International Journal of Computer Mathematics
Vol. 00, No. 00, Month 2007, 1–23

Taylor & Francis
Taylor & Francis Group

A filter algorithm: comparison with NLP solvers

CÂNDIDA ELISA P. SILVA*† and M. TERESA T. MONTEIRO‡

†Management and Industrial School, Polytechnic Institute of Oporto, Portugal
‡Production and Systems Department, Engineering School, University of Minho, Portugal

(Received 01 November 2006; revised version received 02 January 2007; accepted 05 January 2007)

The purpose of this work is to present an algorithm to solve nonlinear constrained optimization problems, using the filter method with the inexact restoration (IR) approach. In the IR approach two independent phases are performed in each iteration—the feasibility and the optimality phases. The first one directs the iterative process into the feasible region, i.e. finds one point with less constraints violation. The optimality phase starts from this point and its goal is to optimize the objective function into the satisfied constraints space. To evaluate the solution approximations in each iteration a scheme based on the filter method is used in both phases of the algorithm. This method replaces the merit functions that are based on penalty schemes, avoiding the related difficulties such as the penalty parameter estimation and the non-differentiability of some of them. The filter method is implemented in the context of the line search globalization technique. A set of more than two hundred AMPL test problems is solved. The algorithm developed is compared with LOQO and NPSOL software packages.

Keywords: Nonlinear programming; Filter method; Inexact restoration

AMS Subject Classifications: 90C30; 65K99

1. Introduction

Fletcher and Leyffer [1] have proposed the filter method as an alternative to the merit functions. These authors presented a SQP algorithm with a trust-region technique to solve nonlinear constrained optimization problems. The motivation given by these authors for the development of the filter method is to avoid the necessity to determine a suitable value of the penalty parameter in the merit function. In constrained optimization, the two competing aims, minimization of the objective function and the satisfaction of the constraints, are combined into a single minimization problem, using a merit function. In the filter strategy two separated aims are considered instead of a combination of both.

The inexact-restoration method introduced by Martínez and Pilotta [2, 3] (see also [4]) and also used in [5] treats feasibility and optimality as two independent phases. Each iteration of the method proceeds in two phases. At the first phase, feasibility of the current iterate is

*Corresponding author. Email: candidasilva@eseig.ipp.pt

International Journal of Computer Mathematics
ISSN 0020-7160 print/ISSN 1029-0265 online © 2007 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/00207160701203401

improved (a 'more feasible' point is computed with respect to the current point). The second phase reduces the objective function value (an 'optimal' point is calculated) in an approximate feasible set. The point that results from the second phase is compared with the current point using a merit function that combines feasibility and optimality.

Gonzaga *et al.* [6] suggests an algorithm to solve nonlinear constrained optimization based on the inexact restoration (IR) approach with the filter method. In this theoretical algorithm the methods in IR phases are not specified and its implementation motivates this work – combining the IR approach with the filter method. In each phase of the IR method a filter scheme with line search technique is used instead of a merit function.

This paper is organized into four sections. The next section defines the problem to be solved, presents the concepts of inexact-restoration and the filter method with line search. Section 3 presents the Filter IR algorithm specifying the feasibility and optimality phases, the filter update scheme and a graphic example. Finally numerical experiments are performed to compare the algorithm with LOQO and NPSOL solvers. Some conclusions and suggestions to future work are reported.

2. Optimization approaches

The general nonlinear constrained optimization problem is defined by

$$\begin{cases} \min & f(x), \\ s.t. & lb_x \leq x \leq ub_x, \\ & lb_c \leq c(x) \leq ub_c, \end{cases} \tag{NLP}$$

where $f(x) : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear objective function, $x \in \mathbb{R}^n$ and lb_x, $ub_x \in \mathbb{R}^n$ are the lower and the upper bounds of the variable x, respectively, and $c(x) : \mathbb{R}^n \to \mathbb{R}^m$ is a set of m general constraints whose lower and upper bounds are lb_c and ub_c, respectively. It is possible to include one-sided constraints by setting the lower bound to $-\infty$ or the upper bound to ∞, depending on which bound is required.

2.1 *Inexact restoration*

The point of view of the IR approach is that feasibility is an important feature of the problem that must be controlled independently of optimality. So, the methods based on IR consider feasibility and optimality at different phases of a single iteration. A well known drawback of feasible methods is their inability to follow very curved domains, which can cause very short steps to be computed far from the solution. IR methodology tries to avoid this inconvenience using procedures that automatically decrease the tolerance of infeasibility as the solution is approximated. In this way, large steps on an enlarged feasible region are computed at the beginning of the process. An essential feature of this methodology is that one is free to choose different algorithms both for the feasibility and for the optimality phase, so that problem characteristics can be exploited. In [5] it is shown that the use of the Lagrangian function in the optimality phase favours practical fast convergence near the solution.

2.2 *Line search filter method*

In the constrained optimization problem two conflicting criteria are always present, the minimization of the objective function, $f(x)$, and the satisfaction of the constraints $c(x)$. All the

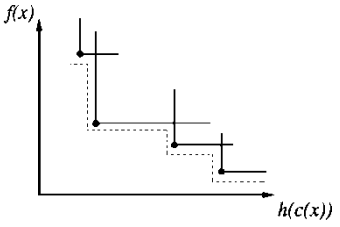

Figure 1. Filter with sufficient decrease.

constraints defined in (NLP) will be handled as $c(x) \leq 0$ and the optimization problem is formulated in its minimization form. A merit function combines these aims into a single minimization problem. These two competing aims can be written as two minimization problems

$$\begin{aligned} &\min f(x), \\ &\min h(c(x)), \end{aligned} \tag{1}$$

where

$$h(c(x)) := \left\| c^+(x) \right\|_1 := \sum_{j=1}^{m} c_j^+(x)$$

with $c_j^+(x) = \max(0, c_j(x))$ being the sum of the constraints violation.

A filter is defined as a set of pairs $(f(x^k), h(x^k))$ so that no pair dominates any other. The dominance concept comes from multi-objective optimization and establishes that $(f(x^k), h(x^k))$ dominates $(f(x^l), h(x^l))$ if and only if both $f(x^k) \leq f(x^l)$ and $h(x^k) \leq h(x^l)$ are verified, where $(f(x^l), h(x^l))$ represents all the filter pairs. The filter, in figure 1, can be represented in $\mathbb{R}^2$ as a set of pairs (f, h), defining the forbidden and allowed regions as well as the envelope that defines the sufficient reduction imposed in the filter acceptance. When a point x^k is accepted by the filter, which means that it belongs to the allowed region, the corresponding pair $(f(x^k), h(x^k))$ is introduced in the filter. All the pairs dominated by it will be removed from the filter.

Fletcher and Leyffer [1] introduced the concept of a filter in an SQP algorithm with the trust-region technique. The filter SQP algorithm starts with an initial point, the solution of the current QP subproblem, produces a trial step, and with it the new trial point is determined. If it is accepted by the filter, this will be the new point, otherwise the step will be rejected, the trust-region radius is reduced and a new QP subproblem must be solved again. With this algorithm the usual criterion of descent in the penalty function is replaced by the requirement that the new point is accepted by the filter. A more feasible and/or optimal point is achieved by the adjustment of the trust-region in the iterations of the algorithm. The idea is to use the filter as a criterion for accepting or rejecting a step.

