# SIAG/OPT Views-and-News

A Forum for the SIAM Activity Group on Optimization

Volume 18 Number 1 March 2007

## Contents

# Filter Methods and Optimization Software

## Introduction by the Editors

Roger Fletcher, Sven Leyffer, and Philippe Toint were the recipients in 2006 of the Lagrange Prize for Continuous Optimization, awarded jointly by the Mathematical Programming Society (MPS) and the Society for Industrial and Applied Mathematics (SIAM), for *"outstanding works in the area of continuous optimization"*. Congratulations!

The citation of the award can be read in the MPS website. We quote here the first paragraph:

> *"In the development of nonlinear programming over the last decade, an outstanding new idea has been the introduction of the filter. This new approach to balancing feasibility and optimality has been quickly picked up by other researchers, spurring the analysis and development of a number of optimization algorithms in such diverse contexts as constrained and unconstrained nonlinear optimization, solving systems of nonlinear equations, and derivative-free optimization. The generality of the filter idea allows its use, for example, in trust region and line search methods, as well as in active*

*set and interior point frameworks. Currently, some of the most effective nonlinear optimization codes are based on filter methods. The importance of the work cited here will continue to grow as more algorithms and codes are developed."*

See `http://www.mathprog.org/prz/citations/lagrange_2006.htm`.

We invited Roger, Sven, and Philippe to express their own views on the relevance of filter methods. Their contribution arrived in the format of an excellent expository paper, which we are extremely pleased to publish in the current issue of SIAG/OPT Views-and-News.

The purpose of this issue is, however, twofold. The other two included articles address the important issue of how optimization software, in a sense the "fruit" of our research, can be made available for easy use to a wide audience of practitioners. We are indebted to Jason Sarich and Hans Mittelmann, respectively, who kindly accepted our invitation to discuss two successful free Internet resources. These resources help non-experts as well as professionals to get acquainted to and use state-of-the-art optimization codes.

The first paper surveys the current status of NEOS (Network-Enabled Optimization System). NEOS is a successful project, providing free and easy access to a variety of optimization codes, and has become an indispensable tool for optimization practitioners and researchers.

The other article is about the Decision Tree for Optimization Software (DTOS), also a popular Internet site for users of optimization solvers. The paper walks us through the resources DTOS provides, both for optimization "beginners" and experts.

**Andreas Wächter and Luís N. Vicente**, February 2007.

# A Brief History of Filter Methods

**Roger Fletcher**
Department of Mathematics, University of Dundee, Dundee, DD1 4HN, UK (`fletcher@maths.dundee.ac.uk`).

**Sven Leyffer**
Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, USA (`leyffer@mcs.anl.gov`).

**Philippe Toint**
Department of Mathematics, The University of Namur (FUNDP), rue de Bruxelles, B5000 Namur, Belgium (`philippe.toint@fundp.ac.be`).

## 1. Motivation

We consider the question of global convergence for optimization algorithms that solve general nonlinear programming problems (NLPs):

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & c(x) \geq 0, \end{array} \tag{1}$$

where the objective function $f(x)$ and the constraint functions $c(x)$ are smooth.

Most methods for solving (1) are based on Newton's method and are iterative. Given an estimate $x_k$ of the solution $x_*$ of (1), a linear or quadratic approximation of (1) is solved for a new and, one hopes better, estimate $x_{k+1}$. Near a solution, this process is guaranteed to converge. Far from the solution, however, the sequence $\{x_k\}$ generated in this way may not converge. How can we ensure convergence even if we start far from a solution? We refer to this question as *global convergence* for NLP methods.

Traditionally, this question has been answered by using penalty or merit functions that are a linear combination of the objective function and a measure of the constraint violation such as $h(x) := \|c(x)^-\|$, where $\|a^-\| = \|\min(a, 0)\|$ for some norm. An example is the $\ell_1$ exact penalty function,

$$p(x; \pi) := f(x) + \pi h(x),$$

where $\pi > 0$ is the penalty parameter. Provided $\pi$ is sufficiently large, we can use this penalty function to ensure progress in our iterative scheme by enforcing sufficient decrease on each step. This trick allows us to invoke well-developed unconstrained optimization techniques.

In traditional penalty methods, a suitable penalty parameter depends on the solution of (1), namely, $\pi > \|y_*\|_D$, where $y_*$ are the optimal multipliers and $\|\cdot\|_D$ is the dual norm. This fact may make it difficult to find a suitable penalty parameter. Worse, if the penalty parameter is too large, then any monotonic method would be forced to follow the nonlinear constraint manifold very closely, resulting in much shortened Newton steps and slow convergence. Yet we have noticed that the unmodified sequential quadratic programming (SQP) method is able to quickly solve a large proportion of test problems without the need for modifications to induce global convergence.

In this paper we review a recent alternative to penalty functions, so-called filter methods. The success of the unmodified SQP method motivates us to find a way of inducing global convergence, which would allow the full Newton step to be taken much more often. Our goal therefore is the development of global optimization safeguards that *interfere as little as possible with Newton's method.* We believe filter methods achieve this goal. In the remainder of this paper, we motivate filter methods, outline the main ideas and convergence results, indicate other areas where filter methods have been used successfully, and provide references for those wishing to delve deeper into filter methods.

## 2. Filter methods for NLP

Filter methods avoid the pitfalls of penalty function methods. Instead of combing the objective and constraint violation into a single function, we view (1) as a biobjective optimization problem that minimizes $f(x)$ and $h(x)$. However, the second objective is clearly more important because we must ensure that $h(x_*) = 0$. We borrow the concept of domination from multiobjective optimization and say that a point $x_k$ *dominates* a point $x_l$ if and only if $f(x_k) \le f(x_l)$ and $h(x_k) \le h(x_l)$. We define a *filter* as a list of pairs $\big(h(x_l), f(x_l)\big)$ such that no pair dominates another pair. A typical filter is illustrated in Figure 1, where the shaded area shows the region dominated by the filter entries. The contours of the $\ell_1$ exact penalty function would be straight lines with slope $-\pi$ in this plot, indicating that at least for a single entry, the filter is less restrictive

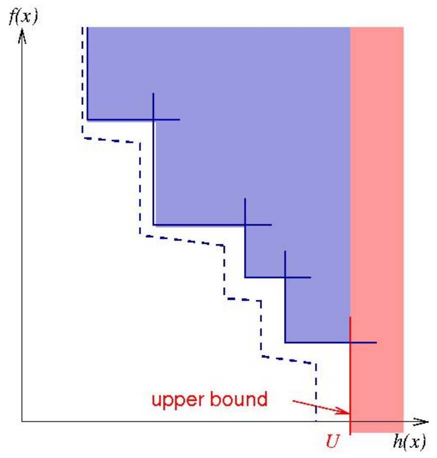

Figure 1: A typical filter. All pairs $(f(x), h(x))$ that are below and to the left of the envelope (dashed line) are acceptable to the filter

than penalty methods.

### 2.1 SQP filter methods

Filter methods were first introduced in the context of trust-region SQP methods, which solve a quadratic approximation of (1) for a trial step $s$ that lies inside a trust region:

$$\begin{array}{ll} \min\limits_{s} & q_k(s) := f_k + \nabla f_k^T s + \frac{1}{2} s^T H_k s \\ \text{s.t.} & c_k + \nabla c_k^T s \ge 0 \\ & \|s\|_\infty \le \rho_k, \end{array} \qquad (1)$$

where $f_k = f(x_k)$ and so on, and $H_k \simeq \nabla^2 \mathcal{L}_k$ approximates the Hessian of the Lagrangian.

A rough outline of a filter trust-region SQP is as follows. At iteration $k = 0$, we initialize the filter $\mathcal{F}_k = \{(U, -\infty)\}$, where $U$ is an upper bound on the acceptable constraint violation. We proceed by accepting only steps that are not dominated by the current filter. If a point is acceptable, then we set $x_{k+1} = x_k + s$, and possibly increase the trust-region radius and update the filter (adding $(h_k, f_k)$ from the previous iterate and removing any dominated entries). If, on the other hand, the step is dominated by the current filter, then we reject it, set $x_{k+1} = x_k$, reduce the trust-region radius, and resolve (1).

This simple description of a filter method requires a number of refinements to ensure convergence:

1. *Filter Envelope.* To avoid convergence to infeasible limit points where $h_* > 0$, we add an en-

velope around the current filter. A new iterate is acceptable if, for all $\forall (h_l, f_l) \in \mathcal{F}_k$,

$$h_{k+1} \leq \beta h_l, \quad \text{or} \quad f_{k+1} \leq f_l - \gamma h_{k+1}, \qquad (2)$$

where $0 < \beta, \gamma < 1$ are constants. This *sloping envelope* is due to Chin [4, 3] and makes the management of redundant entries slightly more convenient. In [4, Lemma 1] it is shown that if an infinite number of points are added to the filter and $f(x)$ is bounded below, then the limit point must be feasible. We note, that this result does not require the presence of an upper bound $U$.

2. *Sufficient Reduction.* The filter alone cannot ensure convergence to stationary points. For example, if the sequence satisfies $h_{k+1} \leq \beta h_k$, then the iterates could converge to an arbitrary feasible point. Therefore, if the constraint violation becomes small, we enforce a sufficient reduction condition similar to unconstrained optimization. We denote the predicted reduction by $\Delta q_k := -\nabla f_k^T s - \frac{1}{2} s^T H_k s$ and introduce the following *switching condition*:

$$\begin{array}{ll} \text{if} \ \ ( \ \Delta q_k > 0 \ ) \ \ \text{then} & \\ \quad \text{check} \ \ f_k - f_{k+1} \geq \sigma \Delta q_k, & \end{array} \qquad (3)$$

where $\sigma \in (0,1)$ is a constant.

