global optimization (e.g., Törn and Zilinskas 1989; Bäck 1996). The high numerical cost of global optimizers has been at the heart of subsequent efforts to speed up the search either by adding problem-specific knowledge to the search, or by mixing efficient local algorithms with global algorithms. There are many ways in which local and global searches can cooperate.

The simplest strategy is to link the searches in series, meaning that firstly, a global optimization of limited cost is executed, the solution of which is refined by a local search. An example of the serial hybrid is given in Shang *et al.* (2001), in which simulated annealing, the global optimizer, is coupled with a sequential quadratic programming and a Nelder–Mead algorithm.

A large number of parallel local-global searches have been proposed (Törn and Zilinskas 1989; Durand and Alliot 1999; Okamoto *et al.* 1998) and analyzed (Törn 1978; Goldberg and Voessner 1999). In these cases, iterations of global and local algorithms are intertwined. One can further classify parallel hybrids into those in which the local searches converge, and those in which local searches may be prematurely stopped. Memetic genetic algorithms (Moscato 1989) and multistart methods (e.g., deterministic restart in Barhen *et al.* (1997); random restarts in Hu *et al.* (1994)) are examples of the former. The latter are usually based on clustering steps, in which local searches approaching already explored regions of the design space are abandoned (Törn 1978; Hickernell and Yuan 1997).

When considering a real engineering optimization problem, a common situation is that the affordable total number of analyses is limited, that the presence of spuri-

ous local minima is unknown, and that it is uncertain if it will be possible to complete as few as two local searches. Nevertheless, achieving global results remains an objective of the optimizer. This typically occurs when dealing with an unknown function of less than 20 variables, for which one is willing to wait for about 1000 evaluations of the objective function. In such a case, a local-global method based on restarts is the safest strategy because it can terminate in a short time (the length of a single local search). The method described in this article, the Globalized Bounded Nelder–Mead algorithm, GBNM, is meant to be a black-box local-global approach to real constrained optimization problems. A restart procedure that uses an adaptive probability density keeps a memory of past local searches. Constraints and limits on variables are taken into account through adaptive penalization and projection, respectively. Finally, GBNM can be applied to discontinuous (no gradient information needed), non-convex functions, since the local searches are based on a variant of the Nelder–Mead algorithm (Nelder and Mead 1965). Improvements to the Nelder–Mead algorithm consist of simplex degeneracy detection and handling through reinitialization.

This paper is structured as follows. The GBNM algorithm is described in Sect. 2. The constraint handling technique is presented in Sect. 3. Section 4 reports numerical experiments on two analytical functions and two composite laminated plate design problems. The GBNM algorithm is compared with a steady-state evolutionary algorithm (Bäck 1996).

2
Globalization of a local search by probabilistic restarts

Local optimizers can make up a global search when repeatedly started from different points. The simplest restart methods initialize the search either from a regular grid of points, or from randomly chosen points. In the first case, one needs to know how many restarts will be performed to calculate the size of the mesh. In the other case, knowledge of past searches is not used, so that the same local optima may be found several times, costing vast unnecessary effort. In the current work, the number of restarts is unknown beforehand because a maximum number of analyses is imposed and the cost of each local search is unknown. A grid method cannot be applied here. Also, a memory of previous local searches is kept by building a spacial probability density of starting a search.

2.1
Probabilistic restart

The probability, $p(x)$, of having sampled a point x is described here by a Gaussian Parzen-windows approach (Duda *et al.* 2001). This method can be considered as a smoothed version of the histogram technique, the histograms being centered at selected sampled points $p(x)$ is written as

$$p(x) = \frac{1}{N} \sum_{i=1}^{N} p_i(x) \ , \tag{1}$$

where N is the number of points already sampled, and p_i is the normal multi-dimensional probability density function,

$$p_i(x) = \frac{1}{(2\pi)^{\frac{n}{2}} (\det(\Sigma))^{\frac{1}{2}}} \times$$

$$\exp\left(-\frac{1}{2} (x - x_i)^T \Sigma^{-1} (x - x_i) \right) , \tag{2}$$

n is the dimension (number of variables), and Σ is the covariance matrix,

$$\Sigma = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_n^2 \end{bmatrix} . \tag{3}$$

The variances, σ_j^2, are estimated by the relation

$$\sigma_j^2 = \alpha \left(x_j^{max} - x_j^{min} \right)^2 , \tag{4}$$

where α is a positive parameter that controls the length of the Gaussians, and x_j^{max} and x_j^{min} are the bounds in the jth direction. Note that, in order to keep the method as simple and cost effective as possible, the variances are kept constant. This strategy would have a cost in terms of the total number of analyses. The probability density is such that $\int_{-\infty}^{\infty} p(x) \, \mathrm{d}x = 1$, but since a bounded domain Ω is considered, a bounded probability $\tilde{p}(x)$ is introduced, i.e.,

$$\tilde{p}(x) = \frac{p(x)}{M} , \qquad M = \int_{\Omega} p(x) \, \mathrm{d}x , \tag{5}$$

so that $\int_{\Omega} \tilde{p}(x) \, \mathrm{d}x = 1$.

The probability density of sampling a new point, $\phi(x)$, is a probability density of not having sampled x before. For its estimation we adopt the following assumption: only the highest point x_H of $\tilde{p}(x)$ has a null probability of being sampled at the next iteration. So, the probability $\phi(x)$ is calculated as

$$\phi(x) = \frac{H - \tilde{p}(x)}{\int_{\Omega} (H - \tilde{p}(x)) \, \mathrm{d}x} , \quad H = \max_{x \in \Omega} \tilde{p}(x) . \tag{6}$$

Figure 1 illustrates $p(x)$, $\tilde{p}(x)$ and $H - \tilde{p}(x)$, in a undidimensional domain.

The maximization of ϕ is not performed exactly, firstly because of its numerical cost, and secondly, as will be seen in Sect. 4.1, because it would be detrimental to the search. Instead, N_r points are chosen randomly and the point that maximizes ϕ is selected to initiate the next

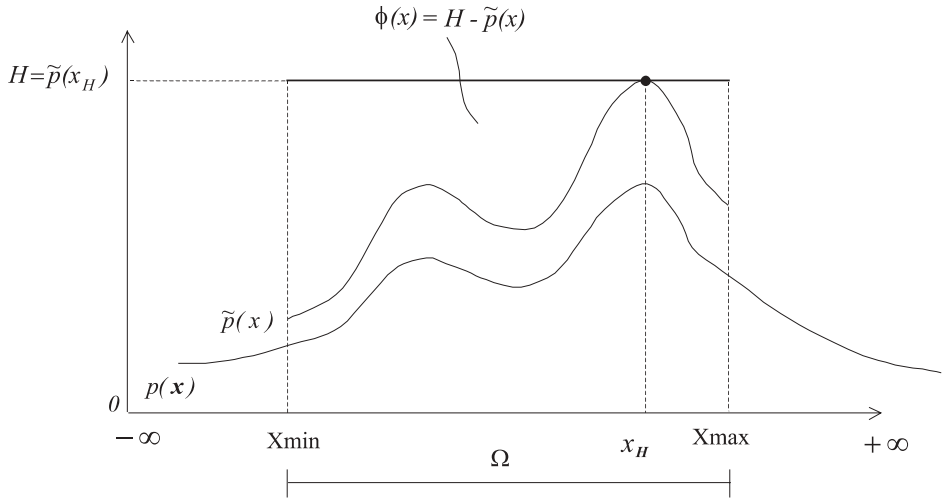

Fig. 1 Probability density functions

search. Note that, in order to maximize ϕ, it is necessary to calculate neither M (5) nor H (6): the maximum of ϕ is the minimum of p, so p only is calculated.

