Int. J. Appl. Math. Comput. Sci., 2010, Vol. 20, No. 4, 679–688
DOI: 10.2478/v10006-010-0051-1

amcs

DERIVATIVE–FREE NONLINEAR OPTIMIZATION FILTER SIMPLEX

ALDINA CORREIA [*,**], JOÃO MATIAS [**], PEDRO MESTRE [***], CARLOS SERODIO [***]

[*] ESTGF-IPP, School of Technology and Management of Felgueiras
Polytechnic Institute of Porto, 4610–156 Felgueiras, Portugal
e-mail: aic@estgf.ipp.pt

[**]CM-UTAD, Centre for Mathematics
University of Trás-os-Montes and Alto Douro, 5000–911 Vila Real, Portugal
e-mail: j_matias@utad.pt

[***]CITAB, Centre for the Research and Technology of Agro-Environment and Biological Sciences
University of Trás-os-Montes and Alto Douro, 5000–911 Vila Real, Portugal
e-mail: {pmestre,cserodio}@utd.pt

The filter method is a technique for solving nonlinear programming problems. The filter algorithm has two phases in each iteration. The first one reduces a measure of infeasibility, while in the second the objective function value is reduced. In real optimization problems, usually the objective function is not differentiable or its derivatives are unknown. In these cases it becomes essential to use optimization methods where the calculation of the derivatives or the verification of their existence is not necessary: direct search methods or derivative-free methods are examples of such techniques. In this work we present a new direct search method, based on simplex methods, for general constrained optimization that combines the features of simplex and filter methods. This method neither computes nor approximates derivatives, penalty constants or Lagrange multipliers.

Keywords: nonlinear constrained optimization, filter methods, direct search methods.

1. Introduction

Let us consider the following Constrained Nonlinear Programming Problem (NLP):

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1a}$$

$$\text{subject to } C(x) \leq 0, \tag{1b}$$

where

- $f : \mathbb{R}^n \to \mathbb{R}$ (1a) is the *objective function*;
- $c_i : \mathbb{R}^n \to \mathbb{R},\ i = 1, 2, \ldots, m$, (1b) are *constraints*, $C(x) = (c_1(x), c_2(x), \ldots, c_m(x))^T$;
- $x = (x_1, x_2, \ldots, x_n)^T,\ x \in \mathbb{R}^n$;
- $\Omega = \{x \in \mathbb{R}^n : c_i(x) \leq 0, i = 1, 2, \ldots, m\}$ is the *feasible region*.

Solving this problem involves two objectives and two concepts: minimize the objective function (*Optimality*) and minimize the constraint violation, which must be zero or tend to zero (*Viability*). The methods presented here are specially dedicated to solve problems for which we cannot compute or directly estimate derivatives of the objective and/or constraint functions, because they can be black box or nonsmooth functions and it is not possible to calculate their derivatives.

There are several methodologies to solve a constrained nonlinear optimization problem (1). The most popular approaches are penalty or barrier methods. The basic idea of these methods is the same: construct a sequence of unconstrained problems, solve them (using unconstrained optimization techniques) and find the solution of the original constrained problem. In these methods, optimality and feasibility are treated together.

Penalty methods consist in adding to the objective function a measure of infeasibility, multiplied by a penalty parameter, penalizing the function value of points that are outside the feasible region. These methods allow infeasible iterates and may start with an infeasible point (Byrd

et al., 2008; Bertsekas, 1999).

Barrier methods consist in adding to the objective function a function multiplied by a barrier parameter. This function approaches $+\infty$ when the feasible iterates move towards the frontier of the feasible region. These methods cannot start with an infeasible point but ensure the feasibility of their iterates. This approach is often used by Audet and colleagues, in the context of optimization without derivatives (Audet and Dennis Jr., 2002; 2004; 2006; 2007; Audet *et al.*, 2008).

Another alternative to solve the NLP problem (1) is to use the filter method. Unlike previous methods, this one considers feasibility and optimality separately, by using the concept of the dominance of multiobjective optimization. The filter method was introduced by Fletcher and Leyffer (2002). A filter algorithm introduces a function that aggregates constraint violations and constructs a bi-objective problem which attempts to minimize simultaneously this function (*feasibility*) and the objective function (*optimality*), giving priority to feasibility. See the work Fletcher *et al*. (2006) for a survey.

The first filter method for derivative-free nonlinear programming, based on pattern search methods, was presented by Audet and Dennis Jr. (2004). Based on this work we have developed a new direct search method, based on simplex methods, for general constrained optimization, which combines the features of the simplex and filter methods (Correia *et al*., 2009).

In this work we present several modifications of the Simplex Filter Algorithm (SFA), presented by Correia *et al*. (2009), which improves its performance. Then we compare the behaviour of this new method with that of our initial simplex filter algorithm.

2. Simplex methods

Simplex methods are derivative-free unconstrained nonlinear optimization methods characterized by changing the search directions at each iteration, building a nondegenerate simplex in $\mathbb{R}^n$ and using it to drive the search.