3. *Feasibility Restoration.* By reducing the trust-region radius, the QP (1) may become inconsistent (halving the trust-region radius in the right plot of Figure 2 illustrates this point). We take the inconsistency of (1) as an indication that the current point is too far from the feasible set to make meaningful progress to optimality. Hence we invoke an SQP-like algorithm that minimizes the constraint violation $h(x)$ (see Section 3.1). We exit the restoration phase once a filter-acceptable point has been found and resume the regular SQP method.

With these modifications, we can define acceptance for a filter method.

**Definition 1** *A trial point $x_k^+ := x_k + s$ is* acceptable to the filter *at iteration $k$ if*

1. *$x_k^+$ is acceptable to the filter $\mathcal{F}_k$ and $x_k$, that is, (2) holds for $\mathcal{F}_k \cup \{(h_k, f_k)\}$, and*
2. *if the switching condition $\Delta q_k > 0$ holds, then we have sufficient reduction, that is, $f_k - f(x_k^+) \geq \sigma \Delta q_k$.*

*Otherwise, we call $x_k^+$* not acceptable.

An outline of a filter method is given next.

**Algorithm 1**: SQP Filter Method

```
$x_0$, $k \leftarrow 0$, $\mathcal{F}_0 \leftarrow \{U, -\infty\}$, optimal ← false
while not optimal do
    reset the trust-region radius: $\rho_k \geq \underline{\rho}$
    terminate ← false
    repeat
        solve the QP (1) for a step $s$
        if $s = 0$ then
            optimal ← true; STOP
        if QP (1) incompatible then
            add $(h_k, f_k)$ to $\mathcal{F}_k$
            enter restoration phase
        else
            if $x_k^+ := x_k + s$ not acceptable then
                reduce trust-region $\rho_k \leftarrow \rho_k/2$
            else
                terminate ← true
    until terminate
    update the filter $\mathcal{F}_{k+1}$
    set $x_{k+1} \leftarrow x_k + s$ and $k \leftarrow k + 1$
```

## Discussion of filter algorithm

Algorithm 1 contains an inner and an outer iteration. During the inner iteration the trust-region radius is reduced until we either find an acceptable point or enter the restoration phase. The aim of the restoration phase is to find an acceptable iterate $x_{k+1}$ such that the corresponding QP (1) is compatible for some $\rho_{k+1} \geq \underline{\rho}$. The iterates and the filter are updated in the outer iteration, which also ensures that the trust-region radius is larger than a lower bound $\underline{\rho} > 0$.

We update the filter by adding entries $(h_k, f_k)$ to $\mathcal{F}_k$ that correspond to an h-type iteration *after we move to $x_{k+1}$*. We can also remove any entries that are dominated by $(h_k, f_k)$.

The switching condition (3) can be motivated as follows. Close to a feasible point, we expect the quadratic model to predict a decrease in the objective function, that is, $\Delta q_k > 0$. However, far from a feasible point, the predicted reduction is sometimes negative, that is $\Delta q_k < 0$, because most of the SQP step is toward feasibility. We will refer to successful steps that satisfy (3) as *f-type steps* and all other steps as *h-type* steps. This is illustrated in Figure 2: the left plot shows an f-type step that reduces $q_k(s)$, while the right plot shows an h-type step that reduces only infeasibility. We note that if $h_k = 0$ at a nonstationary point, then $\Delta q_k > 0$, thereby implying that we can accept only an f-type step. Thus, we never add points to the filter for which $h_k = 0$. This fact ensures that we can always generate a filter-acceptable point during the restoration phase unless the problem is (locally) infeasible.

## Early history of filter methods

NLP filter methods were first proposed by Fletcher in a plenary talk at the SIAM Optimization Conference in Victoria in May 1996; the methods are described in [8]. The initial filter method contained features, such as the NW/SE corner rule and unblocking, that were shown to be redundant in the subsequent convergence analysis. The first global convergence proof of a filter method was given in [11] for a sequential linear programming (SLP) method. This proof was later generalized to SQP methods in [10].

Filter methods for NLP were developed independently of earlier similar ideas. Surry *et al.* [25] describe a multiobjective approach to constrained optimization in the context of genetic algorithms. The algorithm maintains a population of iterates that are evolved over time. The authors modify a vector-evaluated genetic algorithm to adaptively bias the population toward feasibility.

An idea similar to a filter was used by Lemaréchal *et al.* [20] to enforce convergence of a bundle method for convex nonsmooth constrained optimization problems. The authors define an exclusion region that corresponds to the *convex hull* of the filter entries (with the two out-most entries extended to infinity).

## Convergence proof outline

Convergence of filter methods can be established under the following general assumptions: the iterates $x_k$ lie in a compact set $X$, the functions $f(x)$ and

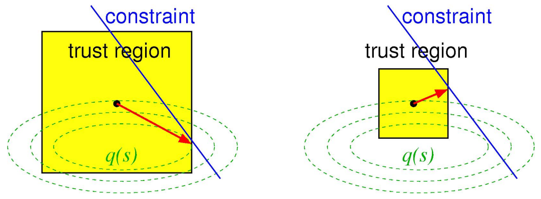

Figure 2: Illustration of f-type and h-type step.

$c(x)$ are twice continuously differentiable, and the Hessian remains bounded $\|H_k\| \le M$. Under these assumptions, one of the following occurs (see [10, Theorem 7], [4, Theorem 1], and [27, Theorem 3]):

1. The restoration phase fails to find a filter-acceptable point for which the QP (1) is consistent for some $\rho \ge \underline{\rho}$.
2. The algorithm terminates at a first-order stationary point.
3. There exists a feasible accumulation point that either is stationary or the Mangasarian-Fromowitz constraint qualification fails.

These results are as strong as can be expected for general NLPs. For example, the first outcome corresponds to a situation where the restoration phase has converged to a local minimum of the constraint violation. One undesirable assumption in [10] is the need for global solution to the QP subproblem (1). This assumption may be difficult to ensure, unless $H_k$ is positive semi-definite.

The convergence proof makes use of the insights from Figure 2. The filter ensures that all limit points are feasible. Next, we consider two cases: (a) an infinite subsequence of h-type iterations, and (b) an infinite sequence of f-type steps. We assume that the limit point is not stationary and seek a contradiction. In case (a), we can show that for sufficiently large $k$ we must generate an f-type iteration, which contradicts the construction of the sequence. In case (b), we obtain the usual contradiction that $f(x_k)$ is unbounded below.

## Fast local convergence

The transition of filter methods to fast local convergence had been an outstanding issue from the start. Early on, we conjectured that filter methods may be able to avoided the Maratos effect. This effect causes penalty function SQP methods to reject the full SQP step arbitrarily close to a solution, leading to a loss of second-order convergence. We applied filter methods to the original example by Maratos and observed second-order convergence. However, the following example shattered the hope that filter methods can avoid the Maratos effect in general:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & 2(x_1^2 + x_2^2 - 1) - x_1 \\ \text{subject to} & x_1^2 + x_1^2 - 1 = 0. \end{array}$$

The starting point $x = (\cos(t), \sin(t))$ for $t > 0$ small and multipliers $y = 3/2$ shows that the SQP step increases *both* $f(x)$ and $\|c(x)\|$, leading to a filter-rejected step. This example motivated us to include second-order correction (SOC) steps. Since then, Ulbrich [27] and Wächter and Biegler [29] have considered the transition to fast local convergence.

Ulbrich [27] proves fast local convergence *without the use of SOC steps* by making three modifications to the filter SQP method: (1) he uses the augmented Lagrangian as a technical tool, which motivates an alternative definition of the filter, replacing $h(x)$ and $f(x)$ by

$$\theta(x,y) := \|c(x)^-\|_2^2 + \left(y^T c(x)^-\right)^2 \text{ and } \mathcal{L}(x,y)$$

respectively; (2) the switching condition (3) is tightened to

$$\Delta\widehat{q_k} := \Delta q_k + y_k^T s_k > \kappa\theta_k^{\psi/2} \text{ and } \Delta\mathcal{L}_k(s) < \sigma\Delta\widehat{q_k},$$

where $\psi \in (\frac{1}{2}, 1)$ is a constant; and (3) the restoration phase is also entered if the multiplier weigh inactive constraints too strongly, which happens, if $\theta_k^{1/2} \le \kappa_\rho \rho^{1+\xi}$, for $\kappa_\rho > 0$ and $\xi \in (0,1)$. Under a linear independence constraint qualification and second-order sufficient condition, Ulbrich is able to show q-quadratic convergence.

Wächter and Biegler [29] analyze a filter method with SOC steps. SOC steps solve a second QP that captures constraint curvature and is often cheap to solve, requiring, for example, only a shift in the QP constraints. Like [27], Wächter and Biegler also modify the switching condition and strengthen it to

$$\nabla f_k^T s_k < 0 \text{ and } \alpha_{k,l}\left(-\nabla f_k^T s_k\right)^{s_f} > \delta(h_k)^{s_h},$$

where $\alpha_{k,l}$ is the Armijo step size and $\delta > 0, s_h > 1$, and $s_f \ge 2s_h$ are constants. Thus, sufficient reduction in the objective is checked less frequently than in [10]. The analysis shows that ultimately, the SQP step or the SOC step is acceptable to the filter implying superlinear convergence for different types of SOC steps.

We currently prefer to use SOC steps to obtain fast local convergence because this approach allows us to keep the original filter definition with $f(x)$, rather than the Lagrangian $\mathcal{L}(x,y)$. This approach also avoids the need for a multiplier function.