Three parameters influence the probability density p and, consequently, the starting points: the points that are kept for the probability calculation, p; the number of random points used to maximize ϕ, N_r; and the Gaussian length parameter, α. The setting of their values is discussed in the numerical results (Sect. 4.1).

The probabilistic restart procedure can be applied to any local optimizer. In this case, an improved Nelder–Mead algorithm is proposed.

2.2 An improved Nelder–Mead search

The original Nelder–Mead algorithm (Nelder and Mead 1965) and the strategy for bounding variables are summarized in Appendix A. The GBNM algorithm differs from the Nelder–Mead method partly because of a set of restart options (other differences are related to constraints handling (see Sect. 3 and Appendix A)). The purpose of the restarts is twofold.

Firstly, *probabilistic* restarts based on the density p (see (1)) aim at repeating local searches until a fixed total cost, C_{max}, has been reached. The probability of having located a global optimum increases with the number of probabilistic restarts. This is the "globalized" aspect of the method. In the current implementation of probabilistic restarts, the size of the new simplex, a (defined in (A.1)), is a uniform random variable taken between 2 and 10% of the smallest domain dimension.

Secondly, restarts are used to check and improve the convergence of the algorithm. The two restart schemes that are convergence related initialize a new simplex from the current best vertex. The *small* and *large test* restarts use a small and large simplex of sizes a_s and a_l, respectively (see (A.2)).

Convergence of the local Nelder–Mead searches is estimated through three criteria, the *small*, *flat*, or *degenerate* simplex tests. The simplex is small if

$$\max_{k=1,\cdots,n+1}\left(\sum_{i=1}^{n}\left|\frac{e_i^k}{x_i^{max}-x_i^{min}}\right|\right) < \varepsilon_{s1}\ , \tag{7}$$

where e_i^k is the i-th component of the k-th edge, x_i^{max} and x_i^{min} are the bounds in the i-th direction, and ε_{s1} is a termination tolerance. The simplex is flat if

$$|f_H - f_L| < \varepsilon_{s2}\ , \tag{8}$$

where f_H and f_L are the highest and lowest objective functions in the simplex, and ε_{s2} is a tolerance value. The simplex is degenerated if it has collapsed into a subspace of the search domain. This is the most common symptom of a failed Nelder–Mead search (Wright 1996) because the method cannot escape the subspace. More precisely, a simplex is called degenerated here if it is neither small, nor touches a variable bound, and one of the two following conditions is satisfied:

$$\frac{\min\limits_{k=1,n}||\mathbf{e}^k||}{\max\limits_{k=1,n}||\mathbf{e}^k||} < \varepsilon_{s3} \quad \text{or} \quad \frac{\det[\mathbf{e}]}{\prod_k||\mathbf{e}^k||} < \varepsilon_{s4}\ , \tag{9}$$

where $\mathbf{e}^k$ is the k-th edge, $\mathbf{e}$ is the edge matrix, and ε_{s3} and ε_{s4} are small positive constants.

The linking of the three restarts and three convergence tests in the GBNM algorithm is shown in Fig. 2. A memory of past convergence locations is kept, thus preventing unnecessary computations on already analyzed points (third test, *T3*, in the flow chart of Fig. 2). When the simplex is flat, a probabilistic restart is performed (*T4*). A simplex that is degenerated induces a large test iteration (*T8*). When the optimality of the convergence point is unsure, such as a convergence on a variable bound where the simplex has degenerated (*T6*), a small test, which stands for an optimality check, is performed. If the