Definition 1. A *simplex* is a set of noncollinear $n+1$ points in $\mathbb{R}^n$:

$$\{x_i\}_{i=1}^{n+1}.$$

The point x_i is the i-th vertex of the simplex.

Thus, in $\mathbb{R}^2$, a simplex is a triangle, in $\mathbb{R}^3$, a tetrahedron, etc. Let us consider an unconstrained nonlinear problem:

$$\min_{x\in\mathbb{R}^n} f(x), \tag{2}$$

an initial point x_0, a step length s and a basis in $\mathbb{R}^n$ (e.g. the canonic basis in $\mathbb{R}^n$ $\{e_i\}_{i=1}^n$).

The most popular simplex method is the that of Nelder and Mead (1965). This method begins with constructing a simplex with edges of equal length, i.e., a set of noncollinear $n+1$ points in $\mathbb{R}^n$, such that

$$v_0 = x_0, \quad v_i = x_0 + s\, e_i, \quad i = 1, \ldots, n.$$

The corresponding objective function values are calculated as

$$f_i = f(v_i), \quad i = 0, \ldots, n.$$

The *worst* vertex in the simplex (which is the least desirable objective value) is identified and then is replaced using four basic operations: reflection; expansion; contraction and shrinkage. Thus, it is necessary to identify the vertices that have extreme function values.

Definition 2.

- The *worst* vertex is the vertex v_w such that

$$f(v_w) = f_w = \max_{0\le i\le n} \{f_i\}.$$

- The *better* vertex is the vertex v_b such that

$$f(v_b) = f_b = \min_{0\le i\le n} \{f_i\}.$$

- The *second worst* vertex is the vertex v_{sw} such that

$$f(x_{sw}) = f_{sw} = \max_{\substack{0\le i\le n\\ i\ne w}} \{f_i\}.$$

In order to determine a new vertex to replace the worst vertex, we can use the centroid of the best n vertices.

Definition 3. Consider the simplex $\{v_i\}_{i=0}^n$ in $\mathbb{R}^n$, with the worst vertex v_w, $w \in \{0, 1, \ldots, n\}$. The *centroid* of the best n vertices is

$$v_c = \frac{1}{n}\sum_{\substack{i=0\\ i\ne w}}^{n} v_i.$$

Definition 4. With parameter values α, β and γ, which must satisfy the conditions $0 < \beta < \alpha \le 1$ and $\gamma > 1$, the auxiliary points corresponding to the four basic operations are as follows:

- *Reflected* vertex:

$$v_r = v_c + \alpha(v_c - v_w),$$

where α is the reflection parameter (usually $\alpha = 1$).

- *Expanded* vertex:

$$v_e = v_c + \beta(v_r - v_c),$$

where β is the expansion parameter (usually $\beta = 2$).

- *Contracted* vertex to the outside:

$$v_{oc} = v_c + \gamma(v_r - v_c),$$

where γ is the contraction parameter (usually $\gamma = 1/2$).

- *Contracted* vertex to the inside:

$$v_{ic} = v_c + \gamma(v_w - v_c),$$

where γ is the contraction parameter (usually $\gamma = 1/2$).

Reflection is the first operation to be executed. If needed, the next operation is expansion, which can be followed by contraction. The expansion step allows a more aggressive move by doubling the length of the step from the centroid to the reflection point. The contraction step allows more conservative moves by halving the length of the step from the centroid to either the reflection point or the worst vertex.

When no step produces a significant improvement in the objective function value, the *shrink* step is used:

$$v_i' = \frac{v_i + v_b}{2}, \quad i = 1, 2, \ldots, n+1, \quad i \neq b.$$

This step consists in reducing the lengths of the edges adjacent to the current best vertex by half.

Whatever the final simplex, the corresponding f values are calculated proceeding again to the identification of the vertices v_b, v_{sw} and v_w in order to start a new iteration.

The Nelder–Mead simplex algorithm is, of all the direct search methods, most often found in numerical software packages because it is efficient and has a relatively easy implementation.

3. Filter method

The filter method treats optimization problems as bi-objective ones attempting to minimize the objective function and a continuous function h that aggregates the constraint violation functions. The priority must be given to h, at least until a feasible iterate is found, where h is a nonnegative function where $h(x) = 0$ if and only if x is feasible.

Thus, we define h as $h(x) = \|C_+(x)\|$, where $\|\cdot\|$ is a vector norm and $C_+(x)$ is the vector of m constraint values:

$$C_+(x) = \begin{cases} c_i(x) & \text{if} \quad c_i(x) > 0, \\ 0 & \text{if} \quad c_i(x) \leq 0. \end{cases}$$

For example, in the case of the quadratic norm, we have

$$h(x) = \|C_+(x)\|_2 = \sqrt{\sum_{i=1}^{m} \max(0, c_i(x))^2}.$$

The filter method uses the concept of dominance from multiobjective optimization. Defining a *forbidden region* and storing in a set (filter) points $(x_k, f(x_k), h(x_k))$ (with good performance in the previous iterations), *dominated* points (in the sense of the Pareto rule) are avoided at subsequent iterations.