## Other SQP filter methods

Fletcher *et al.* [6] (see also [5, Chapter 15.5]) analyze a trust-region SQP filter method that decomposes the SQP step into a normal and tangential step. The normal step attains feasibility for the linearized constraints of (1), and the tangential step reduces a quadratic objective beyond the Cauchy point while maintaining feasibility. The algorithm uses the envelope

$$h_{k+1} \le \beta h_j \text{ or } f_{k+1} \le f_j - \gamma h_j \ , \ \forall j \in \mathcal{F}_k,$$

and removes only entries whose *envelope* is dominated by a new entry. The algorithm also uses a stronger switching condition, namely, $\Delta q_k \ge \sigma h_k^2$, resulting in fewer f-type steps. We note that the algorithm in [6] removes the need for a global solution of the QP.

Recently, there has been renewed interest in trust-region methods that avoid the solution of the computationally expensive QP (1). One such method is SLP-EQP, which dates back to [12] in the context of $\ell_1$-penalty functions. The method solves an LP inside a trust-region to obtain an estimate of the active set (i.e., setting $H_k = 0$ in (1)). This active set is then explored further by solving an equality-constrained QP corresponding to the active constraints (with the trust-region bounds removed). Chin and Fletcher [4] (see also [3]) analyze and implement a filter SLP-EQP method. Their convergence proof adapts the proof in [11] to allow for a finite set of possible steps, namely, a Cauchy step (along the LP solution to the first minimum of the quadratic), an EQP step, and an SOC step.

Gonzaga *et al.* [14] propose a general framework for filter methods where the step computation is decomposed into a normal and tangential step. Unlike [6], however, where the QP solution is decomposed, Gonzaga *et al.* enforce filter conditions on both the normal and tangential step. The normal step must generate an intermediate point $x_{k+1/2}$ such that the constraint violation is acceptable to the current point: $h(x_{k+1/2}) < \beta h_k$. The tangential step must generate a new iterate that reduces the objective function by an amount that is proportional to the *filter slack*:

$$H_k := \min\big(1, \min_{j\in\mathcal{F}_k: f_j \le f(x_k)} h_j\big).$$

The step $s$ satisfies $\nabla c_{k+1/2}^T s + c_{k+1/2} \ge 0$, and the new point, $x_{k+1} = x_{k+1/2} + s$, satisfies the following decrease condition:

$$f(x_{k+1}) \le f(x_{k+1/2}) - M\sqrt{H_k}.$$

The authors show that such a step can be computed by minimizing a quadratic model beyond the Cauchy point within a trust-region framework. In addition, a sufficient decrease condition is also enforced. This framework is very general, but the step acceptance seems slightly more restrictive.

Ribeiro *et al.* [22] extend the analysis in [14] by developing a general global convergence analysis of filter methods that does not depend on the particular way in which the step is computed. Instead, the authors prove convergence under fairly general assumptions that are shown to hold, for example, for SQP methods.

Finally, a *nonmonotone filter method* based on [6] is analyzed in [16]. The authors measure the area that a new entry contributes to the dominated region. This area is positive for monotone filters. The key idea is to request that this area be positive on average only over the last $K$ reference iterations. This strategy allows the filter to accept points that would otherwise be rejected.

## 2.2 Filter interior methods

Interior-point methods (IPMs) are an attractive alternative to SQP methods for solving NLPs. Instead of computing a step by solving a QP, which can be computationally demanding, IPMs compute a step by solving a linear system. Thus, it is not surprising that researchers have extended filter methods to IPMs: Ulbrich *et al.* [26] and Wächter and Biegler [30] develop convergence theory for IPM filter methods. A related filter criterion has also been used by Benson *et al.* [2].

Interior-point methods first reformulate the NLP (1) so that the inequalities are simple bound constraints:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & c(x) = 0, \quad x \ge 0. \end{array} \tag{4}$$

IPMs can be viewed as applying Newton's method to the perturbed optimality conditions of (4):

$$F_\mu(x,y,z) \;:=\; \begin{pmatrix} \nabla_x \mathcal{L}(x,y,z) \\ c(x) \\ Xz - \mu e \end{pmatrix} = 0, \tag{5}$$

for a decreasing sequence of barrier parameters $\mu \searrow 0$, where $X$ is the diagonal matrix with $x$ along its diagonal, $\mathcal{L}(x,y,z) = f(x) - y^T c(x) - z^T x$ is the Lagrangian of (4), and $e = (1,\ldots,1)^T$. The IPM filter methods differ significantly in how the filter is employed to achieve global convergence.

### The interior filter of Ulbrich *et al.*

Ulbrich *et al.* [26] employ the filter to enforce convergence of the IPM as $\mu \searrow 0$. They decompose the perturbed optimality conditions into a normal ($r^n$) and tangential ($r^t$) component,

$$\begin{aligned} F_{\sigma\mu'}(x,y,z) &= r^n + r^t \\ &= \begin{pmatrix} 0 \\ c(x) \\ Xz - \mu' e \end{pmatrix} + \begin{pmatrix} \nabla_x \mathcal{L}(x,y,z) \\ 0 \\ (1-\sigma)\mu' e \end{pmatrix}, \end{aligned}$$

where $\mu = \mu'\sigma$, and $\mu' = x^T z/n$, and $\sigma \in (0,1)$ is a centering parameter. This decomposition motivates the two filter components and a consistent step decomposition. Denoting $w = (x,y,z)$, the filter is defined a a collection of pairs of a *measure of quasicentrality*

$$\theta(w) := \|c(x)\| + \|Xz - x^T z/n \cdot e\|$$

and a *measure of optimality*

$$\theta_g(w) := \|\nabla_x \mathcal{L}(w)\| + x^T z/n.$$

Each step $s = (s_x, s_y, s_z)$ is computed from a normal and tangential step,

$$F'(w)s^n = -r^n, \; F'(w)s^t = -r^t,$$

where $F'(w)$ is the Jacobian of $F_{\sigma\mu'}(w)$. The authors exploit the flexibility of choosing different step sizes for each component. Once a step has been computed, the algorithm performs a backtracking line search until a filter-acceptable point has been found. Similar to SQP filter methods, the algorithm also enforces a sufficient decrease condition on a quadratic model of the residual of $\theta_g(w)$. If no acceptable point can be found, then a restoration phase is entered to restore quasicentrality.

Under the strong assumption that the inverse of the Jacobian is bounded, namely, $\| \left[F'(w)\right]^{-1} \| \leq C$, the authors show finite termination of the restoration phase and the existence of a subsequence converging to a stationary point. The step decomposition has a similar flavor to [6], but the two components have a slightly different interpretation, with quasicentrality replacing feasibility and the optimality measure $\theta_g(w)$ replacing the objective. The latter condition means that the algorithm may be more likely to converge to stationary points that are not local minimizers. The IPM of Wächter and Biegler avoids this problem by taking a different approach.

### The filter of Wächter and Biegler

Wächter and Biegler [30] have successfully incorporated a filter mechanism in the NLP solver IPOPT [31]. They develop a line-search filter method that avoids the pitfall of many IPMs that may converge to spurious stationary points illustrated by the example in [28]. Wächter and Biegler exploit the relationship between (5) and the barrier problem

$$\begin{array}{ll} \min\limits_{x} & \varphi_{\mu_k}(x) := f(x) - \mu_k \sum \ln(x_i) \\ \text{s.t.} & c(x) = 0. \end{array} \tag{6}$$

IPOPT performs a number of line-search SQP iterations to minimize (6) to within a tolerance $\epsilon_k \searrow 0$, whilst keeping $x_i > 0$. In contrast to [26], where the filter safeguards the convergence of IPM as $\mu \searrow 0$, IPOPT employs the filter only for *fixed* $\mu_k$ to ensure convergence of the SQP algorithm. This approach is justified because it can be shown that for a suitable choice of the sequences $\mu_k \searrow 0$, $\epsilon_k \searrow 0$ one SQP iteration and an extrapolation step are sufficient to generate an acceptable point near a solution.

A consequence of employing the filter for a fixed barrier parameter is that we can now use $(h(x), \varphi(x))$ again in the filter. Hence, the method is less likely to converge to stationary points that are not minimizers.

Another important difference from [26] is the absence of a full-rank assumption, which provides robustness for degenerate and infeasible NLPs. As a consequence, however, we must modify the Armijo line-search because a poor step may never be acceptable no matter how small a step size is chosen. Therefore, [30] derives a lower bound that indicates when the algorithm should switch to a restoration phase. The restoration algorithm in [30] differs from the SQP restoration algorithms in the sense that it must also produce a new point $x_{k+1} \geq \epsilon e$ that is strictly feasible with respect to the bounds.

The analysis in [30] is general and includes as special cases SQP methods, IPMs, and *augmented Lagrangian methods.* The augmented Lagrangian is another popular penalty function:

$$\mathcal{L}_\pi(x, y) := f(x) - y^T c(x) + \frac{\pi}{2} c(x)^T c(x).$$

We can split this function into two "objectives" similar to the way we split the exact penalty function. This motivates a filter method where the Lagrangian $\mathcal{L}(x, y)$ replaces $f(x)$. The analysis is readily extended by including a line search on the multipliers $y$ and by modifying the switching condition in an obvious way. We note, that this is similar to the filter in [27]

Benson *et al.* [2] have also included a filter-like mechanism in LOQO. The filter used in LOQO consists of a *single entry.* We are not sure that this device alone can guarantee convergence. The practical performance of LOQO has been encouraging, however, underlining the computational advantage of filter methods.