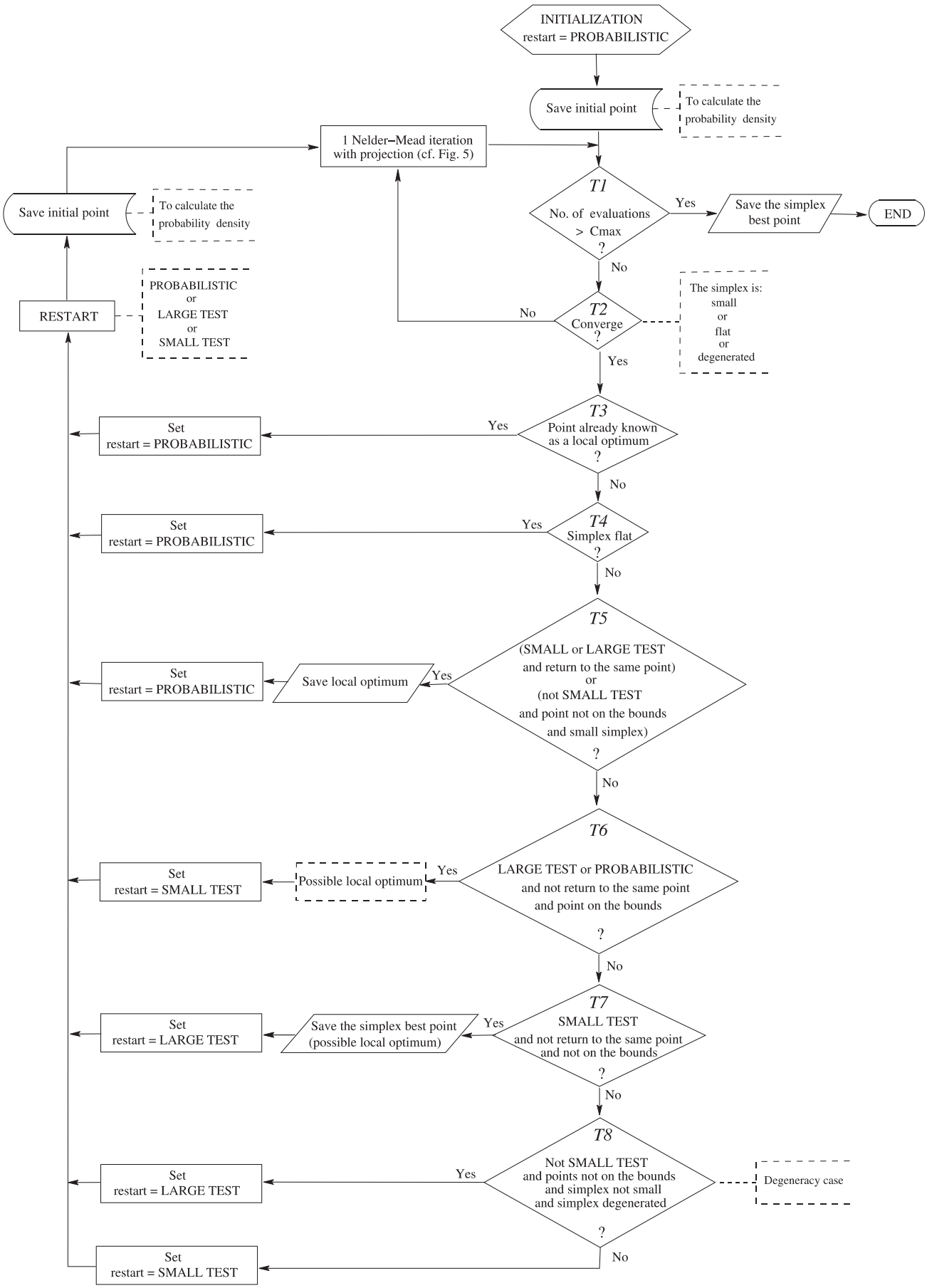

Fig. 2 Linking of restarts and convergence tests in GBNM

small simplex returns to the same convergence point, it is considered to be a local optimum. It should be remembered that the Kuhn and Tucker conditions of mathematical programming are not applicable to the present non-differentiable framework. The tolerances for small and degenerated simplexes, ε_{s1} and $[\varepsilon_{s3},\ \varepsilon_{s4}]$, respectively, may be difficult to tune, so that a simplex that is becoming small may be tagged as degenerated before. Thus, if a degeneration is detected twice consecutively at the same point, the point is taken as a possible optimum, and a probabilistic restart is called. Similarly, if a degeneration is detected after a small test, this point is also saved as a possible optimum, and a large test is ordered.

Once the GBNM algorithm terminates, the list of possible local (eventually global) optima comprises the results of the search. In practice, the calculation of many local or global optima is a benefit of the method in comparison with global optimizers that provide a single solution (e.g., evolutionary algorithms).

3
An adaptive linear penalty for handling constraints

An adaptive linear penalty function is used to handle general inequality constraints. The primal problem,

$$(P)\quad \begin{cases} \min\limits_{x\in\mathcal{S}\subset\Re^n} f(x)\ , \\ \text{such that } g_i(x)\ \leq\ 0\ , \qquad i=1,m\ , \end{cases} \tag{10}$$

is rewritten in a penalized form,

$$(PP)\quad \begin{cases} \min\limits_{x\in\mathcal{S}} \bar{L}(x,\lambda)\ , \quad \text{where} \\ \bar{L}(x,\lambda)=f(x)+\sum\limits_{i=1}^{m}\lambda_i \max(0,g_i(x))\ . \end{cases} \tag{11}$$

This last problem is unconstrained, but appropriate values of the penalty parameters λ_i need to be estimated. More usual penalization approaches are based on quadratic penalties, or ordinary Lagrangian or augmented Lagrangian functions. The current adaptive linear penalty has the following advantages, proofs of which are given in Appendix B using generalized Lagrangian theory (Rockafellar 1976; Minoux 1986; Le Riche and Guyon 2001). With respect to a quadratic penalty, convergence to the feasible optimum can be achieved for finite values of the parameters λ_i. With respect to an ordinary Lagrangian, generalized duality theory can be applied to calculate the λ_i's for a larger class of functions f and g_i. Finally, augmented Lagrangians have more penalty parameters to set than $\bar{L}$. Unlike $\bar{L}$, if f and the g_i's are differentiable, augmented Lagrangians are differentiable at places where $g(x)=0$. This, however, is not a decisive drawback of $\bar{L}$, since a non-differentiable framework is assumed here.

The penalty parameters are updated after each generation of an in-bounds point by the GBNM algorithm. The updating scheme is intuitive since it consists of increasing penalty parameters of violated constraints:

$$\begin{aligned} &\text{if } \ (\bar{L}(x^{new},\lambda^k)\leq \bar{L}(x^{best},\lambda^k))\ , \\ &\quad \lambda_i^{k+1}=\lambda_i^k+s\max(0,g_i(x^{new}))\ , \qquad i=1,m\ , \\ &\quad x^{best}=\arg\min_{x\in\{x^{new},x^{best},\text{simplex vertices}\}} \bar{L}(x,\lambda^{k+1})\ , \\ &\text{end}\ , \end{aligned} \tag{12}$$

where s is a positive step size. This updating strategy is interpreted as a fixed-step approximate-gradient search on the dual function (see Appendix B).

An example of convergence in the dual λ space is now presented. Rosenbrock's test function is minimized in a bounded domain with an additional constraint,

$$\begin{cases} \min\limits_{x_1,x_2\in[0,20]} 100(x_2-x_1^2)^2+(1-x_1)^2\ , \\ \text{such that } 4-x_1^2\leq 0\ . \end{cases} \tag{13}$$

100 independent optimizations with random starting points, $\lambda^0=0$, $\alpha=0.01$ (see (4)), and a step size $s=0.001$ are performed. On average, the penalty parameter stabilizes at $\bar{\lambda}=0.5\pm 1\times 10^{-6}$ after 746 ± 178 analyses. The global minimum, $x_1=2$, $x_2=4$, is always found. It should be noted that, in this case, $\bar{\lambda}=0.5$ is also the Kuhn and Tucker multiplier at the optimum. In general cases like Test 1 below, a larger sensitivity of the converged λ's to the restart points x^0 may occur.

4
Numerical results

In Sect. 4.1, the choice of GBNM parameters is discussed. Results on two test functions are given in Sect. 4.2 and two composite laminate design problems are addressed in Sect. 4.3. The GBNM method is compared with an evolutionary algorithm (EA), both methods handling constraints by the linear adaptive scheme of Sect. 3. The evolutionary algorithm (Bäck 1996) has a steady-state structure (Syswerda 1991) with real encoding, continuous crossover, and Gaussian mutation of the variance $\sigma_i^{mut}=(x_i^{max}-x_i^{min})^2/16$. For fair comparisons, the parameters of the EA chosen for each test are the ones that perform best in 100 independent trials among all combinations of population size (20 or 50), mutation probability (0.15 or 0.20), and crossover probability (0.4 or 0.5).

4.1
GBNM parameter choice

The Gaussian length parameter, α

In this work, α is set to 0.01, which means that one standard deviation away from the Gaussian mean covers about 20% of the domain.