Antunes and Monteiro [7] presented a SQP algorithm based on Fletcher and Leyffer's idea where the trust-region is replaced by the line search technique (see also [8–12] for other line search approaches). The main difference is that instead of using the trust-region to determine a trial point, they use the line search method. The combination of the line search technique with the filter method determines the step length acceptance. First the search direction is computed from a given initial point, solution of the current QP subproblem and with it a trial point x^{k+1} is computed. After this the point will be checked by the filter and if it is accepted, it is introduced

into the filter. Otherwise, α is divided by two and the trial point x^{k+1} is updated and checked again by the filter, without solving another QP subproblem.

3. Filter IR algorithm

The advantage of IR philosophy is the freedom of choosing the internal algorithm for each phase. In the feasibility phase a new optimization problem is solved in order to enforce the iterative procedure towards the feasible region. The problem aims to minimize the constraints violation, in the set of satisfied constraints, to obtain a 'more feasible' intermediate point. The optimality phase solves the initial optimization problem from the intermediate point.

3.1 *Feasibility phase*

The iteration k starts with the evaluation of the x^k feasibility – if x^k is already feasible the feasibility phase is not performed. From x^k, the constraints of the original problem are evaluated in order to identify the subsets of the violated and satisfied constraints. These subsets are linearized in x^k to formulate the feasibility phase optimization problem as follows:

$$c(x^k + d) \approx c(x^k) + \nabla c(x^k)^T d.$$

The new problem has as objective function the sum of all the linearized violated constraints, subject to the linearized satisfied constraints set. From the point x^k the aim is to reach a more feasible point z^k. This point is obtained solving the following LP subproblem:

$$\begin{aligned} &\min_{d \in \mathbb{R}^n} \sum_{i \in J} A_i^k d, \\ &s.t.\ A_i^k d + c_i^k \leq 0, \quad i \in J^{\perp}, \end{aligned} \tag{LP}$$

where $A^k = \nabla c(x^k)^T$ is the Jacobian matrix of the constraints and, J and $J^{\perp}$ are the sets of violated and satisfied constraints, respectively. The solution of this LP subproblem, which is also an iterative procedure, is the search direction d_{fea}^k (denoted by d in (LP)). The intermediate point $z^k = x^k + \alpha\, d_{\text{fea}}^k$ is obtained by the line search technique using the filter, where $\alpha \in \mathbb{R}$ is the step length. Note that both objective and constraints functions are linear functions in d_{fea}.

The management of this Filter IR is quite different from the one used by Fletcher and Leyffer in [1] and Antunes and Monteiro in [7]. In Filter IR a temporary pair $(f(x^k), h(x^k))$ corresponding to x^k is used to evaluate the next trial point x^{k+1}.

The intermediate point z^k is accepted by the filter if the corresponding pair $(f(z^k), h(z^k))$ is not dominated by any other pair in the filter. The sufficient decrease condition for acceptance by the filter is

$$h < \beta h(x^l) \quad \text{or} \quad f < f(x^l) - \gamma h \tag{2}$$

for all pairs $(f(x^l), h(x^l))$ in the filter, where β and γ are parameters such that $0 < \gamma < \beta < 1$, with β close to one and γ close to zero. Then it is also verified if this pair causes a sufficient

decrease in h when compared to the temporary pair $(f(x^k), h(x^k))$:

$$h(z^k) \leq (1-\gamma)h(x^k). \tag{3}$$

If conditions (2) and (3) are not satisfied then the intermediate point z^k is rejected and α is divided by two until the point is accepted or α is smaller than a tolerance. If the intermediate point z^k is a Kuhn–Tucker point and there is no constraints violation at z^k then the algorithm terminates with success, otherwise the next phase will be performed.

3.2 *Optimality phase*

The goal of the optimality phase is to reduce the objective function from the z^k point. The corresponding QP subproblem is an approximation to the original problem (NLP):

$$\begin{aligned} &\min_{d\in\mathbb{R}^n} \frac{1}{2} d^T W^k d + d^T g^k \\ &s.t.\ A^k d + c^k \leq 0 \end{aligned} \tag{QP}$$

where $g^k = \nabla f(x^k)$ is the gradient of the objective function, $A^k = \nabla c(x^k)^T$ is the Jacobian matrix of general constraints $c(x^k)$ and $W^k = \nabla^2 L(x^k, \lambda^k)$ is the Hessian matrix of the Lagrangian function. The solution of this QP subproblem is the search direction d^k_{opt} (denoted by d in (QP)). With this direction the next trial point $x^{k+1} = z^k + \alpha d^k_{\text{opt}}$ is obtained. Then this point is tested by the filter, which is a similar procedure to the one described in the previous phase and it is compared to the intermediate point z^k:

$$f(x^{k+1}) \leq (1-\gamma) f(z^k). \tag{4}$$

If this condition is verified and the point is accepted by the filter then x^{k+1} is the next point.

3.3 *Filter update and stop criterium*

Finally, if the following condition

$$f(x^{k+1}) \geq f(x^k) - \min(h(x^k)^2, \omega) \tag{5}$$

is verified then the temporary pair $(f(x^k), h(x^k))$ is inserted into the filter and the dominated pairs are removed (ω in (5) is a small positive constant).

The first condition of the stop criterium is concerned with stationarity (normalized residuum r) and the second with feasibility analysis:

$$r(x^*) = \frac{\|g^* + \nu^* + A^*\lambda^*\|_2}{\max\{\mu_{\max}, 1.0\}} \leq \epsilon \quad \text{and} \quad h(x^*) \leq \xi$$

where $g^* = \nabla f(x^*)$, $A^* = \nabla c(x^*)$, ν and λ are the Lagrange multiplier vectors of the simple bounds constraints and of the general constraints, respectively, ϵ is a small positive constant, ξ represents the zero and $\mu_{\max} = \max_i \left\{ \|g^*\|_2, |\nu_i|, \left\|a_i^*\right\|_2 \left|\lambda_i^*\right| \right\}$.

The next subsection presents the Filter IR algorithm.

3.4 *The algorithm*

FILTER IR ALGORITHM:

x^0, λ^0, $\mathcal{F}^0$ initial filter, $(f(x^0), h(x^0))$ temporary pair, $k = 0$;
REPEAT
  Feasibility phase:
    Find z^k:
      Solve subproblem (LP) at x^k to find ${d_{\text{fea}}}^k$;
      $\alpha = 1$;
      REPEAT
        $z^k = x^k + \alpha\ {d_{\text{fea}}}^k$;
        If $(f(z^k), h(z^k))$ is accepted by the filter **And** $h(z^k) \leq (1 - \delta)h(x^k)$ **Then**
          Accept z^k;
        Else
          Reject z^k;
          $\alpha = \dfrac{\alpha}{2}$;
      UNTIL Accept z^k **or** $\alpha \leq TolAlpha$;
  If z^k satisfies the stop criterium **Then** Stop with success.
  Else
    Optimality phase:
      Find x^{k+1}:
        Solve subproblem (QP) at z^k to find ${d_{\text{opt}}}^k$;
        $\alpha = 1$;
        REPEAT
          $x^{k+1} = z^k + \alpha\ {d_{\text{opt}}}^k$;
          If $(f(x^{k+1}), h(x^{k+1}))$ is accepted by the filter
          And $f(x^{k+1}) \leq (1 - \delta) f(z^k)$ **Then**
            Accept x^{k+1};
          Else
            Reject x^{k+1};
            $\alpha = \dfrac{\alpha}{2}$;
        UNTIL Accept x^{k+1} **or** $\alpha \leq TolAlpha$;
    Filter update:
      If $f(x^{k+1}) \geq f(x^k) - \min(h(x^k)^2, \omega)$ **Then**
        If $(f(x^k), h(x^k))$ is accepted by the filter **Then**
          Insert the temporary pair $(f(x^k), h(x^k))$ in the filter
          Remove all pairs dominated by $(f(x^k), h(x^k))$ from the filter.
  $k = k + 1$
  $x^k = x^{k+1}$
UNTIL x^k verify stop criterium.