## 3. Filters beyond NLP

Filter methods have been extended to other areas of optimization such as nonlinear equations and inequalities [9, 15, 17], nonsmooth optimization [7, 19, 21], unconstrained optimization [18], derivative-free optimization [1], and augmented Lagrangian methods [13].

### 3.1 Nonlinear equations

We have developed a filter SQP method for solving a nonlinear system of inequalities $c(x) \geq 0$ in [9], similar to the restoration phase suggested in [8].

Formulating $c(x) \geq 0$ as a norm minimization problem,

$$\underset{x}{\text{minimize}}\ h(x) := \|c(x)^-\|, \tag{1}$$

allows us to define two objectives and apply the filter concept. We divide the constraints into two sets indexed by $J$ and its complement $J^\perp$: the set $J^\perp$ collects the constraints that are close to being satisfied, and the set $J$ collects the constraints that are difficult to satisfy. This partition gives rise to the following feasibility problem,

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \sum_{i \in J} c_i(x)^- \\ \text{subject to} & c_i(x) \geq 0\ \forall i \in J^\perp, \end{array} \tag{2}$$

which can be interpreted as a weighted $\ell_1$ constraint residual minimization. The sets $J$ and $J^\perp$ can be chosen adaptively as long as we ensure that

$$J \subset \{i | c_i(x) < 0\}.$$

Motivated by the feasibility problem (2), we define the two filter entries as

$$f_J(x) := \sum_{i \in J} c_i(x)^- \quad \text{and} \quad h_J(x) := \sum_{i \in J^\perp} c_i(x)^-,$$

respectively. We apply an SQP method to the minimization of (2) that enables us to achieve fast local convergence even if no feasible solution exists.

So far, the filter methods have been concerned with two competing aims. However, filter algorithms can also be developed for more objectives. In [15], we develop a multi-dimensional filter for the solution of $c(x) = 0$. The idea is to split the constraint residuals into $p$ components

$$h_j(x) := \|c_{I_j}(x)\|,\ j = 1, \ldots, p,$$

where $\{1, \ldots, m\} = I_1 \cup \ldots \cup I_p$. We adapt the filter-acceptability by saying that a trial point $x_k^+$ is acceptable if and only if, $\forall l \in \mathcal{F}_k$,

$$\exists j \in \{1, \ldots, p\} : h_j(x_k^+) < h_j(x_l) - \beta \|h(x_l)\|,$$

where $\beta \in (0, 1/\sqrt{p})$ ensures that the right-hand side of this condition always has at least one positive entry, which ensures that we can always generate a filter-acceptable point. The algorithm minimizes a Gauss-Newton or, alternatively, a Newton-model of the least-squares formulation of $c(x) = 0$:

$$\underset{x}{\text{minimize}}\ f(x) := \frac{1}{2}\|h(x)\|_2^2.$$

The trust region is enforced only if a trial point is *not* filter-acceptable. The resulting algorithm is very nonmonotone and works best if we choose $p = m$. We extend this work in [17] to general feasibility problems such as (1) by defining $h_j(x) := \|c_{I_j}(x)^-\|,\ j = 1, \ldots, p$.

This multidimensional filter is also extended to unconstrained minimization in [18] by casting the minimization of $f(x)$ as the solution of the system $\nabla f(x) = 0$. The algorithm contains provisions for negative curvature and is shown to be convergent to second-order critical points. We generalize this algorithm to bound-constrained optimization in [24]. In related work, Sainvitu [23] studies the effect of using approximate derivatives within a filter method.

## 3.2 Nonsmooth optimization

Filter methods for nonsmooth optimization provide a convenient extension of bundle methods to include nonsmooth constraints. We can assume without loss of generality, that the nonsmooth NLP has only a single constraint $c(x) \in \mathbb{R}$, because we can reformulate multiple constraints as a single constraint using the max-function. In [7], we present a straightforward extension of filter methods to bundle trust-region methods. We use two bundles (one for the objective, and one for the constraints) and solve an LP inside a trust region for a step. The convergence analysis is an extension of the SLP convergence proof in [11].

In contrast, the filter method of Karas *et al.* [19] combines ideas from proximal point methods and filter methods. The authors create a cutting plane model of the *improvement function*

$$g_x(y) := \max\{f(y) - f(x), -c(y)\}.$$

This function allows standard unconstrained proximal point methods to be used and requires only a single bundle to be maintained. The authors establish convergence to stationary points and present encouraging numerical results.

A recent variable-metric filter method is presented in [21].

### 3.3 Derivative-free optimization

Audet and Dennis [1] incorporate filter into a pattern-search method for derivative-free constrained optimization. Pattern-search methods target "black-box" applications, where the problem functions $f(x)$ and $c(x)$ are available only as oracles, and derivative information is prohibitive to obtain. The filter in [1] differs in three important aspects from the filters described above: (1) it requires only simple decrease similar to unconstrained pattern-search algorithms, (2) the incumbent (POLL center) is either feasible or the least infeasible iterate, and (3) the filter includes an entry $(0, f_F)$ corresponding to a feasible iterate. A new point $x_k^+$ is acceptable if either of the following two conditions hold:

$$h(x_k^+) = 0 \ \text{ and } \ f(x_k^+) < f_F$$

or

$$h(x_k^+) < h_l \ \text{ or } \ f(x_k^+) < f_l, \ \forall l \in \mathcal{F}_k.$$

The authors extend the usual patter-search convergence results to filter methods.

### 3.4 Augmented Lagrangian

An augmented Lagrangian filter method for QPs is developed in [13]. The algorithm efficiently accommodates matrix-free implementation and is based on two main phases. First, gradient projection iterations approximately minimize the augmented Lagrangian function and provide an estimate of the optimal active set. Second, an equality-constrained QP is approximately minimized on this subspace in order to generate a second-order search direction.

The iterations of augmented Lagrangian methods typically are controlled by two fundamental forcing sequences that ensure convergence to a solution. A decreasing sequence $\omega_k \searrow 0$ determines the required optimality of each subproblem solution and controls the convergence of the dual infeasibility. The second decreasing sequence, $\eta_k \searrow 0$, tracks the primal infeasibility $\|Ax - b\|$ and determines whether the penalty parameter $\rho_k$ is increased or left unchanged.

In the definition of our filter we use quantities that are analogous to $\omega_k$ and $\eta_k$. Define

$$\begin{aligned} h(x,y) &= \| \min \left( x, \nabla_x \mathcal{L}(x,y) \right) \|, \\ f(x) &= \|Ax - b\|, \end{aligned}$$

which are based on the optimality and feasibility of a current pair $(x, y)$. The axis in this filter appear to be the reverse of the usual definition ($f(x)$ measures feasibility). This choice reflects the dual view of the augmented Lagrangian: it can be shown that $Ax_k - b$ is a steepest descent direction for the augmented Lagrangian. We use the filter, rather than the usual forcing sequences, to terminate the inner iteration (minimization of the augmented Lagrangian).

## 4. Conclusions

We have presented filter methods that promote convergence for constrained optimization algorithms without the need of artificial penalty parameters. Filter methods are an alternative to penalty function methods and build on the concept of domination from multiobjective optimization. Filter methods were initially designed for nonlinear programming problems but have quickly become popular in other areas such as nonlinear equations, nonsmooth optimization, and derivative-free methods. We believe that filter methods will continue to grow and find application in more diverse areas.

## Acknowledgments

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357.

REFERENCES

[1] C. Audet and J. E. Dennis Jr., *A pattern search filter method for nonlinear programming without derivatives*, SIAM J. Optim., 14 (2004), pp. 980–1010.

[2] H. Y. Benson, D. F. Shanno, and R. J. Vanderbei, *Interior-point methods for nonconvex nonlinear programming: Filter-methods and merit functions*, Comput. Optim. Appl., 23 (2002), pp. 257–272.

[3] C. M. Chin, *A global convergence theory of a filter line search method for nonlinear programming*, Technical report, Department of Electrical Engineering, University of Malaya, Kuala Lumpur, Malaysia, 2002.

[4] C. M. Chin and R. Fletcher, *On the global convergence of an SLP-filter algorithm that takes EQP steps*, Math. Program., 96 (2003), pp. 161–177.

[5] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, Number 1, SIAM, Philadelphia, 2000.

[6] R. Fletcher, N. I. M. Gould, S. Leyffer, Ph. L. Toint, and A. Wächter, *Global convergence of trust-region SQP-filter algorithms for general nonlinear programming*, SIAM J. Optim., 13 (2002), pp. 635–659.

[7] R. Fletcher and S. Leyffer, *A bundle filter method for nonsmooth nonlinear optimization*, Numerical Analysis Report NA/195, University of Dundee, 1999.

[8] R. Fletcher and S. Leyffer, *Nonlinear programming without a penalty function*, Math. Program., 91 (2002), pp. 239–270.

[9] R. Fletcher and S. Leyffer, *Filter-type algorithms for solving systems of algebraic equations and inequalities*, G. di Pillo and A. Murli, editors, High Performance Algorithms and Software for Nonlinear Optimization, Kluwer Academic Publishers, Dordrecht, (2003) pp. 259–278.

[10] R. Fletcher, S. Leyffer, and Ph. L. Toint, *On the global convergence of a filter-SQP algorithm*, SIAM J. Optim., 13 (2002), pp. 44–59.

[11] R. Fletcher, S. Leyffer, and Ph.L. Toint, *On the global convergence of an SLP-filter algorithm*, Numerical Analysis Report NA/183, University of Dundee, 1998.