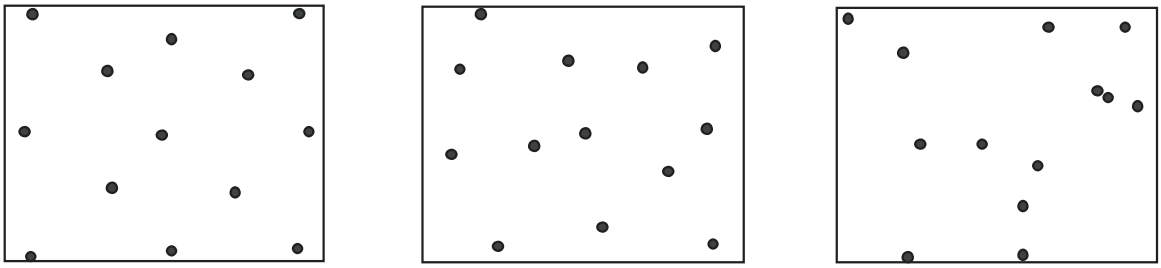

Fig. 3 Starting points in a two dimensional space. The first starting point is at the center of the domain. $N_r = 1000$ (a), $N_r = 10$ (b), and $N_r = 1$ (c)

Points kept for the probability density calculation

In Luersen and Le Riche (2001), three strategies were compared in terms of the probability of not finding at least one local minimum on three multimodal functions, P_{nfm}: the x_i's used in (1) are the starting points, or the starting and local convergence points, or all the points sampled during the search. This last option is memory and time consuming with degraded performance. The second strategy performs best, independently of N_r. It shows that the starting and local convergence points efficiently summarize the topology of the basins of attraction. This scheme is chosen to update p.

Number of random points, N_r

If N_r is equal to 1, the reinitialization is random (see Fig. 3c). If N_r is large, the initial points form a perfect geometric pattern (Fig. 3a). Setting N_r to a small number larger than 1, gives a biased random reinitialization (Fig. 3b). It should be seen as a compromise between the grid and random strategies. The optimum value of N_r depends on the test function: if the basins of attraction are regularly distributed, restarts following a regular pattern (N_r large) are optimal, and vice versa. In Luersen and Le Riche (2001), the probability of not finding at least one local minimum, P_{nfm}, is estimated for N_r ranging from 1 to 1000. On the average of the three test functions, $N_r = 10$ is the value that minimizes this probability. From now on, N_r is set to 10.

Penalty parameters λ_i

With the adaptive penalty scheme of Sect. 3, an imperfect gradient search is performed in the dual space of the penalty parameters, while the GBNM algorithm searches the primal (x) space. For better readibility of the results, the primal and dual searches are decoupled hereafter: 100 runs are first performed, starting from null values of the penalty parameters, $\lambda_i = 0$. The converged values, $\bar{\lambda}_i$, are then averaged and taken as fixed penalty parameters (i.e., setting the step $s = 0$) for the discussed primal searches.

4.2 Analytical test functions

Two analytical functions are now considered to test the GBNM algorithm. The first test, from Michalewicz and Schoenauer (1997), has two variables, two constraints, and is formulated as

$$\begin{cases} \min\limits_{x_1,x_2\in[0.001,20]} -\dfrac{(\sin(2\pi x_1))^3 \sin(2\pi x_2)}{x_1^3(x_1+x_2)} \ , \\ \text{such that} \\ g_1(x_1,x_2) = x_1^2 - x_2 + 1 \le 0 \ , \\ g_2(x_1,x_2) = 1 - x_1 + (x_2-4)^2 \le 0 \ . \end{cases} \tag{14}$$

The global minimum is -0.0958248, at $x_1 = 1.228$, $x_2 = 4.245$. For this problem, using $s = 0.001$, the following stabilized penalties are found: $\bar{\lambda}_1 = 5.5$ and $\bar{\lambda}_2 = 98.4$, obtained after 12204 to 22418 analyses. The second test, from Michalewicz and Schoenauer (1997), has 7 variables and 4 constraints, and is

$$\begin{cases} \min\limits_{x_i\in[-20,20]} (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 + 3(x_4-11)^2 + \\ \qquad 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 \ , \\ \text{such that} \\ -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \le 0 \ , \\ -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \le 0 \ , \\ -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \le 0 \ , \\ 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \le 0 \ . \end{cases} \tag{15}$$

The global minimum is 680.6300573, at $x^* = (2.330499,\ 1.951372,\ -0.4775414,\ 4.365726,\ -0.624487,\ 1.038131,\ 1.594227)$, with the first and last constraints active. For this problem, the following penalties are found: $\bar{\lambda}_1 = 68.5$, $\bar{\lambda}_2 = 26.0$, $\bar{\lambda}_3 = 5.2$, and $\bar{\lambda}_4 = 3.8$, after 52 to 620 analyses, using $s = 0.0001$. Table 1 presents results for Tests 1 and 2. The GBNM method was compared with the best evolutionary algorithm with population size = 50 (Test 1) and 20 (Test 2), mutation probability = 0.2, crossover probability = 0.4, and the same fixed penalty parameters $\bar{\lambda}_i$ as the GBNM method. Three different numbers of function calls were recorded: 500, 1000, and 2000. 100 runs were performed for each case, the first starting point for the GBNM method being randomly selected.

Table 1 Comparison of GBNM and EA on Tests 1 and 2 (average ± standard deviation, 100 runs)

		500 analyses		1000 analyses		2000 analyses	
		Minimum feasible function value	Probability of finding a feasible minimum	Minimum feasible function value	Probability of finding a feasible minimum	Minimum feasible function value	Probability of finding a feasible minimum
Test function 1	GBNM	−0.093768 ±0.011755	100/100	−0.095825 ±0.00000	100/100	−0.095825 ±0.00000	100/100
	EA	−0.059676 ±0.029717	92/100	−0.064979 ±0.032577	100/100	−0.087380 ±0.019187	100/100
Test function 2	GBNM	694.00 ±15.36	69/100	685.18 ±3.47	86/100	683.49 ±2.40	99/100
	EA	904.14 ±102.65	65/100	755.35 ±43.45	86/100	718.74 ±27.97	86/100

Table 2 Examples of possible local optima found by GBNM for Test 1: 2000 analyses, 1 run

(x_1, x_2)	function value	g_1	g_2
(1.22797,4.24537)(*)	−0.095825	−1.73746	−0.167762
(1.73414,4.74608)	−0.0291438	−0.738839	−0.177499
(1.32441,3.43043)	−0.0272629	−0.67637	-8.6956×10^{-10}
(1.67400,3.80228)	−0.0258123	-1.6623×10^{-9}	−0.634906
(0.0386021,4.22424)	−55.7608	−3.22275	1.01168(**)

(*) global optimum
(**) constraint not satisfied

As can be seen in Table 1, the GBNM method finds, on average, better objective function values, with a higher probability of finding a feasible minimum, than does the EA. The advantage of the GBNM method is substantial at a low number of analyses, and slowly decreases as the numerical cost grows. Note that the number of function evaluations shown in Table 1 does not take into account the procedure to obtain the stabilized parameters $\bar{\lambda}_i$. In fact, for Test 1, the majority of the resources were spent finding the penalty parameters $\bar{\lambda}_i$, which confirms observations made by Michalewicz and Schoenauer (1997) and Le Riche and Guyon (2001): finding the right penalty is difficult in Test 1. Test 1 is also a problem that has local optima. Many of them, reported in Table 2, have been characterized by GBNM as possible local optima (see the flow chart in Fig. 2) in a single run of 2000 analyses.