3.5 *Convergence assumptions*

The global convergence was studied by Gonzaga [6], where it is proved that this method is independent of internal algorithms used in each iteration, since these algorithms satisfy some requirements in their efficiency. It is shown that, under some assumptions, for a filter with a minimal size, the algorithm generates a stationary accumulation point and that for bigger filters, all the accumulation points are stationary.

The general hypotheses are:

H1: The iterates x^k and z^k remain in a convex compact domain $X \subset \mathbb{R}^n$.
H2: The objective and constraints functions ($f(x)$ and $c(x)$) are Lipschitz continuously differentiable in an open set containing X.

H3: All feasible accumulation points $\overline{x} \in X$ of x^k satisfy the Mangasarian–Fromovitz (M-F) qualification condition, namely, the gradients of equality constraints are linearly independent, and there exists a direction $d \in \mathbb{R}^n$ such that $A_E(\overline{x})d = 0$ and $A_{\overline{I}}(\overline{x})d < 0$, where $\overline{I} = \{i \in I | c_i(\overline{x}) = 0\}$ ($A_E(.)$ and $A_{\overline{I}}(.)$ are the Jacobian matrix of equality constraints and active inequality constraints, respectively).

There are two more assumptions related to the internal algorithms. The first one (H4) with the feasibility phase algorithm and the other (H5) with the optimality phase algorithm:

H4: At all iterations $k \in \mathbb{N}$, the feasibility step must satisfy $h(z^k) < (1 - \alpha)h(x^k)$ and z^k cannot belong to the forbidden region.

H5: Given a feasible non-stationary point $\overline{x} \in X$, there exists a neighbourhood V of $\overline{x}$ such that for any iterate $x^k \in V$, $f_0(x^{k+1}) \leq f_0(z^k)$ and x^{k+1} cannot belong to the forbidden region.

3.6 *A graphic example*

In order to illustrate the algorithm behaviour, a graphic example has been developed. Consider figure 2 with some pairs representing a filter, defining the allowed and the forbidden region. The blue point is the initial temporary point. This point is in the allowed region of the filter,

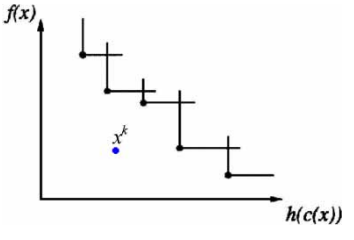

Figure 2. Initial temporary point x^k.

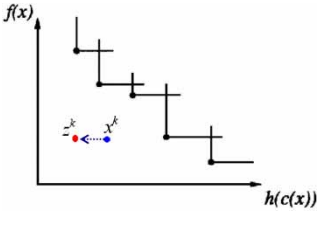

Figure 3. Feasibility—intermediate point z^k.

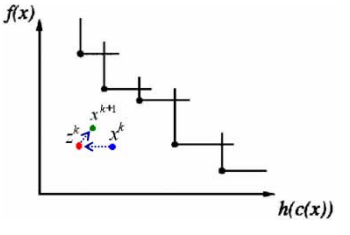

Figure 4. Optimality—new trial point x^{k+1}.

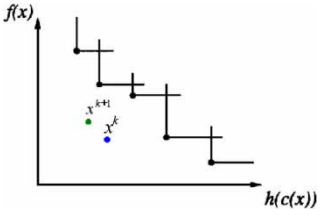

Figure 5. Analysis of filter update with temporary point x^k.

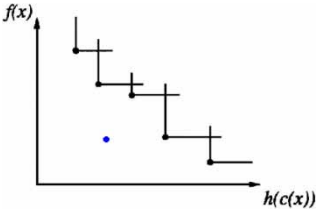

Figure 6. Filter update with the temporary point x^k.

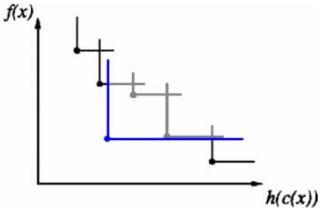

Figure 7. Dominated points removal.

but it does not satisfy all the constraints, so, the feasibility phase will be performed. From the feasibility phase results the intermediate point z^k, shown in figure 3. z^k is not a KKT point, so, the optimality phase will be performed, and the point x^{k+1} is reached from z^k, in figure 4.

Finally, it will be analysed if the temporary point x^k will be inserted in the filter or not. From figure 5, it can be stated that point x^k is in the filter allowed region, and that the point x^{k+1}, in green, has a worse value of f than x^k, in blue. Then, the temporary point x^k must be inserted in the filter (figure 6). This update leads to the filter dominance analysis.

As can be seen in figure 7, the point inserted dominates two of the filter points, so these will be removed from the filter. After the dominated points removal, the filter will be like in figure 8. The new trial point x^{k+1}, shown in figure 9, will be the next temporary point tested in the next iteration.

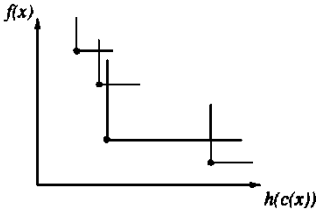

Figure 8. Final filter.

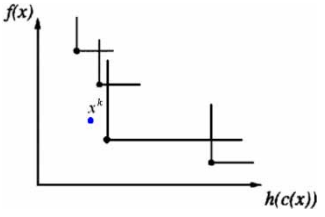

Figure 9. Final filter and next temporary trial x^k.

4. Numerical tests

This section presents the numerical experiences performed to test the algorithm. The computational experiences were made on a *centrino* with 504 MB of RAM. The algorithm was implemented in C language for the Windows operating system. The (LP) and (QP) subproblems are solved by the LSSOL subroutine [13] from the NPSOL solver [14].

The 235 test problems (www.sor.princeton.edu/˜rvdb/ampl/nlmodels) are in AMPL language [15]. Some of these problems are from the CUTE database codified in the AMPL language. The program was interfaced with the AMPL modelling language [16] to read the problems automatically, as figure 10 shows. The file written in AMPL language is converted into an intermediate file and then this file is automatically loaded into the algorithm structures.

The first numeric results of this algorithm were presented in [17], where the algorithm robustness and the relevance of the feasibility phase was evaluated, by testing two versions of the algorithm—the first one with feasibility phase and the second with optimality phase only. The results show that Filter IR with feasibility phase reaches an optimal point faster. In this paper, the filter management is analysed and a comparison with LOQO and NPSOL solvers is presented.

Filter management analysis refers to the number of filter updates during the algorithm process and its final size. The following graphics show, for problems that converge, that the number of filter updates and the number of elements in the end is very small, less than five in more than 90% of cases (figures 11 and 12).