Definition 5. (Karas *et al*., 2006) A point $x \in \mathbb{R}^n$ is said to *dominate* $y \in \mathbb{R}^n$, which is written as $x \prec y$, if

- $f(x) \leq f(y)$ and $h(x) \leq h(y)$, or
- $f(x) < f(y)$ or $h(x) < h(y)$.

Definition 6. (Karas *et al*., 2006) A *filter*, denoted by $\mathcal{F}$, is a finite set of points in the domain of f and h such that no pair of points x and y in the set satisfies the relation $x \prec y$.

Thus, a point is accepted in the filter if and only if it is not dominated by any other point in the filter and its inclusion in the filter eliminates all points that it dominates, i.e., the filter is a dynamic set and it works as an acceptance criterion for the iteration.

The Audet and Dennis filter (Audet and Dennis Jr., 2004) differs from the usual filters in three aspects, according to Fletcher *et al*. (2006):

- It only requires a simple decrease.
- It sets a bound $h_{\max}$ on aggregate constraint violation, so that each point $x \in \mathcal{F}$ satisfies $h(x) < h_{\max}$.
- It includes only infeasible points in the filter and tracks feasible points separately.

Definition 7. (Audet and Dennis Jr., 2004) A point x is said to be *filtered* by a filter $\mathcal{F}$ if any of the following properties is satisfied:

- There exists a point $y \in \mathcal{F}$ such that $y \prec x$ or $y = x$.
- $h(x) \geq h_{\max}$.
- $h(x) = 0$ and $f(x) \geq f^F$, where f^F is the objective function value of the best feasible point found.

Definition 8. (Audet and Dennis Jr., 2004) The point x is said to be *unfiltered* by $\mathcal{F}$ if it is not filtered by $\mathcal{F}$.

Definition 9. (Audet and Dennis Jr., 2004) Consider $x \in \mathcal{F}$. The *set of unfiltered points* $y \in \mathbb{R}^n$, denoted by $\bar{\mathcal{F}}$, is given by

$$\{y : y \prec x \text{ or } y = x\} \cup \left\{y : h(y) = 0, f(y) \geq f^F\right\}.$$

4. Filter simplex algorithm

In this section we compare the behaviour of this new method with our initial filter simplex (Correia *et al.*, 2009). They both use the Nelder–Mead simplex method for internal iterations and neither compute nor estimate any derivatives.

The main characteristics of the SFA (Simplex Filter Algorithm), presented by Correia *et al.* (2009), are as follows:

- It builds an initial simplex from an initial point.
- Each point of the simplex is tested:
  - If a point is feasible, its inclusion in the filter is evaluated.
  - If it is infeasible, only three of the basic operations of the Nelder–Mead method (reflection, contraction and expansion) are applied and only then its inclusion in the filter is evaluated.
- If the point is accepted in the filter, the process continues from the beginning using this point, as long as the stop criterion is not satisfied.
- If none of the simplex points is accepted in the filter, the Shrink step is applied and the process continues from the beginning using the best point, as long as the stop criterion is not satisfied.
- The solution is a feasible point, chosen from the final filter, with the best objective function value.

In this work, a new algorithm is presented, the NSFA (New Simplex Filter Algorithm). It differs from the SFA in two key points:

- It applies the four basic operations of the Nelder–Mead method to h in the simplex search so as to obtain *feasibility*.
- It implements the Nelder–Mead method to f in the filter step, with the aim of *optimality*.

Details on the implementation of the new algorithm are presented in Section 4.1 and schematically depicted in Fig. 1.

Moreover, while the initial algorithm (SFA) is only allowed to find the best feasible solution x_{kf}, with this new algorithm (NSFA) it is possible to find the best feasible solution x_{kf}, and still the best unfeasible solution x_{ki}. This second solution may be important for relaxable problems, where it is possible to accept a solution with the constraint violation value lower than $h_{\max}$.

4.1. New simplex filter algorithm. The procedure used to implement the NFSA is presented in Fig. 1.

It begins with an initial filter that contains the initial iteration, $F_0 = x_0$. Then, we construct an initial simplex (S_k) containing $n+1$ vertices from that iteration (x_k) and

$$S_k = \{x_k = v_0\} \cup \{v_i : v_i = x_k + e_i, i = 1, \ldots, n\},$$

where e_i, $i = 1, \ldots, n$ represent the vectors of the canonic basis in $\mathbb{R}^n$, starting with the simplex search for $i = 0, \ldots, n$.

If the vertex being analysed is a feasible point, its inclusion in the filter is evaluated:

- If it is not accepted:
  - The Nelder–Mead method is applied to the function f.
  - A new point is obtained, x_k.
  - Go back to the construction of the simplex:

    $$S_k = \{x_k\} \cup \{v_i : v_i = x_k + e_i, i = 1, \ldots, n\},$$

    using x_k.
- If it is accepted:
  - The filter is updated with the new approximation to the solution, i.e., the new iteration.
  - If the stop criterion is verified, this approximation is the solution. Otherwise, go back to the simplex construction, using this point.