[12] R. Fletcher and E. Sainz de la Maza, *Nonlinear programming and nonsmooth optimization by successive linear programming*, Math. Program., 43 (1989), pp. 235–256.

[13] M. P. Friedlander and S. Leyffer, *Gradient projection for general quadratic programs*, Preprint ANL/MCS-P1370-0906, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, 2006.

[14] C. C. Gonzaga, E. Karas, and M. Vanti, *A globally convergent filter method for nonlinear programming*, SIAM J. Optim., 14 (2003), pp. 646–669.

[15] N. I. M. Gould, S. Leyffer, and Ph. L. Toint, *A multidimensional filter algorithm for nonlinear equations and nonlinear least squares*, SIAM J. Optim., 15 (2004), pp. 17–38.

[16] N. I. M. Gould and Ph. L. Toint, *Global convergence of a non-monotone trust-region filter algorithm for nonlinear programming*, Numerical Analysis Report RAL-TR-2003-003, Rutherford Appleton Laboratory, 2003.

[17] N. I. M. Gould and Ph. L. Toint, *FILTRANE, a Fortran 95 filter-trust-region package for solving nonlinear least-squares and nonlinear feasibility problems*, ACM Trans. Math. Software, to appear.

[18] N. I. M. Gould, Ph. L. Toint, and C. Sainvitu, *A filter-trust-region method for unconstrained optimization*, SIAM J. Optim., 16 (2006), pp. 341–357.

[19] E. Karas, A. Ribeiro, C. Sagastizabal, and M. Solodov, *A bundle-filter method for nonsmooth convex constrained optimization*, Technical report, Instituto de Matemática Pura e Aplicada, Rio de Janeiro, Brazil, 2005.

[20] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov, *New variants of bundle methods*, Math. Program., 69 (1995), pp. 111–147.

[21] Y. Peng and Z. Liu, *A filter-variable-metric method for nonsmooth convex constrained optimization*, Technical report, School of Mathematical Sciences and Computing Technology, Central South University, Changsha, Hunan, P. R. China, 2006.

[22] A. A. Ribeiro, E. W. Karas, and C. C. Gonzaga, *Global convergence of filter methods for nonlinear programming*, Technical report, Department of Mathematics, Federal University of Paraná, Brazil, 2006.

[23] C. Sainvitu, *A numerical investigation of the influence of approximate derivatives on the filter-trust-region method for unconstrained optimization*, Report 06/03, Dept. of Mathematics, FUNDP, Namur, 2006.

[24] C. Sainvitu and Ph. L. Toint, *A filter-trust-region method for simple-bound constrained optimization*, Report 05/06, Dept. of Mathematics, FUNDP, Namur, 2006.

[25] P. D. Surry, N. J. Radcliffe, and I. D. Boyd, *A multi-objective approach to constrained optimization of gas supply networks: The COMOGA method*, T. C. Fogarty, editor, Evolutionary Computing: AISB Workshop, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Vol. 993 (1995), pp. 166–180.

[26] M. Ulbrich, S. Ulbrich, and L. N. Vicente, *A globally convergent primal-dual interior-point filter method for nonconvex nonlinear programming*, Math. Program., 100 (2003), pp. 379–410.

[27] S. Ulbrich, *On the superlinear local convergence of a filter-SQP method*, Math. Program., 100 (2004), pp. 217–245.

[28] A. Wächter and L. T. Biegler, *Failure of global convergence for a class of interior point methods for nonlinear programming*, Math. Program., 88 (2000), pp. 565–574.

[29] A. Wächter and L. T. Biegler, *Line search filter methods for nonlinear programming: Local convergence*, SIAM J. Optim., 16 (2005), pp. 32–48.

[30] A. Wächter and L. T. Biegler, *Line search filter methods for nonlinear programming: Motivation and global convergence*, SIAM J. Optim., 16 (2005), pp. 1–31.

[31] A. Wächter and L. T. Biegler, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Math. Program., 106 (2006), pp. 22–57.

# NEOS Version 5

**Jason Sarich**
Industrial Engineering and
Management Sciences Department,
Northwestern University, USA
Mathematics and Computer Science Division,
Argonne National Laboratory,
Argonne, IL 60439, USA (`sarich@mcs.anl.gov`).

## 1. Introduction

The NEOS (Network-Enabled Optimization System) server [2, 3, 5], `neos.mcs.anl.gov`, has been providing a framework for users to access optimization solvers since 1996. The server is operated by the Optimization Technology Center, a joint effort between Argonne National Laboratory and Northwestern University, and handles between 10,000 and 20,000 optimization requests every month. It currently has about fifty commercial, academic, and open-source solvers available for linear, nonlinear, mixed integer, stochastic and semidefinite programs. Depending on the solver chosen, users can submit their optimization problems in the form of AMPL [4] or GAMS [1] models, C or Fortran subroutines, MPS files, and several other solver-specific formats. The NEOS server has been used both by teachers as an educational tool for numerical optimization, and by academic and industrial researchers to solve optimization problems in mathematics, engineering, economics, chemistry, and many other fields.

In 2005, version 5 of NEOS was released, introducing several important improvements. These improvements accommodate both feature request from NEOS users and the desire of the NEOS developers to make the server easier to use and maintain.

The most fundamental change has been the introduction of an application programming interface (API) to the NEOS server. This interface allows users direct access to NEOS functionality — such as submitting jobs, checking the queue, or adding new solvers — through remote procedure calls. Other changes discussed in this article are a new format for email submissions, the introduction of a priority queue to help schedule jobs, and a new system for adding solvers to the NEOS server.

## 2. Submitting jobs to NEOS

### 2.1 Version 4

Version 4 offered four ways to submit jobs to NEOS:

- **Web Browser**
  By far the most popular way to submit jobs to NEOS is through a Web browser such as Microsoft's Internet Explorer or Mozilla Firefox. The advantages to using a browser are clear: There is no software to install, no knowledge of programming or XML (Extensible Markup Language [7]) formatting is required, and job results are conveniently displayed in the browser itself. All one needs to do is decide which solver to use, fill in a form to upload any files and set parameters, and click on the "submit to NEOS" button.

- **Java Submission Tool**
  As an alternative to using a Web browser to submit optimization jobs, NEOS provides a Java graphical user interface (GUI). The main advantages to using this GUI instead of a Web browser are the ability to save submission entries in order to simplify resubmitting similar jobs and the ability to easily compare results from multiple jobs.

- **Email**
  Email can be a good option when no windowing display is available or if one wishes to submit several similar jobs to NEOS without having to fill in all of the forms. To email a job submission to NEOS, one must first download a template for the particular solver desired. This template can be found either at the solver's page on NEOS or by sending an email to NEOS with the text `<category>:<solver>:<input>`. A complete list of solvers can be retrieved by sending an email with the text "help". This template describes each piece of input that the solver needs and how to format an email so that NEOS can parse it into the necessary files.

- **Kestrel**
  Many of the solvers on NEOS accept input in the form of AMPL or GAMS models, and over 90 percent of NEOS job submissions are in one of these two formats. Because of security precautions, however, much of the functionality of the modeling languages is lost when they are submitted to NEOS through the Web browser, Java submission tool, email, or the NEOS API interfaces. As an example, NEOS will not allow any files to be read from or written to the local filesystem when executing a job. Also, any data returned from NEOS is in the form of a text file containing information about the solver, model, and solution. This can make analyzing the data difficult because the text file must be externally parsed into separate vector or matrix elements before any operations can be performed on them. The Kestrel interface to NEOS allows users to combine the flexibility and convenience of using a locally installed AMPL or GAMS system with the ability to access the solvers available on the NEOS server.

Users commonly ask NEOS administrators for a convenient way to automate NEOS submissions. Such requests point out the main disadvantage of using the Web browser or the Java submission tool interface: the necessity of clicking a "Submit" button. Having to click a button can quickly become impractical if one has a large set of problems to submit or if an optimization problem must be solved from inside another application. An example is the Sudoku applet on the NEOS Guide Web pages. This applet reads in variables from the user interface, forms an XML document, submits it to NEOS, and then displays the solution to the user.

Writing a program to generate and email a sequence of problems to NEOS is one approach that has been used for automating solutions, but this causes a strain on the NEOS email server and can result in high latency. Also, one cannot check the NEOS queue to make sure that each new job can be processed. Another approach is to create an AMPL or GAMS program and use the Kestrel interface, but this will work only on solvers that accept AMPL or GAMS input, and for large problems one must purchase an AMPL or GAMS license to use Kestrel.

### 2.2 Version 5

Version 5 of NEOS introduced a fifth way to submit jobs to NEOS, through an application programming interface using the XML-RPC (XML Remote Procedure Call) protocol [6]. This interface allows programs to directly transact with the NEOS server using any XML-RPC client. Libraries for XML-RPC clients are available for most common programming languages including C, C++, Java, Perl, Python, and Ruby. Using these libraries, users can remotely access functions that will run on the NEOS server and return information to the user. A few of the NEOS functions necessary to submit a job and view the results are listed here. The full NEOS API specifications containing over twenty-five functions can be found on the NEOS Web site.

```
submitJob(xmlstring)
  Submit an optimization job to NEOS.

  Use this method to submit your job to NEOS.
  It will return a tuple containing (jobnumber,
  password).  You can the use this jobnumber
  and password to get the results or status
  of your job using the methods getStatus,
  getIntermediateResults, and getFinalResults.

  In case of an error (NEOS Job queue is full),
  submitJob() will return (0,errorMessage)

  For more information on the format of the
  xmlstring, you can use the getSolverTemplate
  function.


getJobStatus(jobNumber, password)
  Get the current status of your job.

  Returns "Done", "Running", "Waiting",
  "Unknown Job", or "Bad Password".


getFinalResults(jobNumber, password)
  Gets results of job from NEOS.

  Retrieve results from NEOS.  If the job is still
  running, then  this function will hang until the
  job is finished.

  This function will return a base-64 encoded object.
  Please read your XML-RPC client documentation for
  decoding.

  (For Python's xmlrpclib library, you can use the
  object's 'data' data member).
```

This API gives users and application programmers access to NEOS functionality such as checking the queue, submitting jobs, checking the status of jobs, retrieving results, and setting up new solvers to run on NEOS. This API is new to NEOS, but as can be seen in Table 1, it already comprises almost one-sixth of all new NEOS job submissions. This number does not mean that one-sixth of all users are using this interface, however, because the ability to automate submissions allows a minority of NEOS users to skew the statistics by easily submitting hundreds or in some cases even thousands of jobs to the NEOS server.