For problems that do not converge, 52 out of 235 problems, the number of filter updates increases considerably (figure 13). In 60% of the problems, the number of filter updates is between 400 and 800, and in 25% it is greater than 800. Nevertheless, the final filter size is small—less than 40 in 73% of problems (figure 14).

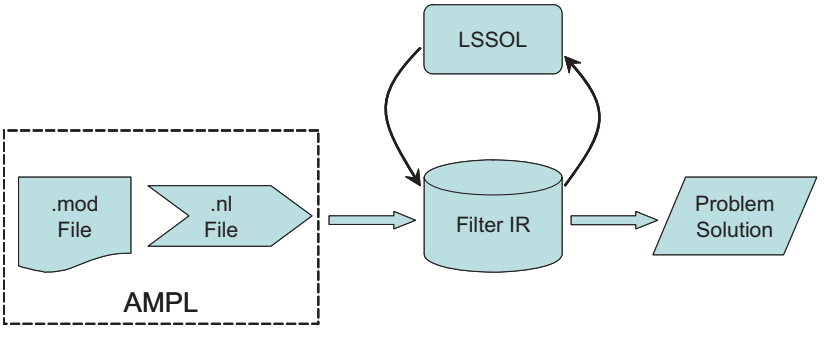

Figure 10. AMPL interface with Filter IR.

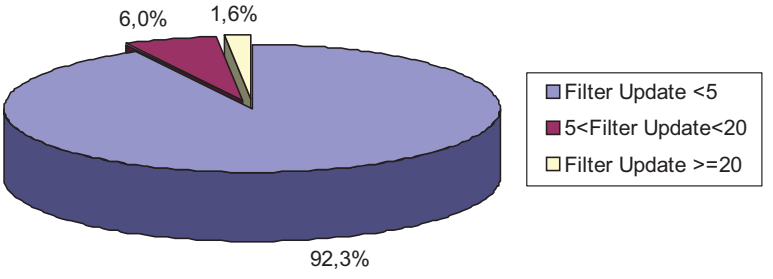

Figure 11. Filter update.

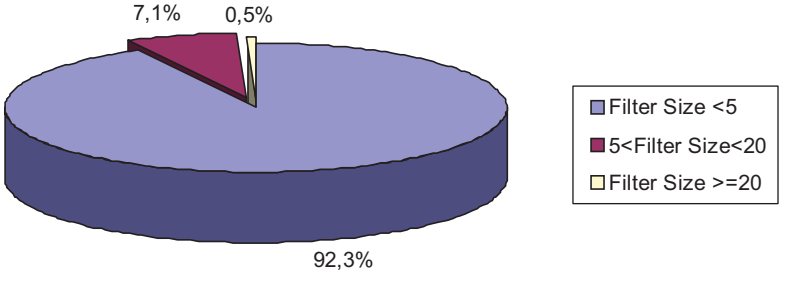

Figure 12. Elements in final filter.

4.1 *Comparison with LOQO and NPSOL solvers*

The computational experience uses the IR algorithm to test all the problems in order to evaluate the algorithm robustness. 235 inequality constrained optimization problems are tested. These results are reported in table 1. The table shows the results of the three codes: Filter IR, LOQO [18] (an interior point method) and NPSOL [14] (an SQP method). The table shows the problem name (*Problem*), its dimension (n) and the number of constraints

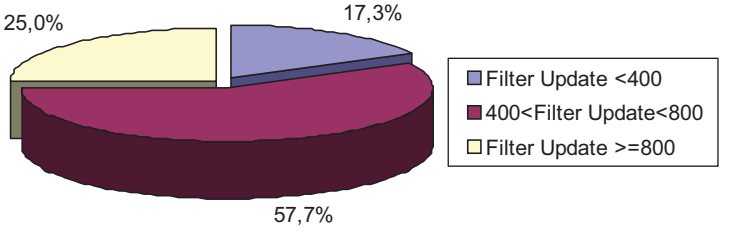

Figure 13. Filter update.

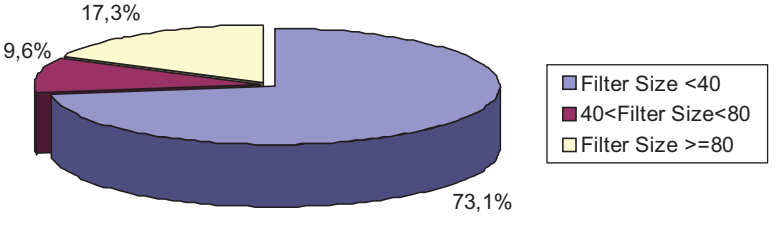

Figure 14. Elements in final filter.

Table 1. Filter IR versus LOQO and NPSOL.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
alsotame	2	1	3	31	0.08	12	23	9.81	N	4	6	9.81	N	*d*)
biggsc4	4	7	2	4	−24.38	18	35	−24.5	N	2	3	−24.38	S	
bqp1var	1	0	1	1	0	10	19	6.27	S	1	2	0	S	
camel6	2	0	8	9	−0.22	12	23	−0.22	N	8	15	−0.22	S	
cantilvr	5	1	10^3	19962	1.34	16	31	1.34	S	19	22	1.34	S	*b*)
cb2	3	3	10^3	19981	1.95	11	21	1.95	S	10	13	1.95	S	*b*)
cb3	3	3	10^3	19981	2	11	21	2	S	6	7	2	S	*b*)
chaconn1	3	3	10^3	19962	1.95	11	21	1.95	S	8	11	1.95	S	*b*)
chaconn2	3	3	10^3	19962	2	11	21	2	S	5	6	2	S	*b*)
congigmz	3	5	16	243	28	33	65	28	S	4	5	28	S	
dembo7	16	20	10^3	20013	174.79	10^3	2016	210.12	N	18	23	174.79	S	
dipigri	7	4	10^3	19941	680.63	11	22	680.63	S	14	29	680.63	S	*b*)
dixmaana	15	1	1	1	2975324.9	79	278	1	N	24	33	1	N	
dixmaanc	15	1	1	5	168.12	20	40	1	N	16	22	1	N	
dixmaane	15	1	1	5	101.35	19	37	1	N	22	24	1	N	
dixmaanf	15	1	1	5	92.2	20	40	1	N	22	24	1	N	
dixmaang	15	1	1	5	147.83	23	52	1	N	23	25	1	N	
dixmaanh	15	1	1	5	266.59	10^3	2011	1.0	N	27	31	1	N	
dixmaani	15	1	1	5	91.32	19	39	1	N	39	42	1	N	
dixmaanj	15	1	1	6	77.88	18	43	0.0	N	29	52	−0.32	N	*a*)
dixmaank	15	1	1	6	132.75	18	38	0.0	N	22	27	0	N	
dixmaanl	15	1	1	5	265.26	18	41	0.0	N	19	25	0	N	
engval1	2	1	15	15	0	20	39	0.0	S	26	32	0	N	

(*continued*)