If the vertex is an infeasible point, its inclusion in the filter is evaluated:

- If it is not accepted:
  - The Nelder–Mead method is applied to the function h.
  - A new point is obtained, x_k.
  - Go back to the construction of the simplex:

    $$S_k = \{x_k\} \cup \{v_i : v_i = x_k + e_i, i = 1, \ldots, n\},$$

    using x_k.
- If it is accepted:
  - The filter is updated with the new approximation to the solution, i.e., the new iteration.
  - If the stop criterion is verified, this approximation is the solution. Otherwise, go back to the simplex construction, using this point.

Thus, the method contains two distinct processes:

- The *external iterative process*, involving the simplex construction and the filter update.

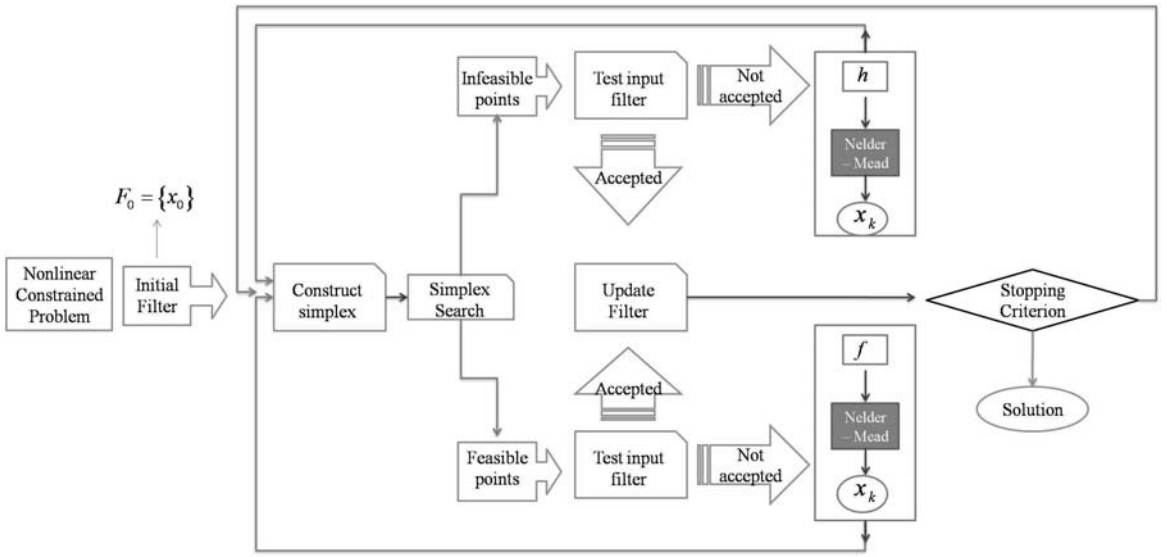

Fig. 1. Implemented algorithm.

- The *internal iterative process*, involving the optimization of f and h, where unconstrained optimization problems are solved, with objective functions f or h, using the Nelder–Mead simplex method.

Besides the above mentioned characteristics, the NSFA also has a stronger component dedicated to feasibility, when compared with with SFA. It includes an unconstrained optimization process for h, when the simplex point being analysed, in the simplex search, is infeasible. For feasible simplex points, it includes an unconstrained optimization process for f.

Furthermore, the SFA only optimizes f (and not h) for infeasible points. And the SFA, for feasible points, only verifies its admittance to the filter. If accepted, the point is added to the filter, otherwise the shrink step is applied. No optimization of h is then made in the SFA, unlike the NSFA.

5. Numerical results

The algorithms of our method were implemented in the Java language. To test the performance of our method, we considered three problems reported in the work of Correia *et al*. (2009). Neither, the SFA nor NSFA computes or estimates derivatives because they use the Nelder–Mead simplex method for internal iterations.

5.1. Problems. The first problem was presented by Lewis and Torczon and used by Audet and Dennis Jr. (2004) to illustrate the performance of their method. The second and third problems were selected in the Cute collection (Bongartz *et al*., 1995).

We consider $k_{\max} = 40$ in the external iterative process and $k_{\max} = 40$ in the internal process. The tolerances T_1 and T_2 ($T_1 = T_2 < 0.0001$) were chosen in the external iterative process, where

- $T_1 = \dfrac{\|x_{k+1} - x_k\|_2}{\|x_{k+1}\|_2}$ (or $T_1 = \|x_{k+1} - x_k\|_2$) and
- $T_2 = |f_{k+1} - f_k|$,

while x_k and x_{k+1} are the values obtained at iterations k and $k+1$ respectively, whereas f_k and f_{k+1} are their corresponding function values.

Problem A.

$$\begin{array}{ll} \underset{x\in\mathbb{R}^2}{\text{minimize}} & -x_1 - 2x_2, \\ \text{subject to} & 0 \le x_1 \le 1, \\ & x_2 \le 0. \end{array} \tag{3}$$

This problem is a linear program whose optimal solution is $x^* = (1, 0)^T$.