4.3 Composite laminated plates design

Composite laminates are made of stacked layers, in which each layer has oriented fibers melted in an isotropic matrix (see sketch in Fig. 4). The design problems addressed here aim at finding the optimal orientation of the fibers within each layer i, θ_i, where θ_i is a continuous variable bounded by 0 and 90°. The plates are analyzed using the classical lamination theory (see Berthelot 1999).

Longitudinal stiffness maximization, constrained by shear stiffness and Poisson's ratio

A 16-ply balanced and symmetric plate, made of glass-epoxy, is to be designed by maximizing the longitudinal stiffness E_x such that the shear stiffness G_{xy} is at least 12 GPa and Poisson's ratio ν_{xy} is not larger

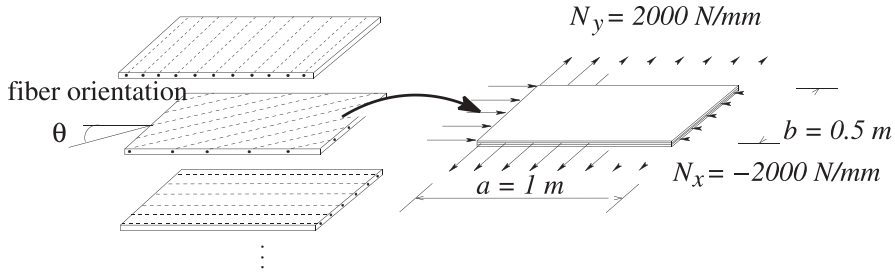

Fig. 4 Simply supported rectangular plate subjected to in plane loads

Table 3 Comparison of GBNM and EA: constrained E_x maximization, 100 runs

	100 analyses		200 analyses		500 analyses	
	Highest feasible E_x (GPa) avg. $\pm$ std.dev.	Probability of finding a feasible optimum	Highest feasible E_x (GPa) avg. $\pm$ std.dev.	Probability of finding a feasible optimum	Highest feasible E_x (GPa) avg. $\pm$ std.dev.	Probability of finding a feasible optimum
GBNM	14.4906 ± 0.0862	47/100	14.5302 ± 0.0059	97/100	14.5311 ± 0.0000	99/100
EA	13.9060 ± 0.3287	44/100	14.2502 ± 0.2090	56/100	14.4550 ± 0.0965	70/100

than 0.5. The elastic proprieties for the glass-epoxy layers are $E_1 = 45\,\text{GPa}$, $E_2 = 10\,\text{GPa}$, $G_{12} = 4.5\,\text{GPa}$, and $\nu_{12} = 0.31$. The plate is balanced and symmetric, so there are 4 fiber orientations to be found. For this problem, the averaged converged penalty parameters are $\bar{\lambda}_1 = 6.24$ and $\bar{\lambda}_2 = 1.18$, obtained after 44 to 435 analyses, with $s = 1.0$. The best performing evolutionary algorithm on this problem has a population size of 20, a mutation probability of 0.2, and a crossover probability of 0.4.

Table 3 compares the algorithms after 100, 200, and 500 analyses based on 100 independent runs. The best designs found are any permutation of the stacking sequence $[\pm 36.6/\pm 43.1/\pm 50.1/\pm 54.9]_s$. They have $E_x = 14.54\,\text{GPa}$ and the two constraints are active.

Other quasi-feasible stacking sequences create local convergence of the simplexes and are tagged possible local optima by GBNM (see Fig. 2). Some of them, which have a difference of less than 0.2% in the E_x best value, are listed in Table 4.

Buckling load maximization, constrained by Hoffman failure criterion and thermal expansion

Let us consider a rectangular simply supported carbon-epoxy plate, subjected to an in-plane load $N_x = -2000\,\text{N/mm}$ and $N_y = 2000\,\text{N/mm}$, as shown in Fig. 4. The plate is balanced and symmetric and has 48 layers, each of which are 0.125-mm thick. The elastic material properties of the layers are $E_1 = 115\,\text{GPa}$, $E_2 = 5\,\text{GPa}$, $G_{12} = 5\,\text{GPa}$, and $\nu_{12} = 0.35$. The coefficients of thermal expansion are $\alpha_1 = -0.5 \times 10^{-6}\,/^\circ\text{C}$ and $\alpha_2 = 20 \times 10^{-6}\,/^\circ\text{C}$. The ultimate strengths in the longitudinal (X_t and X_c), transversal (Y_t and Y_c), and shear (S) directions are $X_t = 964\,\text{MPa}$ and $X_c = 895\,\text{MPa}$, $Y_t = 50\,\text{MPa}$ and $Y_c = 100\,\text{MPa}$, and $S = 94\,\text{MPa}$. The laminate is designed to maximize its buckling safety factor, f_{buckl}, while not failing as predicted by the Hoffman criterion and while having magnitudes of the thermal expansion coefficients, $|\alpha_x|$ and $|\alpha_y|$, below $1 \times 10^{-6}\,/^\circ\text{C}$. An elastic linear buckling model is used (Berthelot 1999). Since the plate is balanced and symmetric, there are 12 continuous design variables, the ply orientations, which are bounded between 0 and 90°. The problem is of medium size.

The averaged converged penalty parameters are $\bar{\lambda}_1 = 0.0062$, $\bar{\lambda}_2 = 0.0598$, and $\bar{\lambda}_3 = 0.4048$, and convergence in the dual space occurs after 116 to 978 analyses with $s = 0.001$. The best evolutionary algorithm tested has a population size of 50, a mutation probability of 0.15, and a crossover probability of 0.5. Table 5 sum-

Table 4 Examples of near-optimal designs found by GBNM for the E_x maximization problem: 2000 analyses, 1 run

stacking sequence	E_x (GPa)	G_{xy} (GPa)	ν_{xy}
$[\pm 41.7/\pm 57.7/\pm 46.2/\pm 39.5]_s$	14.53	12.00	0.50
$[\pm 43.0/\pm 45.2/\pm 39.1/\pm 57.8]_s$	14.53	12.00	0.50
$[\pm 48.5/\pm 34.9/\pm 47.4/\pm 53.9]_s$	14.53	12.00	0.50
$[\pm 49.7/\pm 45.4/\pm 35.4/\pm 54.2]_s$	14.53	12.00	0.50
$[\pm 51.9/\pm 52.1/\pm 45.5/\pm 35.1]_s$	14.53	12.00	0.50
$[\pm 57.7/\pm 38.8/\pm 45.0/\pm 43.6]_s$	14.52	12.00	0.50

Table 5 Buckling load maximization: 100 runs

	200 analyses		500 analyses		1000 analyses	
	Highest feasible buckling safety factor f_{buckl} avg. $\pm$ std.dev.	Probability of finding a feasible optimum	Highest feasible buckling safety factor f_{buckl} avg. $\pm$ std.dev.	Probability of finding a feasible optimum	Highest feasible buckling safety factor f_{buckl} avg. $\pm$ std.dev.	Probability of finding a feasible optimum
GBNM	1.4260 ± 0.0513	78/100	1.4883 ± 0.0145	99/100	1.4959 ± 0.0125	100/100
EA	1.4557 ± 0.0134	69/100	1.4806 ± 0.0073	83/100	1.4919 ± 0.0035	94/100

marizes the comparison of the methods at 200, 500, and 1000 analyses based on 100 independents runs. The best solution found is $[\pm 27.5/\pm 27.5/\pm 28.0/\pm 28.7/\pm 29.7/\pm 30.7/\pm 32.2/\pm 35.8/\pm 43.9/\pm 70.7/\pm 89.9/\pm 89.9]_s$, where $f_{buckl} = 1.5002$, $\alpha_x = 3.04 \times 10^{-7}$ /°C, $\alpha_y = 1.00 \times 10^{-6}$ /°C, and the Hoffman failure load factor is equal to 1.286.