Table 1. Continued.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
engval4	5	1	9	47	3.63	18	37	3.63	S	19	34	3.63	S	*c*)
explin	120	1	20	58	27.62	103	315	0.0	N	10	11	0.0	N	
expquad	120	1	1	1	0	10	19	0.00	N	2	3	0.00	N	
fletcher	4	4	10^3	39980	64	14	27	19.53	N	1	1	4	N	*a*)
gigomez2	3	3	10^3	19981	1.95	11	21	1.95	S	9	12	1.95	S	*b*)
gigomez3	3	3	10^3	19981	2	10	19	2	S	7	9	2	S	*b*)
goffin	51	50	3	41	0	12	23	0.0	N	1	1	0.0	N	*d*)
harkerp2	100	1	18	18	0.002	38	75	0.0	N	17	90	39622216.5	N	
hatfldb	4	0	11	128	0.006	12	23	0.006	S	13	16	0.006	S	
hatfldc	4	0	12	13	0	26	51	0	S	22	30	0	S	*c*)
hatfldh	4	7	8	256	−24.38	20	39	−24.5	N	2	4	−24.375	S	
himmelp1	2	0	6	26	−62.05	15	29	−62.05	S	6	12	−23.9	N	
himmelp2	2	1	6	7	−62.05	19	38	−62.05	S	30	31	−62.05	S	*c*)
himmelp3	2	2	4	4	−59.01	16	31	−59.01	S	3	4	−59.01	S	
himmelp4	2	3	3	22	−59.01	16	32	−59.01	S	10	12	−59.01	S	*c*)
himmelp5	2	3	3	3	−59.01	36	76	−59.01	S	10	17	−59.01	S	*c*)
himmelp6	2	4	4	5	−59.01	25	51	−59.01	S	9	11	−59.01	S	*c*)
hs3mod	2	0	1	1	0	12	23	0.0	S	4	8	0	S	*c*)
hs21mod	7	2	8	10	−95.96	30	60	−95.96	S	16	19	−95.96	S	*c*)
hs35mod	2	1	1	1	0.25	16	31	0.25	S	3	4	0.25	S	*c*)
hs44new	4	5	3	4	−13	18	35	−13	S	4	6	−15	N	
hs100mod	7	4	10^3	19951	678.76	15	29	678.76	S	14	28	678.76	S	*b*)
hs268	5	5	1	1	9994.42	18	35	9994.42	S	15	22	9994.42	S	*c*)

hubfit	2	1	1	1	0.017	12	23	0.017	S	6	7	0.017	S	*c*)
humps	2	1	1	1	13669.4	271	595	0.0	N	194	338	0	N	
liarwhd	36	1	19	19	0.0	27	55	0.0	S	26	27	0.0	S	
liswet2	103	100	1	1	−34.97	19	37	−34.97	S	2	3	−34.97	S	*c*)
liswet3	103	100	1	1	−21.38	19	37	−21.38	S	2	3	−21.38	S	*c*)
liswet4	103	100	1	1	−15.55	23	45	−15.55	S	2	3	−15.55	S	*c*)
liswet5	103	100	1	1	−331.06	16	31	−331.06	S	2	4	−331.06	S	*c*)
liswet6	103	100	1	1	−45.19	19	37	−45.19	S	2	3	−45.19	S	*c*)
liswet7	103	100	1	1	−51.24	28	55	−51.24	S	2	3	−51.24	S	*c*)
liswet8	103	100	1	1	−51.25	21	41	−51.25	S	2	3	−51.25	S	*c*)
liswet9	103	100	1	1	−29.97	77	153	−29.97	S	2	3	−29.97	S	*c*)
liswet10	103	100	1	1	−52.24	19	37	−52.24	S	2	3	−52.24	S	*c*)
liswet11	103	100	1	1	−51.24	52	103	−51.24	S	2	3	−51.24	S	*c*)
liswet12	103	100	1	1	−32.11	75	149	−32.11	S	2	3	−32.11	S	*c*)
logros	2	1	2	6	0.67	79	227	0	N	77	109	0.00	N	
lootsma	3	2	10	145	1.41	10^3	11363	18.88	N	1	1	0	N	*a*)
lsqfit	2	1	1	1	0.03	12	23	0.03	S	5	6	0.03	S	*c*)
madsen	3	6	10^3	19924	0.62	24	48	0.62	S	13	17	0.62	S	*b*)
makela1	3	2	10^3	1	0	14	28	−1.41	N	9	13	−1.41	N	
makela3	21	20	23	384	0	16	31	0.0	N	28	31	0.0	N	
makela4	21	40	2	21	0	12	23	0.0	S	20	56	0.0	S	*c*)
matrix2	6	2	14	38	0	26	51	0.0	N	21	24	0.0	N	
mifflin1	3	2	10^3	17966	−1.088	9	17	−1	N	8	10	−1	N	

(*continued*)

Table 1. Continued.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
mifflin2	3	2	3	22	−2.98	13	25	−1	N	9	12	−1	N	*d*)
minmaxrb	3	4	2	21	0	13	25	0	S	4	5	0	S	*c*)
nondquad	100	1	18	18	0	23	45	0.002	N	10^3	1007	0.006	N	
oslbqp	8	0	1	21	6.25	40	80	6.25	S	1	2	6.25	S	
pfit1	3	1	1	1	372.12	337	809	1.26	N	389	578	1.27	N	
pfit1ls	3	1	1	1	372.12	337	809	1.26	N	390	578	1.27	N	
pfit2	3	1	1	1	3660.08	339	802	35.99	N	557	807	35.97	N	
pfit2ls	3	1	1	1	3660.08	339	802	35.99	N	558	807	35.97	N	
pfit3	3	1	1	1	15280.62	345	840	253.87	N	433	612	253.97	N	
pfit3ls	3	1	1	1	15280.62	345	840	253.87	N	434	612	253.97	N	
pfit4	3	1	1	1	43522.45	334	816	943.33	N	503	736	943.3	N	
pfit4ls	3	1	1	1	43522.45	334	816	943.33	N	504	736	944.3	N	
polak1	3	2	10	104	2.72	14	27	2.72	S	13	15	2.72	S	
polak3	12	10	10^3	19905	5.88	24	47	5.88	S	23	29	5.88	S	*b*)
powell20	10	10	1	2	57.81	12	23	57.81	S	1	2	57.81	S	
power	10	1	15	15	0	21	41	0.0	N	37	43	0	S	
pspdoc	4	0	5	27	2.41	11	21	2.41	S	12	13	2.41	S	*c*)
qrtquad	12	1	15	41	0.0	29	83	0.0	N	1	2	0	N	
qudlin	12	1	3	42	0.75	17	36	−0.0	N	4	5	0	N	
rosenmmx	5	4	10^3	17974	−48.27	15	29	−44	N	18	33	−44	N	
s365mod	7	4	10^3	61	0.25	28	66	52.14	N	26	39	52.14	N	
simbqp	2	0	1	1	0	13	25	0.0	S	4	6	0	S	*c*)
simpllpa	2	2	2	20	1	12	23	1	S	0	1	1	S	