Table 1: The number of submissions through each interface to NEOS since the August, 2005, release of NEOS version 5.

| Interface | Submissions |
|---|---:|
| Web Browser | 206,559 |
| XML-RPC | 46,016 |
| Kestrel | 26,298 |
| Java GUI | 3,197 |
| email | 1,047 |

The main disadvantage of using the NEOS API is the need for XML-RPC client libraries; however, the Python language installation has an XML-RPC module included that makes it ideal for accessing NEOS. An example program for submitting a job to NEOS follows; keep in mind that this is simplified, and in most cases the return values should be checked for errors. The file `example.xml` used in the example is an XML file describing the optimization job. The format used for this file is identical to the format now used for email submissions, which is described in Section 3.. More information about using Python's `xmlrpclib` module can be found in the Python Reference Manual at `http://www.python.org`.

```
import xmlrpclib
neos = xmlrpclib.ServerProxy(
    "http://neos.mcs.anl.gov:3332")
xmlfile = file("example.xml", "r")
xmlstring = xmlfile.read()
(jobNumber,password) =  neos.submitJob(xmlstring)
status = "Waiting"
while (status=="Waiting" or status=="Running"):
  status = neos.getJobStatus(jobNumber, password)
results = neos.getFinalResults(jobNumber, password)
print results.data
```

## 3. New format for email

Users can still submit jobs to NEOS through email, although this method is rarely used now compared to Web page submissions, Kestrel, and NEOS API calls. Jobs are sent to `neos@mcs.anl.gov`, and results are emailed to the user. In order for NEOS to understand the request, a particular format must be used in the email. Previously this format involved text tokens to delimit different parts of the email body. For example, an email submission to the BLMVM solver in the past looked something like the following:

```
TYPE BCO
SOLVER BLMVM
LANG=C
BEGIN.FUNC
(actual C function)
END.FUNC
...
END-SERVER-INPUT
```

In the current version of NEOS, a new format uses XML to delimit the text. The previous submission now looks like the following:

```
<neosjob>
<category>bco</category>
<solver>BLMVM</solver>
<inputType>C</inputType>
<func><![CDATA[
(actual C function)
]]></func>
...
</neosjob>
```

The `<![CDATA[` and `]]>` sequences signify that anything in between should be treated by NEOS as verbatim text, avoiding the need to use an escape sequence for each XML control character such as `<`, `>`, and `&`. If the submitted text does not contain any of these characters, then the `<![CDATA[` and `]]>` sequences are not necessary. The name of the outermost tags (in this case `<document>` and `</document>`) does not matter as long as an outermost set of tags is present.

To help in submitting NEOS jobs via email, XML templates are available either from the particular solver's Web page on NEOS or by emailing NEOS with the text `help category:solver:input`, for example:

help bco:BLMVM:C

A few moments later a response will arrive which includes the following text:

```
<document>
<category>bco</category>
<solver>BLMVM</solver>
<inputMethod>C</inputMethod>
...
<funceval><![CDATA[
...Insert Value Here...
]]></funceval>
...
</document>
```

Moreover, we believe this format is easier to understand, many users are already familiar with XML, and NEOS can now take advantage of available XML parsers instead of writing and maintaining its own parser.

## 4. Separate queues

Another improvement to NEOS is a new two-level queuing system. In the past, it was possible for several large jobs to be sent to NEOS and monopolize the NEOS workstations for several hours, effectively denying service during this time to other users who may only need a minute or two of NEOS computing time. Many solvers on NEOS are now set up to allow users to request that their job be sent to a separate priority queue. Dedicated workstations have been set up to accept jobs only from this queue. Any jobs submitted to this priority queue are limited to running for a set amount of time (usually five minutes), after which they will be terminated if still running. Users can select this priority queue by checking the "priority" box on the solver's Web submission page, by setting the `priority` option to `short` in Kestrel, or for XML-RPC and email submissions by adding the XML tag `<priority>short</priority>`.

## 5. New method for adding solvers to NEOS

A key element to the scalability of NEOS is that the solvers are not all managed by the Optimization Technology Center. People with an optimization solver, an Internet connection, and some computing cycles to donate are encouraged to add their solver to the NEOS server. Currently, NEOS solvers are hosted at Argonne National Laboratory, Northwestern University, Arizona State University, Lehigh University, and Rheinisch-Westfälische Technische Hochschule Aachen.

The process for adding a solver involves the following steps:

- Download and install the NEOS server package from the NEOS Web pages.
- Register the solver with NEOS by creating and submitting an XML file describing the solver, the necessary user inputs, and the workstations it will run on. Instructions for creating the XML file and examples are available on the NEOS Web pages.
- Write a driver program that will run the solver based on the user input files.
- Execute the provided XML-RPC server on each machine the solver will run on.

The technical details for setting up a NEOS solver is beyond the scope of this article. Those interested in hosting a NEOS solver are encouraged to visit the Web page `neos.mcs.anl.gov/neos/SolverHowTo.html` for instructions, and to contact the NEOS administrators for further advice.

## 6. Future directions

In order to continue being a useful tool for optimization, NEOS must continue to adapt and respond to its users. We feel that the NEOS API is an important element to maintaining the flexibility of NEOS. It is not a feature that was simply added on to the existing NEOS server; instead, NEOS was redesigned so that all communication with NEOS ultimately goes through the API.

Several specific changes are in mind for the NEOS server, including a redesign of the NEOS Web to make it easier to use and navigate NEOS. Also under way is a Web-based system for adding and maintaining NEOS solvers.

Other ideas for future work include further improving the NEOS scheduling and queuing system, extending the Kestrel interface to work with more environments such as Mathematica or Matlab, adding more solvers to the NEOS server.

REFERENCES

[1] A. Brooke, D. Kendrick, and A. Meeraus, *GAMS: A User's Guide, Release 2.25.* Scientific Press/Duxbury Press (1992).

[2] J. Czyzyk, M. P. Mesnier, and J. J. Moré, The NEOS Server. *IEEE Journal on Computational Science and Engineering* **5** (1998) 68–75.

[3] J. Czyzyk, J. H. Owen, and S. J. Wright, Optimization on the Internet. *OR/MS Today* **24,** 5 (October 1997) 48–51; also at `www.mcs.anl.gov/otc/Guide/TechReports/otc97-04/orms.html`.

[4] R. Fourer, D.M. Gay, and B.W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming* 2nd ed. Duxbury Press, Pacific Grove, CA (2003).

[5] W. Gropp and J. J. Moré, Optimization Environments and the NEOS Server. In *Approximation Theory and Optimization,* M. D. Buhmann and A. Iserles, eds., Cambridge University Press, 1997, pp. 167–182.

[6] D. Winer, XML-RPC Website, `www.xmlrpc.com` (2003).

[7] *XML 1.1,* 2nd ed., W3C Recommendation, 16 August 2006, T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, J. Cowan, eds.

# DTOS — A Service for the Optimization Community

**Hans D. Mittelmann**
Department of Mathematics and Statistics,
Arizona State University, Tempe, AZ 85287-1804, USA
(mittelmann@asu.edu).

## 1. Introduction

For many years we have been providing a number of free services to the community. The effort began about ten years ago, when Peter Spellucci started with us a web-based listing of optimization software. We maintained this effort together for several years and, later, started the related one of benchmarking selected codes. Our rationale was that although a large number of users of optimization methods exist, the knowledge about which methods are best or even just suitable for a given problem is much more sparsely distributed. Researchers and users of optimization frequently have knowledge in only one subarea, such as in either continuous or discrete optimization.

So, the goal was to provide a list of readily available software for a large number of problem classes and, if there were several codes available for the same class, to provide the additional information of comparative performance. This was done through well-documented and reproducible benchmarks on typical problems. It was natural that only over time several other components of this service activity were added. They are all integrated into the web-based guide which was at the start and which had been given the name

**Decision Tree for Optimization Software**
`http://plato.asu.edu/guide.html`

and which will be subsequently called DTOS. The current components of DTOS are the following:

1. The DTOS Webpage.
2. The benchmark subpage.
3. A collection of test problems in various formats.
4. Some optimization software not provided elsewhere.
5. A substantial portion of the interactive optimization solvers accessible through the NEOS gateway.

While no paper or report has been written about this service effort, several talks accessible through our personal webpage `http://plato.asu.edu` have addressed some of its aspects. The fact, for example, that the third and fourth component are fully integrated into the first two makes them less visible. This is intended, but here we shed a bit more light on the structure of the components of the DTOS project.

A word is in order about the fifth component. Many users of optimization software, once they know which program should be able to solve their problem, are not in a position to implement and run this program. They may not have access to suitable hardware, they may be unfamiliar with Unix/Linux and cannot install/compile the program in this system or in Windows. We were aware of the NEOS project `http://neos.mcs.anl.gov` and decided to join it by enriching the collection of solvers accessible through one gateway, instead of starting a parallel effort, mostly for the convenience of the users. After a short while our share of solvers had increased to a third of the total.