The results presented in Tables 3 and 5 corroborate the tests performed on analytical functions: the GBNM method is faster and more robust at finding feasible minima than the evolutionary algorithm on the functions studied. The advantage, which is important below 500 analyses, shrinks progressively as more analyses are performed.

5 Concluding remarks

A local/global optimization method based on probabilistic restarts has been presented. Local searches are performed by an improved Nelder–Mead algorithm with which design variables can be bounded, inequality constraints taken into account, and some search failure cases prevented. The method, called the Globalized Bounded Nelder–Mead search, does not need sensitivities and constructively uses computer resources up to a given limit. It yields a list of candidate local optima, which contain global solutions with an increasing probability with increasing computer time.

The GBNM method is simple in principle and the aforementioned features make it particularly useful in an engineering design context.

Acknowledgements The first author would like to express his thanks to the Federal Center for Technological Education of Paraná (CEFET-PR), Brazil, and to the Brazilian funding agency CNPq for financial support during this research.

References

Bäck, T. 1996: *Evolutionary Algorithms in Theory and Practice*. Oxford: Oxford University Press

Barhen, J.; Protopopescu, V.; Reister, D. 1997: TRUST: a deterministic algorithm for global constrained optimization. *Science* **276**, 1094–1097

Berthelot, J.-M. 1999: *Composite Materials: Mechanical Behavior and Structural Analysis*, Mechanical Engineering Series. Berlin: Springer

Duda, O.R.; Hart, P.E.; Stork, D.G. 2001: *Pattern Classification*, 2nd edn. New York: John Wiley & Sons

Durand, N.; Alliot, J.-M. 1999: A combined Nelder–Mead simplex and genetic algorithm. Available at: http://www.recherche.enac.fr/opti/papers/

Goldberg, D.E.; Voessner, S. 1999: Optimizing global-local search hybrids. In: *GECCO 99 – Genetic and Evolutionary Computation Conference* (held in Orlando), pp. 220–228

Haftka, R.T.; Gürdal, Z. 1993: *Elements of Structural Optimization*, 3rd edn. Boston: Kluwer Academic Publishers

Hickernell, F.J.; Yuan, Y.-X. 1997: A simple multistart algorithm for global optimization. *OR Trans.* **1**(2), 1–11

Hu, X.; Shonkwiller, R.; Spruill, M.C. 1994: *Random Restarts in Global Optimization*. Technical Report, School of Mathematics, Georgia Institute of Technology, Atlanta

Le Riche, R.; Guyon, F. 2001: Dual evolutionary optimization. In: Collet, P.; Lutton, E.; Schoenauer, M.; Fonlupt, C.; Hao, J.-K. (eds.) *Artificial Evolution, Lecture Notes in Computer Science, No. 2310, selected papers of the 5th International Conference on Artificial Evolution* (held in Le Creusot), pp. 281–294

Luersen, M.A.; Le Riche, R. 2001: *Globalisation de l'Algorithme de Nelder–Mead : Application aux Composites*. Technical Report, LMR, INSA de Rouen, France; in French

Michalewicz, Z.; Schoenauer, M. 1997: Evolutionary algorithms for constrained parameter optimization. *Evolut. Comput.* **4**(1), 1–32

Minoux, M. 1986: *Mathematical Programming: Theory and Algorithms*. New York: John Wiley & Sons

Moscato, P. 1989: *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms*. Caltech Concurrent Computation Program, C3P Report 826

Nelder, J.A.; Mead, R. 1965: A simplex for function minimization. *Comput. J.* **7**, 308–313

Okamoto, M.; Nonaka, T.; Ochiai, S.; Tominaga, D. 1998: Nonlinear numerical optimization with use of a hybrid genetic algorithm incorporating the modified Powell method. *Appl. Math. Comput.* **91**, 63–72

Rockafellar, R.T. 1976: Lagrange multipliers in optimization. In: Cottle R.W.; Lemke C.E. (eds.) *Nonlinear Programming, Proc. SIAM-AMS*, **9**, 145–168

Shang, Y.; Wan, Y.; Fromherz, M.P.J.; Crawford, L. 2001: Toward adaptive cooperation between global and local solvers for continuous constraint problems. In: *CP'01 Workshop on Cooperative Solvers in Constraints Programming* (held in Pahos)

Syswerda, G. 1991: A study of reproduction in generational and steady state genetic algorithms. In: Rawlins, G.J.E. (ed.) *Foundations of Genetic Algorithms*. San Mateo: Morgan Kaufmann

Törn, A.A. 1978: A search-glustering approach to global optimization. In: *Towards Global Optimization 2*, pp. 49–62

Törn, A.A.; Zilinskas A. 1989: *Global Optimization*. Berlin: Springer-Verlag

Wright, M.H. 1996: Direct search methods: once scorned, now respectable. In: *Dundee Biennial Conference in Numerical Analysis* (held in Harlow), pp. 191–208

Appendix A:
A Nelder–Mead algorithm with bounded variables

The Nelder–Mead method (Nelder and Mead 1965) is the most popular direct search method for minimizing unconstrained real functions. It is based on the comparison of function values at the $n+1$ vertices x_i of a simplex. A simplex of size a is initialized at x_0 based on the rule (see Haftka and Gürdal 1993)

$$x_i = x_0 + pe_i + \sum_{\substack{k=1 \\ k\neq i}}^{n} qe_k \ , \quad i = 1, n \ , \tag{A.1}$$

where e_i are the unit base vectors and

$$p = \frac{a}{n\sqrt{2}} \left(\sqrt{n+1} + n - 1\right) \ ,$$

$$q = \frac{a}{n\sqrt{2}} \left(\sqrt{n+1} - 1\right) \ . \tag{A.2}$$

The simplex vertices are changed through *reflection*, *expansion*, and *contraction* operations in order to find an improving point (see Fig. 5). The algorithm terminates when the vertices function values become similar, which is measured with the inequality,

$$\sqrt{\sum_{i=1}^{n+1} (f_i - \bar{f})^2 / n} < \varepsilon \ , \ \bar{f} = \frac{1}{n+1} \sum_{i=1}^{n+1} f_i \ , \tag{A.3}$$

where ε is a small positive scalar. The cumulative effect of the operations on the simplex is, roughly speaking, to stretch the shape along the descent directions, and to zoom around local optima. Two comments on the properties of the algorithm are added. Firstly, the Nelder–Mead al-