simpllpb	2	3	2	20	1.1	13	25	1.1	S	2	4	1.1	S	
sineval	2	1	13	25	1.34	48	107	0.0	N	81	119	0	N	
sisser	2	1	1	7	2.79	19	38	0.0	N	40	42	0	N	
snake	2	2	1	1	0	10^3	11367	−16.91	N	272	564	0	N	
stancim	3	3	10^3	19981	5	41	81	5	S	2	11	7.46	N	*a*) & *b*)
tf12	3	100	10^3	20^4	0.65	15	29	0.65	S	20	32	0.65	S	*b*)
vardim	10	1	25	25	0	39	82	0.0	S	27	28	0	S	
womflet	3	3	1	1	0	11	21	6.05	N	182	519	0	N	
zecevic2	2	2	1	1	−4.13	11	21	−4.13	S	2	4	−4.13	S	*c*)
zecevic3	2	2	10^3	2500	97.31	12	23	97.31	S	8	11	97.31	S	*b*)
zecevic4	2	2	3	4	7.56	12	24	7.56	S	6	7	7.56	S	*c*)
zy2	3	1	5	8	2	14	27	2	S	6	16	2	S	
branin	2	0	5	26	0.4	15	29	0.4	S	7	10	0.4	S	
chi	2	0	5	8	−19.25	16	40	498.36	N	9	15	−34.28	N	*d*)
hs5	2	0	3	23	1.23	10	19	1.23	S	7	9	1.23	S	*c*)
hs15	2	2	3	21	306.5	16	31	306.5	S	5	6	306.5	S	*c*)
hs23	2	5	13	51	2	12	23	9.47	N	4	7	9.47	N	*d*)
hs35	3	1	1	1	0.11	11	21	0.11	S	7	8	0.11	S	*c*)
hs44	4	6	6	9	−15	12	23	−13	N	2	3	−13	N	*d*)
hs64	3	1	10^3	19810	6299.8	27	53	6299.8	S	29	39	6299.8	S	*b*)
kowalik	4	0	11	33	0.0	12	23	0.0	S	20	25	0.0	S	
levy3	2	0	4	10	−23.24	17	35	−28.48	N	5	9	−43.03	N	
osborne1	5	0	17	78	0.03	515	1444	0.03	N	148	214	0.0	N	

(*continued*)

Table 1. Continued.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
powell	4	0	16	19	0	22	43	0.0	S	59	67	0	N	
price	2	0	6	13	0	70	139	0	N	25	32	0	N	
s324	2	2	10^3	19962	5	15	29	4.99	S	13	23	5	S	*b*)
schwefel	5	0	9	10	0	9	17	0	N	34	36	0	N	
shekel	4	0	13	57	−2.63	19	37	−2.68	N	24	57	−10.15	N	
tre	2	0	4	5	0	14	27	0	S	7	11	0	N	
weapon	100	12	11	12	−1735.57	23	45	−1735.57	N	54	64	−1735.57	N	
fir_convex	11	243	10^3	20^3	1.05	40	79	1.05	N	4	6	1.05	S	*b*)
fir_exp	12	244	1	2	1374.3	43	85	1.05	N	5	6	1.05	N	
fir_linear	11	243	1	2	0.05	18	35	0.05	S	1	2	0.05	S	
fir_socp	12	244	4	26	10.38	56	111	1.05	N	4	5	1.05	N	
fermat_socp_eps	5	3	10^3	20^4	7.5	11	21	7.5	S	9	10	7.5	S	*b*)
steiner_nonconvex	33	17	10^3	19728	0	10^3	16718	25.4	N	53	200	0	N	
steiner_socp_eps	33	17	10^3	20^4	25.4	27	53	25.4	S	55	107	25.4	S	*b*)
steiner_socp_vareps	33	17	10^3	20^4	25.4	64	359	25.4	N	82	190	25.4	S	*b*)
hs001	2	0	24	33	0	32	63	0	S	17	20	0	S	
hs002	2	0	10^3	19887	4.94	21	41	4.94	S	9	14	0.05	N	*b*)
hs003	2	0	1	1	0	11	21	0.0	S	3	8	0	S	*c*)
hs004	2	0	2	1	2.67	8	15	2.67	S	1	2	2.67	S	
hs005	2	0	4	25	1.23	13	25	−1.91	N	7	13	1.23	S	
hs011	2	1	10^3	16256	−24.96	13	25	−8.5	N	8	12	−8.5	N	
hs012	2	1	10^3	17966	−34.78	10	19	−30	S	8	9	−30	N	
hs015	2	2	1	1	306.5	31	61	306.5	S	3	5	306.5	S	

hs016	2	2	4	5	23.15	18	35	0.25	N	4	5	23.15	S	
hs017	2	2	20	29	1.0	29	58	1	S	12	15	1	S	
hs018	2	2	15	15	5	15	29	5	S	13	23	5	S	
hs019	2	2	2	17	−7972.99	17	33	−6961.81	N	6	8	−6961.81	N	*d*)
hs020	2	3	5	6	40.2	16	31	40.2	S	4	5	40.2	S	
hs021	2	1	1	12	−99.96	12	23	−99.96	S	2	4	−99.96	S	*c*)
hs022	2	2	9	9	1	9	17	1	S	5	7	1	S	
hs023	2	5	13	51	2	13	26	9.472	N	7	16	9.472	N	*d*)
hs024	2	2	3	4	−1	23	45	−1	S	1	4	−1	S	
hs025	3	0	25	45	0	15	30	0	S	0	1	32.835	S	
hs029	3	1	10^3	10	−3.973	10	19	−22.627	N	12	15	−22.627	N	
hs030	3	1	2	9	1	9	17	1	S	2	4	1	S	
hs031	3	1	7	17	6	17	33	6	S	8	12	6	S	
hs033	3	2	7	8	−4.586	20	39	2	N	7	8	−4.586	S	*d*)
hs034	3	2	10^3	14	−2.003	14	27	−0.834	N	7	8	−0.834	N	
hs035	3	1	1	10	0.111	10	19	0.111	S	5	7	0.111	S	*c*)
hs036	3	1	2	16	−3300	16	35	−3300	S	1	2	−3300	S	
hs037	3	1	10^3	11	−3455.9	11	21	−3455.9	S	6	9	−3456	S	*b*)
hs038	4	0	23	27	0	25	50	0	S	26	35	0	S	
hs043	4	3	10^3	11	−48.126	11	21	−44	N	11	17	−44	N	
hs044	4	6	6	16	−15	16	31	−15	S	4	6	−15	S	
hs045	5	0	3	23	1	23	45	1	S	0	1	2	N	
hs059	2	3	7	22	−6.75	22	47	−7.803	N	13	17	−6.75	S	

(*continued*)