Similarly to Audet and Dennis Jr. (2007), we consider the initial point $x_0 = (0, 0)^T$, the mesh size parameter $\Delta_0 = 1$ and the four directions $\pm(1, 1)^T$ and $\pm(1, -1)^T$ to implement the pattern search algorithm. For the simplex search algorithm, we take the same initial point, the step size $s_0 = 1$ and the directions $(1, 0)^T$,$(0, 1)^T$.

Problem C-801.

$$\begin{array}{ll} \underset{x\in\mathbb{R}^2}{\text{minimize}} & f(x) \equiv 6x_1^2 + x_2^2 - 60x_1 - 8x_2 + 166, \\ \text{subject to} & 0 \le x_1 \le 10, \\ & 0 \le x_2 \le 10, \\ & x_1 + x_2 - x_1x_2 \ge 0, \\ & x_1 + x_2 - 3 \ge 0. \end{array} \tag{4}$$

This problem is a nonlinear program whose optimal solution x^* is unknown, but its function value is $f(x^*) = 7.563$. We consider in this problem the feasible initial point $x_0 = (5, 1)^T$.

Problem C-802.

$$\begin{array}{ll} \underset{x\in\mathbb{R}^2}{\text{minimize}} & f(x) \equiv 7x_1^2 + 3x_2^2 - 84x_1 \\ & \qquad - 34x_2 + 300, \\ \text{subject to} & 0 \le x_1 \le 10, \\ & 0 \le x_2 \le 10, \\ & x_1 x_2 - 1 \ge 0, \\ & 9 - x_1^2 - x_2^2 \ge 0. \end{array} \tag{5}$$

This problem is a nonlinear program whose optimal solution x^* is unknown, but its function value is $f(x^*) = -97.30952$. We consider in this problem the feasible initial point $x_0 = (2.5, 1.5)^T$.

5.2. Numerical results with feasible initial points. Table 1 shows the comparison between results obtained by Correia *et al*. (2009), using the SFA, and the NSFA presented in this work, for initial feasible points. Comparing the numerical results we can conclude what follows.

Problem A.

- Both methods found the solution to the problem, but the NSFA found it in the first iteration.
- The number of h evaluations with the NSFA is much smaller.
- Regarding the number of f evaluations, the NSFA, in an attempt to find feasible solutions better than that found in the first iteration (with a value of f minor), evaluated f many more times than the SFA did.
- In this search, the NSFA never found any infeasible solution.
- It should be noted that these methods might not be most appropriate to solve Problem A, since it is a linear problem. However, it has been used as a test for these methods that allows a comparison.

Problem C-801. Considering the feasible initial point $(5, 1)^T$:

- The SFA found the solution $(5, 1.25)^T$ with a function value 7.5625, a lower one than the solution presented in the original problem of the CUTE collection.
- In this case, the NSFA needed to evaluate h less often than the SFA, but it needs to evaluate f more often than SFA.
- The NSFA could not find a feasible solution which would be better than the initial point.
- But the NSFA found an infeasible solution $x_{ki} = (5, 2)^T$ with a function value of 4 and a violation value of 3.
- Thus, if the problem is relaxable and admits a violation $h_{\max} > 3$, the NSFA would be more efficient.
- Note that the solution obtained by the SFA is very close to the initial point.

In Fig. 2 the level curves of the objective function, the representation of the constraints and the feasible region (shadowed) are presented. Some points of the iterative processes, mentioned above, are also shown.

Problem C-802. Considering the feasible initial point $(2.5, 1.5)^T$:

- The SFA found the solution $(2.5, 1.6582)^T$ with a function value of 85.620, a higher one than the solution presented in the original problem of the collection CUTE.
- In this case, the NSFA needed to evaluate h and f less often than the SFA did, but it could not find a feasible solution which would be better than the initial point.
- The NSFA found an infeasible solution $x_{ki} = (2.5, 2)^T$ with a function value of 77.75 and a violation value of 1.25.

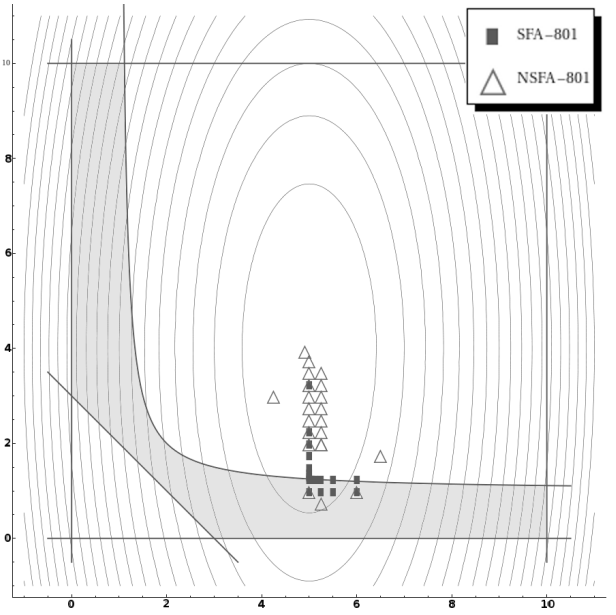

Fig. 2. Problem C-801 with feasible initial points.