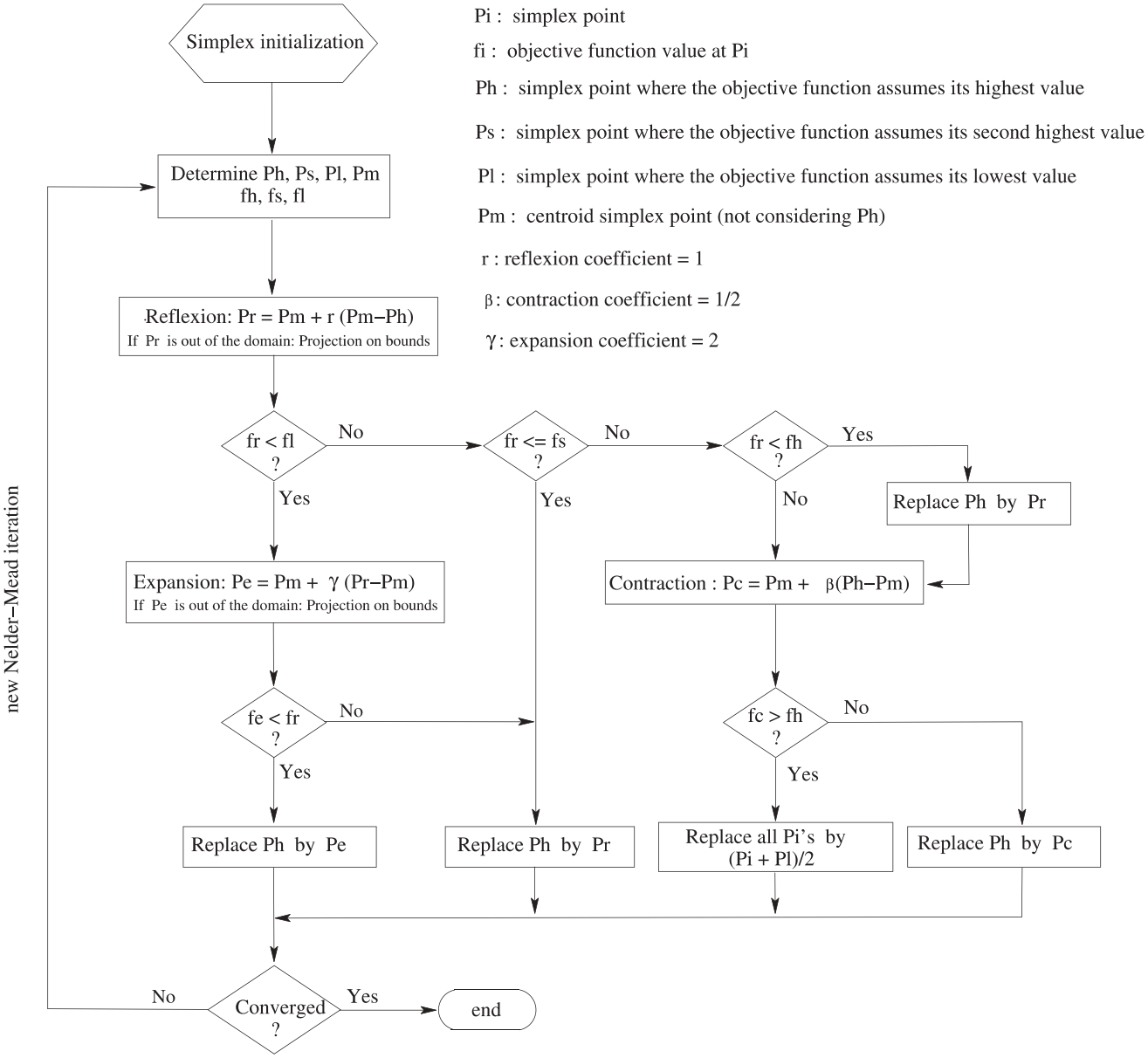

Fig. 5 Nelder–Mead algorithm with bounded variables

gorithm may fail to converge to a local optimum, which happens in particular when the simplex collapses into a subspace. Secondly, the method may escape a region that would be a basin of attraction for a pointwise descent search if the simplex were large enough. Ultimately, as the size of the simplex decreases, the algorithm becomes local.

The original Nelder–Mead algorithm was conceived for unbounded domain problems. With bounded variables, the points can leave the domain after either the reflection or the expansion operation. It is straightforward to account for variable bounds by projection:

$$\begin{cases} \text{if } (x_i < x_i^{min}),\ x_i = x_i^{min}\ , \\ \text{if } (x_i > x_i^{max}),\ x_i = x_i^{max}\ . \end{cases} \tag{A.4}$$

The flowchart of the Nelder–Mead method shown in Fig. 5 differs from the original method only in the initialization (see (A.1)) and in the bounded variables. An important side effect of accounting for the bounded variables through projection is that it tends to make the simplex collapse into the subspace of the saturated variables. A specific convergence test, based on a small simplex reinitialization at the point of convergence, is then required (see "small test" in Sect. 2.2).

Appendix B: Adaptive linear penalty as generalized Lagrangian

The adaptive linear penalty scheme of (11) and (12) can be analyzed using generalized Lagrangian theory (see Rockafellar 1976; Minoux 1986). In the following, operators $<$, $\leq$, $>$, and $\geq$ used with vectors apply to all components independently.

Proposition 1 ($\bar{L}$ generalized Lagrangian). *Let $\bar{L}$ be defined as*

$$\begin{cases} \bar{L}(x,\lambda) = f(x) + \lambda.g^+(x) \quad \textit{if } \lambda \geq 0\ , \\ -\infty \quad \textit{if } \exists\ k\ ,\ 1 \leq k \leq m\ /\ \lambda_k < 0\ , \end{cases} \tag{B.1}$$

where $g^+(x) = \max(0, g(x))$. $\bar{L}$ is a generalized Lagrangian.

This ensues directly from the definition of generalized Lagrangians : $\bar{L}$ is a closed concave function of λ, and, introducing the essential objective function $\bar{f}$, for which

$$\begin{cases} \bar{f}(x) = f(x) \quad \text{if } x \text{ feasible}\ , \\ +\infty \quad \text{otherwise}\ , \end{cases} \tag{B.2}$$

one readily shows that $\bar{f}(x) = \max\limits_{\lambda \in \Re^m} \bar{L}(x,\lambda)$, which completes the definition.

Denoting by x^* a solution of the primal problem (P), a given problem, defined by f and g, has a saddle point with respect to $\bar{L}$ if

$$f(x^*) = \min_x \bar{f}(x) = \min_x \max_\lambda \bar{L}(x,\lambda) =$$

$$\max_\lambda \min_x \bar{L}(x,\lambda) = \max_\lambda \bar{\varphi}(\lambda)\ , \tag{B.3}$$

where

$$\bar{\varphi}(\lambda) = \min_x \bar{L}(x,\lambda) \tag{B.4}$$

is the dual function associated with $\bar{L}$. As (B.3) shows, if a problem has a saddle point with respect to the generalized Lagrangian $\bar{L}$, there is a well-founded way to build a penalty function that leads to x^*: the primal constrained formulation (P) can be replaced equally by the dual problem

$$(\bar{D}) \qquad \max_{\lambda \in \Re^m} \bar{\varphi}(\lambda)\ . \tag{B.5}$$

If a problem has a saddle point and $\bar{\lambda}$ is a solution of $(\bar{D})$, x^* is obtained by minimizing the penalty function $\bar{L}(x,\bar{\lambda})$. In terms of λ, $(\bar{D})$ is easy to solve because $\bar{\varphi}(\lambda)$ is a concave function (see proof in Minoux (1986)). Moreover, there is a convenient expression for its sub-gradient.