Table 1. Continued.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
hs064	3	1	10^3	27	6299.84	27	53	6299.84	S	29	39	6299.84	S	*b*)
hs066	3	2	10^3	15	0.518	15	29	0.518	S	7	8	0.518	S	*b*)
hs072	4	3	1	23	−4.681	23	45	727.679	N	28	34	727.679	N	*d*)
hs076	4	3	1	11	−4.681	11	21	−4.681	S	7	8	−4.681	S	*c*)
hs083	5	3	4	13	−30665.5	13	25	−30665.5	N	5	7	−30665.5	S	
hs084	5	3	3	4	−5280335	20	39	−5280340	S	2	3	−5280335	S	
hs086	5	6	3	4	−32.35	15	29	−32.35	S	5	7	−32.35	S	
hs089	3	1	10^3	28	1.363	28	58	1.363	N	48	89	1.363	S	*b*)
hs095	6	4	2	16	9.745	16	32	0.016	N	1	2	0.016	N	
hs096	6	4	4	45	0.462	47	98	0.016	N	1	2	0.016	N	
hs097	6	3	9	18	4.616	18	36	4.071	N	3	6	3.136	N	
hs098	6	3	3	45	9.081	45	137	3.136	N	3	6	3.136	N	
hs100	7	4	10^3	11	680.63	11	22	680.63	S	14	29	680.63	S	*b*)
hs102	7	6	25	55	2736.71	55	208	911.88	N	119	445	911.88	N	
hs103	7	6	17	129	2185.87	48	136	543.67	N	75	285	543.67	N	
hs104	8	6	10^3	14	3.95	14	27	3.95	S	18	20	3.95	N	*b*)
hs106	8	6	630	11220	20589.2	46	97	7049.3	N	17	21	7049.2	N	
hs116	9	13	10^3	38335	50	74	194	97.6	N	10	20	97.6	N	
hs117	15	5	11	11	32.35	34	67	32.35	N	56	65	32.35	N	
hs118	15	17	1	2	664.82	22	43	664.82	S	14	28	664.82	S	*c*)
s215	2	1	3	25	0	25	49	0	S	6	8	0	N	
s218	2	1	14	14	0	11	21	2.38	N	26	29	0	S	*d*)
s221	2	1	19	261	−1.0	261	531	−1.0	N	28	31	−1.0	S	

s222	2	1	61	10	−1.5	10	19	−1.5	S	4	6	−1.5	S	
s223	2	2	6	12	0	12	23	−0.83	N	8	9	−0.83	N	
s224	2	2	2	79	−304	12	23	−304	S	2	3	−304	S	
s225	2	5	8	17	2	17	33	2	S	6	7	2	S	
s226	2	2	2	9	0	9	17	−0.5	N	8	10	−0.5	N	
s227	2	2	7	101	1	9	17	1	S	6	8	1	S	
s229	2	0	19	28	0	27	56	0	S	32	41	0	S	
s230	2	2	10^3	9	0.375	9	17	0.375	S	5	7	0.375	S	*b*)
s231	2	2	21	32	0	32	63	0	S	14	19	0	S	
s232	2	2	2	12	−1	12	23	−1	S	2	5	−1	S	
s233	2	1	7	13	0	13	25	0	S	26	47	2.1	N	
s234	2	1	13	18	−0.8	18	35	−0.8	S	0	1	−0.8	S	
s236	2	2	2	17	−58.9	17	33	−58.9	S	7	10	−58.9	S	*c*)
s237	2	3	2	46	−58.9	46	99	−57.9	S	13	22	−58.9	S	*c*)
s238	2	3	6	10^3	−58.9	10^3	11635	−818.8	N	8	11	−58.9	S	
s239	2	1	2	15	−58.9	15	30	−58.9	S	16	24	−8.2	N	
s242	3	0	2	25	0	25	51	0.0	N	10	13	0	N	
s244	3	0	9	31	0	22	43	0.0	S	11	22	0	S	
s249	3	0	8	10	0	10	19	1	N	19	20	1	N	
s250	3	1	2	16	−3300	16	35	−3300	S	1	2	−3300	S	
s251	3	1	10^3	11	−3456	11	21	−3456	N	6	9	−3456	S	*b*)
s253	3	1	3	15	69.3	15	29	69.3	S	7	8	69.3	S	*c*)
s257	4	0	10	11	0	26	51	0	S	25	36	0	S	*c*)
s259	4	0	9	16	−8.5	12	23	−8.5	S	21	37	3.9	N	
s268	5	5	1	27	0	27	53	0.0	S	11	15	0	S	*c*)
s270	5	1	12	14	−1	17	33	0.0	N	5	8	−1	S	*d*)

(*continued*)

Table 1. Continued.

			Filter IR			LOQO				NPSOL				
Problem	n	m	$\#it$	$\#f$	f	$\#it$	$\#f$	f	SS	$\#it$	$\#f$	f	SS	Note
s277	4	4	10^3	13	5.1	13	25	5.1	S	5	12	5.1	S	*b*)
s278	6	6	10^3	13	7.8	13	25	7.8	S	10	26	7.8	N	*a*)
s279	8	8	10^3	14	10.6	14	27	10.6	S	14	30	10.6	S	*a*) & *b*)
s280	10	10	10^3	16	13.4	16	31	13.4	N	12	25	13.4	N	*a*)
s307	2	0	10	14	124.4	14	27	124.4	S	15	28	124.4	S	*b*)
s315	2	3	10^3	14	−0.2	14	27	−0.8	N	18	22	−0.8	N	
s326	2	2	412	12	−79.8	12	23	−79.8	S	6	9	−79.8	S	
s328	2	0	5	22	1.74	22	43	1.7	S	10	15	1.7	S	*c*)
s329	2	3	120	3452	−6961.8	18	37	−6961.8	S	8	11	−6961.8	S	
s331	2	1	24	44	4.3	9	20	4.3	S	E				
s332	2	1	10^3	6643	34.8	10^3	2152	29.4	N	15	105	29.4	N	
s337	3	1	6	11	6	11	21	6	S	7	10	6	S	
s341	3	1	2	11	0	11	21	−22.6	N	12	15	−22.6	N	
s342	3	1	3	23	0	23	45	−22.6	N	112	213	−22.6	N	
s343	3	2	1	27	−0.0	27	53	−5.7	N	5	9	−5.7	N	
s346	3	2	1	27	−0.000000	27	53	−5.68478	N	5	9	−5.7	N	
s354	4	1	10^3	16	0.11	16	32	0.11	S	17	25	0.11	S	*b*)
s357	4	35	6	8	0.4	21	42	0.4	S	21	22	0.4	S	*c*)
s358	5	0	781	9281	0.0	141	281	73783200	N	42	62	0.0	S	*d*)
s359	5	14	1	17	−5504451	17	33	−5504450	S	1	2	−5504451	S	
s360	5	2	10^3	28	−5273583	28	92	−5280340	S	2	3	−5280335	S	*b*)
s361	5	6	2	27	−15260.2	27	66	−15260.2	S	3	4	−15260.2	S	
s366	7	14	10^3	47	1226.972633	47	95	1226.97	S	9	11	1226.97	S	*b*)
s368	8	0	2	22	0	20	41	−0.9375	N	1	1	0	N	
s372	9	12	10^3	32	13390.1	32	65	13390.1	S	336	942	23273.3	N	*b*)

(m). For each code are reported the iterations count ($\#it$), the function evaluations count ($\#f$) and the objective function (f). The SS column checks if both codes have the same solution and the $Note$ is dedicated to remarks.

Problems noted by a) identifies the cases of NPSOL solver that were locked in the iterative process.

The comparison with NPSOL and LOQO shows that the problems that do not converge have the same optimal solution as both solvers or at least one of them. These problems are denoted by b). Note that most of these cases were identified in [17] as problems suffering from Marato's effect.

Another interesting remark, denoted by c), is related to the problems with the same solution for the three solvers, but the Filter IR reached the solution in less than half the iterations and in most of these cases the difference between the number of iterations is greater than double.

Comparatively, numeric results approach the ones obtained by NPSOL and 34% of cases have the same solution with less or the same number of iterations. Filter IR performance increases when compared with LOQO, 41% of problems have the same solution, but Filter IR has less iterations.

Finally, it was noted, by d), the problems where Filter IR has a better solution in terms of f than both solvers.