Table 1. Numerical results with feasible initial points.

	Problem A		Problem C801		Problem C802	
Feasible initial point	$(0,0)^T$		$(5,1)^T$		$(2.5,1.5)^T$	
Numerical results	**SFA**	**NSFA**	**SFA**	**NSFA**	**SFA**	**NSFA**
Number of evaluations of h	129	123	118	69	153	61
Number of evaluations of f	158	3262	146	217	189	42
Unsuccessful iterations	29	39	29	33	37	25
Successful iterations	1	1	1	7	3	15
Last successful iteration	2	1	4	20	24	40
Best feasible solution: x_{kf}	$(1.0,0.0)^T$	$(1.0,0.0)^T$	$(5.0,1.25)^T$	$(5,1)^T$	$(2.5,1.6582)^T$	$(2.5,1.5)^T$
Function value: $f(x_{kf})$	-1.0	-1.0	7.5625	9	85.620	89.5
Best infeasible solution: x_{ki}	*	+	*	$(5,2)^T$	*	$(2.5,2)^T$
Violation value: $h(x_{ki})$	*	+	*	3	*	1.25
Function value: $f(x_{ki})$	*	+	*	4	*	77.75

SFA: Simplex Filter Algorithm (Correia *et al*., 2009), **NSFA**: New Simplex Filter Algorithm.
* Algorithm rejects infeasible solutions. + There was no infeasible solutions.
Successful iterations: iterations accepted in the filter.

- Thus, if the problem is relaxable and admits a violation $h_{\max} > 1.25$, the NSFA would be more efficient.

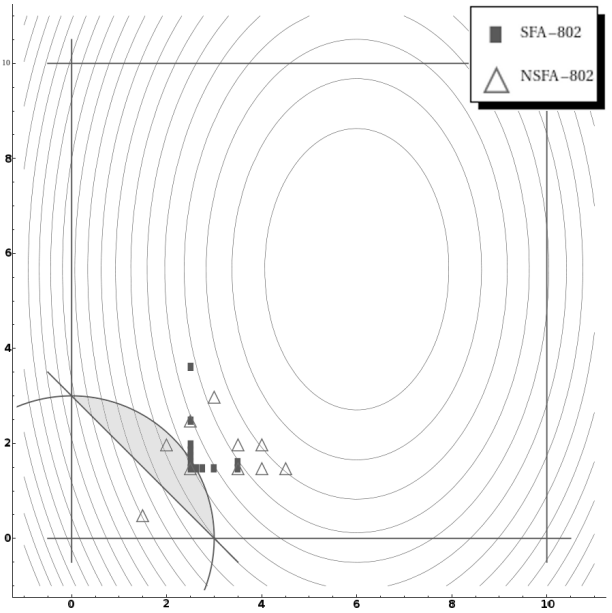

Fig. 3. Problem C-802 with feasible initial points.

- Note that the solution found by the SFA is very close to the initial point.

As for Problem C-801, Fig. 3 presents a graphical representation of Problem C-802, i.e., the level curves of the objective function, constraints, the feasible region and some points of the iterative processes.

5.3. Numerical results with infeasible initial points. Table 2 shows the comparison between results obtained by Correia *et al*. (2009), using the SFA, and the NSFA presented in this work, for initial infeasible points. Comparing the numerical results we can conclude what follows.

Problem C-801. If we consider an infeasible initial point $(0.1,-0.1)^T$, the NSFA is clearly more efficient:

- Identifying the best feasible solution $(5.1,0.9)^T$ with a function value of 9.67, lower than that given by the SFA and very close to the solution presented in the original problem of the CUTE collection.

- Finding an infeasible solution $(3.1,-0.1)^T$ with a function value of 38.47 and a violation value of 0.1. Thus, if the problem is relaxable and admits a violation $h_{\max} > 0.1$, it can be an excellent solution.

Table 2. Numerical results with infeasible initial points.

	Prob. C801		Prob. C802	
Infeasible initial point	$(0.1, -0.1)^T$		$(0.1, -0.1)^T$	
Numerical results	**SFA**	**NSFA**	**SFA**	**NSFA**
Number of evaluations of h	1226	41	1226	55
Number of evaluations of f	1526	41	1526	41
Unsuccessful iterations	39	24	40	23
Successful iterations	1	15	0	16
Last successful iteration	1	39	0	39
Best feasible solution: x_{kf}	$(0.1, -0.09)^T$	$(5.1, 0.9)^T$	$(0.1, -0.1)^T$	$(2.5, 1.4)^T$
Function value: $f(x_{kf})$	160.87	9.67	295.1	87.2
Best infeasible solution x_{ki}	*	$(3.1, -0.1)^T$	*	$(0, 0)^T$
Violation value: $h(x_{ki})$	*	0.1	*	1.01
Function value: $f(x_{ki})$	*	38.47	*	295.1

SFA: Simplex Filter Algorithm (Correia *et al*., 2009), **NSFA**: New Simplex Filter Algorithm.
* Algorithm rejects infeasible solutions.
Successful iterations: iterations accepted in the filter.