In the following we address each of the five components of our service separately. We refrain from giving complete listings of available resources and services, something that can be much better seen by inspection and that also changes over time.

## 2. The DTOS webpage

This webpage is continually updated and has not undergone a major change since its inception except for a recent format change. A major overhaul has to wait for at least a temporary reduction of our academic workload. Currently the DTOS lists nearly exclusively non-commercial software in nine categories. Few entries are listed in the categories “Complementarity” and “Multi-objective Optimization”. The category “Approximation” is listed separately as an area in which optimization algorithms are employed. The remaining six categories comprise the main body of optimization and the rationale behind its structure is the following.

In most cases the solution sought is the global optimizer, so the first category is "Global Optimization". A limited number of codes is listed in two sections for unconstrained and constrained global optimization. The selection is meant to provide users with a recommendation in many important cases but is not exhaustive and a reference to Arnold Neumaier's related webpage `www.mat.univie.ac.at/~neum/glopt.html` is given.

The remaining bulk of DTOS has the categories "Unconstrained and Constrained Optimization, Least Squares, Zeros, and Discrete Optimization". Even more than for global optimization, an exhaustive listing would become unwieldy for discrete optimization. Due to the lack of a standard reference webpage in this area, however, the number of codes listed is not small.

The categories of finding zeros of (systems of) functions and of least squares problems have their own webpages to expedite the search for users interested in them. Also, in these sections of the DTOS and in those described below an attempt is made to provide a somewhat more complete listing in the sense that, for every major problem class of interest, several available codes, preferable in different languages, are given. The major subareas on the "Zeros" page are univariate and multivariate functions, while the least squares page has a major division into unconstrained and constrained problems with further subdivisions for both.

The most extensive pages are those on unconstrained and constrained optimization both having a three-level structure. The order within these pages is from the simpler to the more general problem with, for example, LP first on the constrained optimization page, followed by QP, SDP, NLP *etc.*. It is thus clear that in many cases codes listed further down can also solve a problem for which the more specialized codes are listed before. Still, in a few cases, codes are listed repeatedly. The code LOQO [1] may serve as an example. It started out as a code for LP, was then extended to QP and then to NLP with facilities to solve SOCP and MPEC problems. It is listed a total of five times.

Codes for generic discrete optimization problems such as MILP, MIQP, MINLP *etc.* are also listed on these pages in addition to their listing in the discrete category while more specialized codes are not.

For a three-year period starting in 2000 the National Science Foundation supported our software evaluation effort. Attempts to obtain funding before and after that time have been unsuccessful. This lead to a limitation of the scope of our work including that on DTOS in general. We next describe our benchmarking effort.

## 3. The benchmarks

Currently, our frequently cited benchmark webpage `http://plato.asu.edu/bench.html` nearly exclusively considers non-commercial codes, mostly developed by researchers in academia. There are eight categories of problems but in most only one benchmark is listed. An exception is the area of SDP and SOCP with five benchmarks. Important features of all our benchmarks are a complete listing of the sources for the codes and of all the test problems, as well as links to the full log files of the runs. Standard platforms were used (Unix or Linux as operating system). Two of the benchmarks that had to be dropped because of lack of support are the commercial LP/MILP benchmarks. The DTOS subpage "Benchmarks" actually lists as the first link our own benchmarks but it subsequently also lists selected papers on the topic as well as a limited number of benchmarks done by others. Our software evaluation effort is the only part of our DTOS project which has led to journal publications [2, 3]. In addition, several of our papers contain some form of comparison of different solvers, *e.g.*, the work on PDE-constrained optimization.

A rather substantial work was our participation in the Seventh DIMACS Implementation Challenge [4]. Over an extended period, ten codes for SDP/SOCP problems and their numerous updates had to be evaluated on a growing (and changing) number of test examples. This did not end at the DIMACS workshop. A rather detailed report was published in Mathematical Programming [3]. We subsequently kept a benchmark alive on some of the DIMACS Challenge problems and for further updates of the codes, and on a large number of additional test problems which we collected and generated (see the section on test problems below). We further installed most of the codes for use through NEOS and we added several further benchmarks in this area. In addition to standard classes such as LP, MILP, or NLP, we also included in our benchmarks areas such as (Q)QP, MI(Q)QP, MINLP, and MPEC including codes in development or having more recently been enabled to solve such problems. Because of the ongoing development of practically all the software considered, results will not be reported here.

## 4. The test problem collection

Over the years of maintaining the DTOS project, test problems in various areas were collected or generated. In most cases, this did not happen in a very systematic way. Still, users of DTOS have found them useful, from PhD students, for example [5, 6], to established researchers, for example [7], and even to commercial vendors of software including ILOG-CPLEX and DASH. There is no URL leading to all the problems we have integrated into DTOS. On the "Testcases" subpage there are more than twenty such links. Here we just mention the extensive collection of LP and MILP problems, both partly taken from submissions to our NEOS solvers, the SDP problems, many of them generated by us, and the "testenvironment" for NLP by Peter Spellucci. Further, there is a link to our problems in the AMPL format. This is a rather extensive collection with 18 subdirectories. Several of these problems come from our own research and were used in the research of others.

## 5. The optimization software collection

Even less systematically than the benchmark and test problem collection, a small repository of programs has developed over time. Again, no single URL shows all the software available but links are spread throughout the DTOS. A number of useful codes that their authors stopped providing is one part of this collection, another is a directory holding all the different versions of Peter Spellucci's SQP code DONLP2(3) as well as his other optimization codes. Further, there are several codes by Mike Powell. All the codes are provided in source form. In addition, we provide binaries for different platforms of a few codes for which sources cannot be made available and for DONLP2 with an AMPL interface. For small to medium-size problems, DONLP2 is quite robust and compares well even with commercial codes. It is downloaded frequently, especially its C version (DONLP3).

## 6. The NEOS Solvers

While the installation and maintenance of web-based optimization solvers seems a project only remotely related to DTOS, there is, in fact a close connection. We were contacted by users who wanted to apply one of the programs they had found on DTOS but for which they had encountered difficulties. A typical example, is an NLP code such as DONLP2 in a language such as Fortran or C, which required the user to compile the code and to provide in the respective language both coded functions and data in specific formats. For NLPs, it is a substantial help to be able to formulate the problem in a modeling language, but the same happens in LP and in most other optimization problems. One additional advantage is then the ability to submit the problem to all solvers that have an interface to the modeling language.

It was thus only natural that we started to implement solvers for access through NEOS which had suitable interfaces and input formats. Our share of NEOS solvers is more than a third (counting input formats). and several are quite frequently used. Here we mention the LP/QP solver BPMPD which has four different input formats and is powerful and quite popular in spite of the fact that two commercial and four other non-commercial codes are available. Another frequently used solver in the MILP category is FEASPUMP, a pure feasibility solver. Due to its power for MILPs, the code SCIP is a favorite, as are several others of our solvers such as the traveling salesman solver CONCORDE (the only NEOS solver with a graphical output), the semi-infinite solver NSIPS, the nondifferentiable solver CONDOR, several of our eight SDP/SOCP solvers, and, more recently, also the exact LP code QSOPT-EX.

Important considerations in the implementation of our NEOS solvers are the number of available input formats and the resources provided, both in CPU time and in memory. Our solvers currently use a total of thirteen different input formats and available memory ranges up to 16 GB; up to four parallel processors are used (by CSDP). We were the first to use Matlab for NEOS solvers and the first to use CPLEX as subsolver, and do both now extensively.

## 7. The future of DTOS

No specific plans exist for the DTOS project except for trying to maintain as many of its components as possible in our spare time. Only with additional time can a major overhaul of the main DTOS webpage be done, reorganizing, updating, and extending the information provided but also making it easier to navigate for users with little or no domain knowledge. Unfortunately, we cannot count on user feedback in the process of updating or extending the DTOS project, there is none. Not even NEOS users alert us to a temporary problem with one of the solvers. Extensions of the benchmarking and web solver efforts are highly unlikely without some additional form of support.

REFERENCES

[1] R. J. Vanderbei and D. F. Shanno, *An interior-point algorithm for nonconvex nonlinear programming*, Comput. Optim. Appl., 13 (1999), pp. 231–252.

[2] H. D. Mittelmann, *Benchmarking interior point LP/QP solvers*, Optim. Methods. Softw., 12 (1999), pp. 655–670.

[3] H. D. Mittelmann, *An independent benchmarking of SDP and SOCP solvers*, Math. Program., 95 (2003), pp. 407–430.

[4] `dimacs.rutgers.edu/Workshops/7thchallenge`

[5] A. Wächter, *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*, Ph.D. Thesis, Carnegie Mellon University, 2002.

[6] S. Bonettini, E. Galligani, and V. Ruggiero, *An inexact Newton method combined with Hestenes multipliers' scheme for the solution of Karush-Kuhn-Tucker systems*, Appl. Math. Comput., 168 (2005), pp. 651–676.

[7] O. Schenk, A. Wächter, and M. Hagemann, *Combinatorial Approaches to the Solution of Saddle-Point Problems in Large-Scale Parallel Interior-Point Methods*, Research Report RC23824, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, USA, 2005.