Dolan and Moré in [19] present a tool for the evaluation and performance of optimization codes. The performance profile for a solver is the (cumulative) distribution function for a performance metric. Performance profiles provide a means of visualizing the expected performance difference among solvers, while avoiding arbitrary parameter choices and need to discard solver failures from the performance. This comparison is very interesting when the test set has a large number of problems. Graphically it is interpreted as the probability distribution

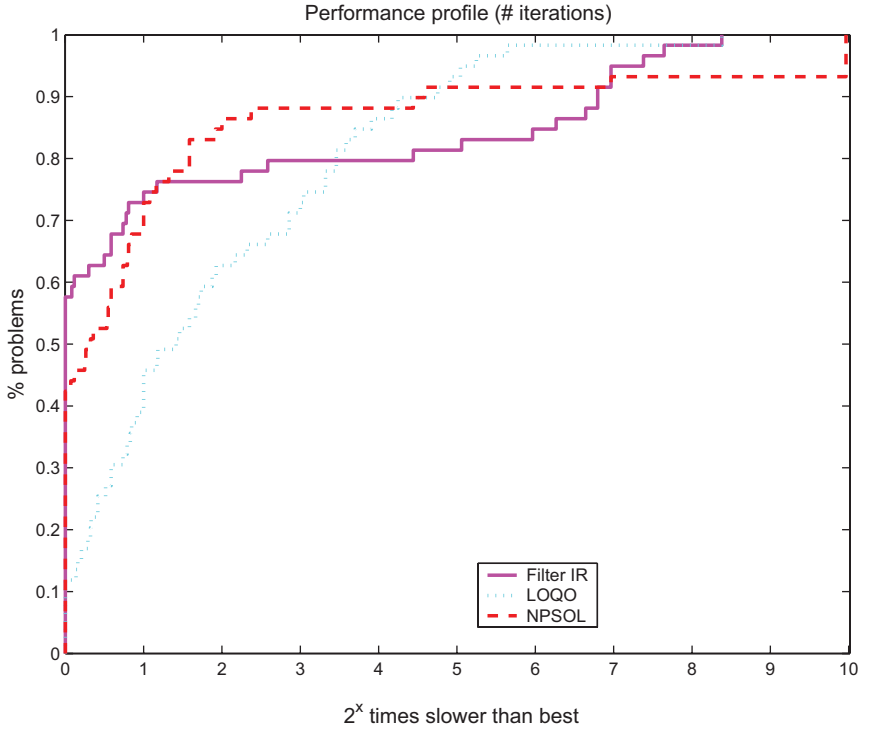

Figure 15. Comparison with LOQO and NPSOL.

that a given solver is at worst x times slower than the best. An easy interpretation of this graphic is that for any given measure a solver is the best when its graphic is tending faster to 1.

The graphic of performance profiles with respect to the number of iterations is presented in figure 15, where a log scale is used. This figure shows that in almost 60% of the problems the Filter IR has the highest probability of being the optimal solver for the number of iterations.

4.2 *Some conclusions and future work*

The algorithm is still in an improvement phase but some conclusions can already be made. With respect to IR philosophy, one can conclude that the feasibility phase speeds up the convergence. The filter method is very simple to implement and replaces the penalty function. The temporary pair creates filters with smaller size, improving its management. The line search strategy in the filter method avoids the resolution of another subproblem when the trial point is not accepted by the filter whereas in the trust-region another subproblem must be solved.

Future work perspectives will be concerned with the algorithm performance, namely, to expand the algorithm for solving equality constraints optimization problems and improvement of the filter management. To study other schemes to estimate the Lagrange multipliers, to introduce second-order corrections to avoid Marato's effect and to implement the filter restoration phase are also ideas to consider. The algorithm evaluation with other performance measurements, such as CPU time, gradient and function counts and its convergence analysis with internal procedures, are other suggestions for future work.

Acknowledgements

This work has been partially supported by project POCTI/MAT/45276/2002, *Nonlinear semi-infinite programming solver*.

References

[1] Fletcher, R. and Leyffer, S., 2002, Nonlinear programming without a penalty function. *Mathematical Programming*, **91**, 239–270.
[2] Martínez, J.M. and Pilotta, E.A., 2000, Inexact-restoration algorithm for constrained optimization. *Journal of Optimization Theory and Applications*, **104**, 135–163.
[3] Martínez, J.M. and Pilotta, E.A., 2005, Inexact restoration methods for nonlinear programming: advances and perspectives. *Optimization and Control with Applications*, in the series *Applications in Optimization*, Vol. 96 (New York: Springer), pp. 271–292.
[4] Birgin, E.G. and Martínez, J.M., 2005, Local convergence of an inexact-restoration method and numerical experiments. *Journal of Optimization Theory and Applications*, **127**, 229–247.
[5] Martínez, J.M., 2001, Inexact-restoration method with lagrangian tangent decrease and new merit function for nonlinear programming. *Journal of Optimization Theory and Applications*, **111**, 39–58.
[6] Gonzaga, C.C., Karas, E. and Vanti, M., 2003, A globally convergent filter method for nonlinear programming. *SIAM Journal on Optimization*, **14**, 646–669.
[7] Antunes, A.S. and Monteiro, M.T.T., 2004, Paper presented at the XXVIII Congreso Nacional de Estatística e Investigación Operativa, Cádiz, 25–29 October.
[8] Wächter, A. and Biegler, L.T., 2006, On the implementation of a interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, **106**, 25–57.
[9] Wächter, A. and Biegler, L.T., 2005, Line search filter methods for nonlinear programming: local convergence. *SIAM Journal Optimization*, **16**, 32–48.
[10] Wächter, A. and Biegler, L.T., 2005, Line search methods for nonlinear programming: motivation and global convergence. *SIAM Journal of Optimization*, **16**, 1–31.
[11] Chin, C.M., 2002, A global convergence theory of a filter line search method for nonlinear programming. *Numerical Optimization Report*.
[12] Chin, C.M., 2003, A local convergence theory of a filter line search method for nonlinear programming. *Numerical Optimization Report*.

[13] Gill, P.E., Hammarling, S.J., Saunders, M.A. and Wright, M.H., 1986, User's guide for LSSOL: a fortran package for constrained linear least-squares and convex quadratic programming. Technical Report 86-1, Systems Optimization Laboratory, Department of Operations Research, Stanford University.
[14] Gill, P.E., Murray, W., Saunders, M.A. and Wright, M.H., 1986, User's guide for NPSOL: a fortran package for nonlinear programming. Technical Report 86-2, Systens Optimization Laboratory, Department of Operations Research, Stanford University.
[15] Fourer, R., Gay, D.M. and Kernighan, B.W., 1993, *AMPL: A Modelling Language for Mathematical Programming* (Pacific Grove, CA: Duxburg Press).
[16] Gay, D.M., 1997, Hooking your solver to AMPL. Technical Report 97-4-06, Computing Sciences Research Center, Bell Laboratories.
[17] Silva, C.E.P. and Monteiro, M.T.T., 2006, A filter inexact-restoration method for nonlinear programming. *TOP*, submitted.
[18] Vanderbei, R.J., 1999, LOQO user's manual, version 4.05. Technical Report ORFE-99, Operations Research and Financial Engineering, Princeton University.
[19] Dolan, E.D. and Moré, J.J., 2001, Benchmarking optimization software with performance profiles. *Mathematical Programming A*, **91**, 201–213.