- The best number of f and h evaluations.
- The SFA cannot find a feasible solution which would be by far better than the initial point.

Problem C-802. If we consider an infeasible initial point $(0.1, -0.1)^T$, the NSFA is clearly more efficient:

- Finding the solution $(2.6, 1.4)^T$ with a function value 87.2, a higher one than that obtained using the SFA.
- Both of the obtained function values, were far from the function value $f(x^*) = -97.30952$ presented in the original problem of the collection CUTE, but the solution found by the NSFA is better.
- The NSFA found an infeasible solution $x_{ki} = (0, 0)^T$ with a violation value of 1.01, but with a function value of 295.1, which is no better than the initial point.
- The best number of f and h evaluations.
- Note that the SFA failed and was unable to find a solution approximation different from the initial point.

In Figs. 4 and 5 Problems C-801 and C-802 are presented, respectively, in the same conditions as in the case of the previous two figures. Some points of the iterative process, which started in infeasible points, are also shown.

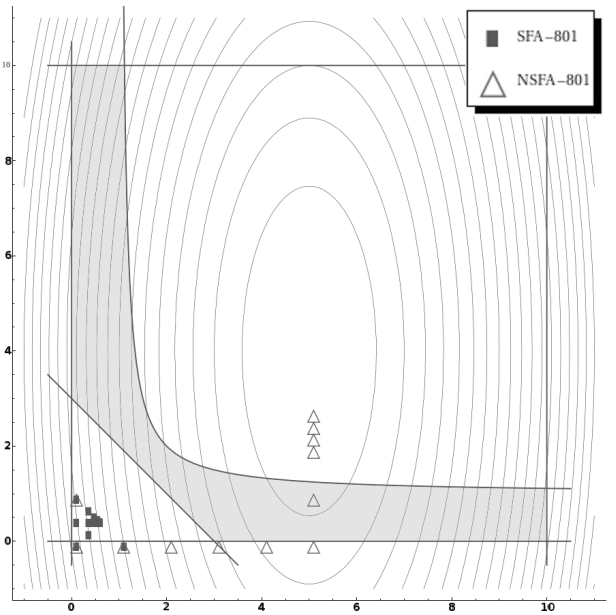

Fig. 4. Problem C-801 with infeasible initial points.

6. Conclusions and future work

Comparing the previous results, we can say that if the initial point is feasible, the NSFA and the SFA behave similarly finding feasible solutions. If the problem is relaxable and admits some violation, then the NSFA is more efficient. If the initial point is infeasible, the NSFA is clearly more efficient than the SFA.

In this work we presented a new direct search method, based on simplex methods, for general constrained optimization that combines the features of the simplex method and filter methods. This new approach is an alternative to commonly used methods to solve such problems. With regard to our previous approach, this allows creating positive expectations, since the results are

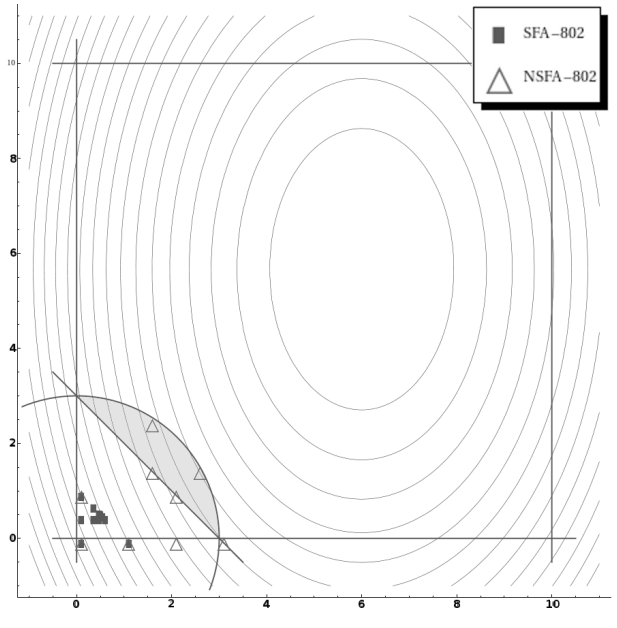

Fig. 5. Problem C-802 with infeasible initial points.

promising and this algorithm has a strong component with respect to the phase of optimality.

In the future, our goal is to create a web application able to solve constrained or unconstrained nonlinear problems. The methods are implemented using the Java technology. Our application will give the user the option to indicate the type of problem he/she wants to solve: a constrained or an unconstrained nonlinear problem. If the user wants to solve the latter, he/she can use immediately direct search methods: pattern search methods or simplex methods. If the user wants to solve the former, he/she must first choose between penalty methods or filter methods and then apply one of direct search methods.