# Bulletin

## 1. Event Announcements

**Joint EUROPT-OMS Meeting 2007**
July 4–7, 2007
Prague, Czech Republic
`http://cio.umh.es/europt-oms`

The conference aims to review and discuss recent advances and promising research trends in continuous and discrete optimization theory, methods, applications and software development. Topics include: Linear and Nonlinear Optimization, Integer and Combinatorial Optimization, Convex and Nonsmooth Optimization, Global Optimization, Semidefinite Programming, Semi-infinite Programming, Multi-objective Optimization, Stochastic Optimization, Complementarity and Variational Inequality Problems, Derivative-free Optimization, Network Optimization, Scheduling Problems, Optimization in Technological, Bio- and Social Systems, Financial Optimization, Optimal Control, Automatic Differentiation, and Optimization Software,

This international conference is the first joint Meeting organized by the journal Optimization Methods and Software (OMS) and the Working Group on Continuous Optimization (EUROPT) `http://www.iam.metu.edu.tr/EUROPT` of the Association of European Operational Research Societies (EURO). The EUROPT-OMS Conference will be held prior to the 22nd European Conference on Operational Research (EURO XXII) in Prague `http://euro2007.vse.cz` and the 6th International Congress on Industrial and Applied Mathematics (ICIAM 2007) in Zurich `http://www.iciam07.ch`.

The invited speakers are:

- Frederic Bonnans (France)
- Yury Evtushenko (Russia)
- Komei Fukuda (Switzerland)
- Luigi Grippo (Italy)
- Dorit Hochbaum (USA)
- Tibor Illes (Hungary)
- Adrian Lewis (USA)
- Michal Kocvara (Czech Republic)
- Alexander Martin (Germany)
- Florian Potra (USA)
- Liqun Qi (China)
- Philippe Toint (Belgium)
- Stefan Ulbrich (Germany)
- Stephen Wright (USA)
- Yinyu Ye (USA)
- Ya-xiang Yuan (China)

The Conference Organizers are: Oleg Burdakov (Sweden), Conference Co-Chair, Florian Jarre (Germany), Conference Co-Chair, Karel Zimmermann (Czech Republic), Conference Co-Chair, Gerhard Wilhelm Weber (Turkey), Program Committee Chair, Marco A. López (Spain), Organizing Committee Chair, Martin Gavalec (Czech Republic), Local Organizing Committee Chair. The complete list of the Committees Members is available on the conference web site.

Information about the the conference venue, hotel arrangements, abstract submition, session organization, software exhibition, conference publications, dates and deadlines, registration fee, sponsors and partners, and e-mail addresses can be found on the webpage.

Ettore Majorana Centre for Scientific Culture
International School
of Mathematics "G. Stampacchia"
**46th Workshop**
**New Problems and Innovative Methods in Nonlinear Optimization**
July 31 – August 9, 2007
Erice, Italy
`http://www.dis.uniroma1.it/~erice2007`

The Workshop aims to review and discuss recent advances in the development of methods and algorithms for Nonlinear Optimization and its Applications.

Topics include constrained and unconstrained optimization, large scale optimization, global optimization, derivative-free methods, interior point techniques for nonlinear programming, nonlinear complementarity problems, variational inequalities, nonsmooth optimization, neural networks and optimization, and applications of nonlinear optimization.

The Workshop will include invited lectures (1 hour) and contributed lectures (30 min.) Members of international scientific community are cordially encouraged to contribute a lecture describing their current research and applications. Acceptance will be decided by the Advisory Committee of the School.

Invited lecturers who have confirmed the participation are:

- Immanuel Bomze, University of Vienna
- Marco D'Apuzzo, II Universita' di Napoli
- Christodoulos A. Floudas, Princeton Univ.
- William Hager, University of Florida
- Igor Konnov, Kazan University
- Stefano Lucidi, Univ. Roma "La Sapienza"
- Jose Mario Martinez, State Univ. Campinas
- Jorge Nocedal, Northwerstern University
- Jong-Shi Pang, Rensselaer Polytechnic Inst.
- Massimo Pappalardo, Universita' di Pisa
- Panos M. Pardalos, University of Florida
- Daniel Ralph, University of Cambridge
- Fabio Schoen, Universita' di Firenze
- Defeng Sun, National University of Singapore
- Marc Teboulle, Tel-Aviv University
- Luis Nunes Vicente, Univ. Coimbra
- Henry Wolkowicz, University of Waterloo
- Yinyu Ye, Stanford University
- Ya-xiang Yuan, Chinese Academy of Sciences

A special issue of *Computational Optimization and Applications* will be dedicated to the Workshop, including a selection of invited and contributed lectures.

Franco Giannessi
Universita' di Pisa, Italy
Director of the School

Gianni Di Pillo
Universita' di Roma "La Sapienza", Italy
Director of the Workshop

**Czech–French–German 2007**
**Conference on Optimization 2007**
September 17-21, 2007
Heidelberg, Germany
`http://cfg07.uni-hd.de`

The conference is the 13th of the series of French-German meetings which started in Oberwolfach in 1980. This time, it is organized jointly with Czech optimizers, and takes place in the historical center of Germany's oldest university town, Heidelberg. The conference will consist of invited plenary, invited minisymposium and regular talks on all aspects of optimization.

Included topics: continuous optimization (smooth and nonsmooth), numerical methods for mathematical programming, optimal control and calculus of variations, robust optimization, mixed integer optimization, optimization with PDE, differential inclusions and set-valued analysis, stochastic optimization, multicriteria optimization, and optimization techniques for industrial applications.

The invited speakers (confirmed) are:

- Guillaume Carlier (Paris Dauphine)
- Roger Fletcher (University of Dundee)
- Roland Griesse (Austrian Acad. of Sciences)
- Pierre Maréchal (UPS Toulouse)
- Alexander Martin (TU Darmstadt)
- David Preiss (University College London)
- Carsten Scherer (TU Delft)
- Zdenek Strakos (Czech Academy of Sciences)
- Emmanuel Trlat (Universit d'Orlans)
- Michael Valasek (Czech Technical University, Prague)
- Luís Nunes Vicente (Univ. Coimbra)

- Andreas Wächter (IBM, Yorktown Heights)
- Andrea Walther (TU Dresden)

The Organizing Committee is formed by Hans Georg Bock, Moritz Diehl (chair), Gerhard Reinelt, Sebastian Sager (co-chair), and Johannes P. Schlöder.

The complete Program Committee and deadlines can be found on the webpage.

**Follow-up Workshop on**
**Optimization in Finance**
October 26-27, 2007
Coimbra, Portugal
`www.mat.uc.pt/tt2005/follow-up`

The Workshop on Optimization in Finance was held in Coimbra, Portugal, in July 5-8, 2005 Coimbra, Portugal. The quality of the meeting was considered high and the participation surpassed the best expectations. The conference program is still available at the web site `www.mat.uc.pt/tt2005/of`. The workshop was organized under the auspices of the portuguese CIM (International Center for Mathematics, a member of ERCOM European Research Centres on Mathematics), and was one of the events of the CIM Thematic Term on Optimization 2005.

As in the 2005 workshop, the targeted audience for the Follow-up Workshop on Optimization in Finance includes graduate students and faculty members working in applied mathematics, operations research, and economics, who have been or plan to be interested in optimization methods in finance. The event will also be attractive for those doing quantitative modeling in the financial market. The Follow-up Workshop topics include, among others, asset allocation, risk management, and derivative pricing.

The Follow-up Workshop will consist of a limited number of invited lectures. Those interested in contributing can submit a title and an abstract for a poster session. The list of invited speakers include:

- Alexandre d'Aspremont (Princeton University)
- Victor DeMiguel (London Business School)
- Jacek Gondzio (The University of Edinburgh)
- Peter Laurence (Università di Roma "La Sapienza")
- Ioana Popescu (INSEAD)
- Ekkehard Sachs (University of Trier)
- Ralf Werner (Technische Universität Mnchen & Hypo Real Estate Holding)

The meeting is organized by A. M. Monteiro (Univ. Coimbra), R. H. Tütüncü (Goldman Sachs), and L. N. Vicente (Univ. Coimbra).

## Chairman's Column

Reading through the last few issues of Views-and-News, I cannot help but remark on the wonderful job that Luís Vicente has done as the editor of SIAG/OPT's newsletter. In his own comments below, Luís mentions that perhaps some recent issues have been a bit too technical. I certainly agree that it is good to have a diversity of topics in the newsletter, but I for one have really come to appreciate expository articles such as the one by Roger Fletcher, Sven Leyffer and Philippe Toint on Filter Methods in this issue. Views-and-News is one of very few outlets where articles like this, written to be technically precise but at the same time accessible to a relatively wide audience, can appear. I look forward to the remaining issues of Views-and-News that Luís will produce as he and the current elected officers of SIAG/OPT enter our final year of service!

**Kurt M. Anstreicher**, SIAG/OPT Chair
Department of Management Sciences
University of Iowa
S210 PBB Iowa City, IA 52242,
USA
**kurt-anstreicher@uiowa.edu**
`http://www.biz.uiowa.edu/faculty/anstreicher`

## Comments from the Editor

When I look back to the issues previously edited, I have the feeling that the Newsletter might have become a little too technical. I am thus slightly changing the scope of the next couple of numbers, to enrich the pattern and diversity of this publication.

In the current issue we already see a little of this change. The papers on NEOS and DTOS follow this trend. I believe that papers like these can be useful to members of our Activity Group and to other researchers interested in optimization.

The next issue will express personal views of a few first class optimizers who worked for the industry in the last decade. We expect to learn from their views and... to have some fun.

**Luís N. Vicente**, Editor
Department of Mathematics
University of Coimbra
3001-454 Coimbra
Portugal
**lnv@mat.uc.pt**
`http://www.mat.uc.pt/~lnv`