Proposition 2 (Sub-gradient of the dual function). *A sub-gradient of the (generalized) dual function $\bar{\varphi}$ at λ is $g^+(\bar{x}(\lambda))$, where $\bar{x}(\lambda)$ is such that $\bar{L}(\bar{x}(\lambda),\lambda) = \min\limits_x \bar{L}(x,\lambda)$.*

The proof is the direct application of Theorem 4′ in Chapter 6 of Minoux (1986). Proposition 2 shows that calculating a sub-gradient of $\bar{\varphi}$ is a by-product of calculating φ itself, since both involve minimizing $\bar{L}(,\lambda)$. Such minimization remains the most costly operation when solving $(\bar{D})$. Therefore, in this work, the dual function and its sub-gradient are not calculated by the complete resolution of (B.4) but are approximated by the heuristic of (12). The condition $\bar{L}(x^{new},\lambda^k) \leq \bar{L}(x^{best},\lambda^k)$ and the update of x^{best} guarantee that the approximation of $\bar{x}(\lambda)$ improves at each iteration. In light of the definition of $\bar{\varphi}$'s sub-gradient, the updating formula of λ in (12) is interpreted as a fixed step gradient algorithm that solves $(\bar{D})$, where $g(\bar{x}(\lambda))$ is approximated by $g(x^{best})$.

All the above considerations assume that the problem has a saddle point. Perturbation analysis (Rockafellar 1976) helps to characterize the existence of saddle points. For a short while, let $\bar{L}$ denote any generalized Lagrangian. The ordinary Lagrangian, $L(x,\lambda) = f(x) + \lambda g(x)$, is an instance of $\bar{L}$. Perturbation analysis uses the perturbational representation, $\bar{F}$, and the optimal value function, $\bar{\phi}$. Let Y be a real linear space and y be a vector in Y. The perturbational representation is defined as

$$\bar{F}(x,y) = \max_{\lambda \in \Re^m} (\bar{L}(x,\lambda) - \lambda y) \tag{B.6}$$

and the optimal value function is

$$\bar{\phi}(y) = \min_x \bar{F}(x,y)\ . \tag{B.7}$$

The following theorem, from Minoux (1986), states that if there is a supporting hyperplane of $\bar{\phi}$ at $y = 0$, then solving $(\bar{D})$ also solves (P).

Theorem 1. *A necessary and sufficient condition for* $\bar{\lambda}$ *to be a saddle point multiplier is that*

$$\forall y \in Y\ , \qquad \bar{\phi}(y)\ \geq\ \bar{\phi}(0) - \bar{\lambda} y\ . \tag{B.8}$$

Using (B.7), the optimal value function of the ordinary Lagrangian is

$$\phi(y) = \min_{x\ /\ g(x) \leq y} f(x)\ . \tag{B.9}$$

From now on, $\bar{L}$ denotes the only generalized Lagrangian discussed here, the linear adaptive penalty function of (B.1). It can be shown that the associated optimal value function is

$$\bar{\phi}(y) = \begin{cases} +\infty & \text{if } \exists\, k\ ,\ 1 \leq k \leq m\ /\ y_k\ <\ 0\ , \\ \phi(y) & \text{if } y\ \geq\ 0\ . \end{cases} \tag{B.10}$$

Figure 6 shows the optimal value functions ϕ and $\bar{\phi}$ on two example problems. On the left, the problem does not have a saddle point in terms of the ordinary Lagrangian, but has saddle points in terms of the linear adaptive penalty function. On the right, L and $\bar{L}$ both have saddle points. Because $\bar{\phi}$ is equal to ϕ in the subspace of Y such that $y\ \geq\ 0$ and is infinite elsewhere, it has supporting hyperplanes at $y = 0$ more frequently than does ϕ.

Proposition 3 (Saddle points of $\bar{L}$).
The class of problems for which $\bar{L}$ *has a saddle point contains the class of problems for which ordinary Lagrangians have a saddle point.*

Two other features of the penalty function $\bar{L}$ are visible in Figure 6 and are presented without proof.

Proposition 4 (Set of $\bar{\lambda}$'s).
The set of saddle point multipliers of $\bar{L}$, $\{\bar{\lambda}\}$, *is bounded below. Furthermore, if* λ^* *is a saddle point multiplier of the ordinary Lagrangian* L, *it is the lower bound of* $\{\bar{\lambda}\}$, *i.e.,* $\bar{\lambda}\ \geq\ \lambda^*$.

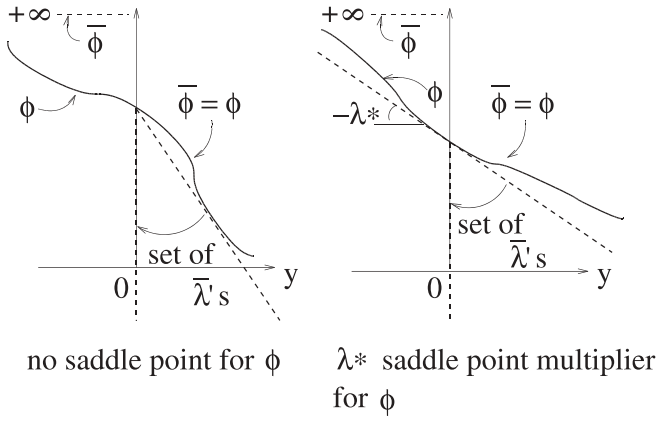

Fig. 6 Two examples of optimal value functions ϕ and $\bar{\phi}$ for a single constraint. λ^* and $\bar{\lambda}$ are saddle point multipliers for L and $\bar{L}$, respectively. On the left, unlike $\bar{\phi}$, ϕ has no supporting hyperplane at $y = 0$ (i.e., no saddle point). A problem with a saddle point for the ordinary Lagrangian is sketched on the right: ϕ and $\bar{\phi}$ both have supporting hyperplanes

In particular, it should be stressed that the lower bound of $\{\bar{\lambda}\}$ is often finite.

Proposition 5 (Finite penalty parameters).
x^* *can be solution of* $\min_x \bar{L}(x, \bar{\lambda})$ *for finite values of* $\bar{\lambda}$.

Quadratic and higher order exterior penalty functions achieve differentiability at $g(x) = 0$ (providing f and g are differentiable) but need, in all cases, infinitely large penalty parameters in order to approach the feasible domain. On the contrary, $\bar{L}$ is not differentiable but convergence to the optimum x^* can occur for finite penalties.