Our first step to build the application was the implementation of direct search methods in the Java language. Then we implemented some penalty methods. Now we presented an alternative to those: a new direct search method, based on simplex methods, for general constrained optimization, that combines the features of the simplex method and filter methods. Our next objective is the implementation of dynamic penalty methods, the improvement of the GUI (Graphical User Interface) application. We also plan to perform more tests in order to improve our algorithms.

References

Audet, C. (2004). Convergence results for pattern search algorithms are tight, *Optimization and Engineering* **2**(5): 101–122.

Audet, C., Bchard, V. and Le Digabel, S. (2008). Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search, *Journal of Global Optimization* **41**(2): 299–318.

Audet, C. and Dennis Jr., J.E. (2007). A mads algorithm with a progressive barrier for derivative-free nonlinear programming, *Technical Report G-2007-37*, GERAD Reports, Polytechnic School of Montreal, Montreal.

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

Audet, C. and Dennis Jr., J. (2004). A pattern search filter method for nonlinear programming without derivatives, *SIAM Journal on Optimization* **5**(14): 980–1010.

Audet, C. and Dennis Jr., J.E. (2002). Analysis of generalized pattern searches, *SIAM Journal on Optimization* **13**(3): 889–903.

Audet, C., Dennis Jr., J.E. and Le Digabel, S. (2008). Globalization strategies for mesh adaptative direct search, *Technical Report G-2008-74*, GERAD Reports, Polytechnic School of Montreal, Montreal.

Bertsekas, D.P. (1999). *Nonlinear Programming*, Athena Scientific, Belmont, MA.

Bongartz, I., Conn, A., Gould, N. and Toint, P. (1995). Cute: Constrained and unconstrained testing environment, *ACM Transactions and Mathematical Software* **21**(1): 123–160.

Byrd, R.H., Nocedal, J. and Waltz, R.A. (2008). Steering exact penalty methods for nonlinear programming, *Optimization Methods & Software* **23**(2): 197–213.

Correia, A., Matias, J., Mestre, P. and Serôdio, C. (2009). Derivative-free optimization and filter methods to solve nonlinear constrained problems, *International Journal of Computer Mathematics* **86**(10): 1841–1851.

Fletcher, R. and Leyffer, S. (2002). Nonlinear programming without a penalty function, *Mathematical Programming. Series A* **91**(2): 239–269.

Fletcher, R., Leyffer, S. and Toint, P. L. (2006). A brief history of filter method, *Technical Report ANL/MCS-P1372-0906*, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL.

Karas, E.W., Ribeiro, A.A., Sagastizábal, C. and Solodov, M. (2006). A bundle-filter method for nonsmooth convex constrained optimization, *Mathematical Programming* **1**(116): 297–320.

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

Aldina Correia: *Academic degrees*: degree in mathematics (teaching), University of Trás-os-Montes and Alto Douro (UTAD), 2002; M.Sc. in applied mathematics, Faculty of Science, University of Porto, 2004; Ph.D. student in applied mathematics, UTAD, since 2006. *Professional positions:* lecturer, School of Industrial Studies and Management, Polytechnic Institute of Porto (ESEIG-IPP), 2003–2009; lecturer, School of Technology and Management Felgueiras, Polytechnic Institute of Porto (ESTGF-IPP), since 2009.

João Matias: *Academic degrees:* degree in mathematics (teaching), Faculty of Science and Technology, University of Coimbra, 1990; M.Sc. in informatics, Department of Informatics, School of Engineering, University of Minho, 1996; Ph.D. in applied mathematics, University of Trás-os-Montes and Alto Douro, 2003. *Professional positions:* invited lecturer, University of Trás-os-Montes and Alto Douro, Department of Mathematics, 1991–1996; lecturer, University of Trás-os-Montes and Alto Douro, Department of Mathematics, 1996–2003; assistant professor, University of Trás-os-Montes and Alto Douro, Department of Mathematics, since 2003.

Pedro Mestre: *Academic degrees:* degree in electrical engineering, University of Trás-os-Montes and Alto Douro, 1997; M.Sc. in industrial electronics, Minho University, 2000; Ph.D. in electrical engineering, University of Trás-os-Montes and Alto Douro, 2007. *Professional positions:* collaborator, University of Trás-os-Montes and Alto Douro, Engineering Department, 1995–2000; lecturer, University of Trás-os-Montes and Alto Douro, Engineering Department, 2000–2007; assistant professor, University of Trás-os-Montes and Alto Douro, Engineering Department, since 2007.

Carlos Serodio: *Academic degrees:* degree in electrical engineering, University of Trás-os-Montes and Alto Douro, 1990; Ms.C. equivalence degree, University of Trás-os-Montes and Alto Douro, 1996; Ph.D. in electrical engineering, University of Trás-os-Montes e Alto Douro, 2002. *Professional positions:* lecturer, University of Trás-os-Montes and Alto Douro, Engineering Department, 1990–2002; assistant professor, University of Trás-os-Montes and Alto Douro, Engineering Department, since 2002; researcher at CITAB. *Research interests*: mobile communications, ad-hoc wireless sensors networks, CAN, RFID and Java-enabled dynamic and distributed systems.

Received: 13 January 2010
Revised: 24 June 2010