The filter idea and its application to the nonlinear feasibility problem

by N.I.M. Gould[1] and Ph. L. Toint[2]

September 24th 2003

[1] Computational Science and Engineering Department
Rutherford Appleton Laboratory,
Chilton, Oxfordshire, England
Email: nick.gould@stfc.ac.uk

[2] Department of Mathematics,
University of Namur,
61, rue de Bruxelles, B-5000 Namur, Belgium,
Email: philippe.toint@fundp.ac.be

In "Proceedings 20th Biennial Conference on Numerical Analysis" (D. Griffiths and A. Watson, eds), University of Dundee, Scotland (2003)

The filter-idea and its application to the nonlinear feasibility problem

Nick Gould and Philippe L. Toint

24 September 2003

1 Introduction

We consider the solution of the general smooth feasibility problem, that is the problem of finding a vector $x \in \mathbb{R}^n$ such that

$$c_{\mathcal{E}}(x) = 0, \tag{1.1}$$

and

$$c_{\mathcal{I}}(x) \geq 0, \tag{1.2}$$

where $c_{\mathcal{E}}(x)$ and $c_{\mathcal{I}}(x)$ are smooth functions from $\mathbb{R}^n$ into $\mathbb{R}^m$ and $\mathbb{R}^q$, respectively. If such a point cannot be found, it is desirable to find a local minimizer of the constraint violations. We choose here to consider the Eucliden norm of these violations, that is to find a local minimizer of the function

$$\min_x \tfrac{1}{2}\|\theta(x)\|^2, \tag{1.3}$$

where we define

$$\theta(x) \overset{\text{def}}{=} \begin{pmatrix} c_{\mathcal{E}}(x) \\ [c_{\mathcal{I}}(x)]_- \end{pmatrix} \in \mathbb{R}^p, \tag{1.4}$$

with $\|\cdot\|$ denoting the Euclidean norm, with $p = m + q$ and $[c_{\mathcal{I}}(x)]_- = \min[0, c_{\mathcal{I}}(x)]$, the minimum being taken componentwise. An important special case of this problem is when $q = 0, m = n$ and thus $\mathcal{E} = \{1, \ldots, n\}$, which gives systems of smooth nonlinear equations. The problem under consideration is therefore not only fairly general, but also practically important because a large number of applications can be cast in this form. Moreover, solving the feasibility problem may also occur as a subproblem in practically more complicated contexts, such as the the "restoration" phase in the solution of the nonlinear programming problem using filter methods (see Fletcher and Leyffer, 2002, Fletcher and Leyffer, 1998, Fletcher, Leyffer and Toint, 2002*b*, Gonzaga, Karas and Vanti, 2002 or Fletcher, Gould, Leyffer, Toint and Wächter, 2002*a*, amongst others).

The method of choice for solving (1.1)–(1.2) or (1.3) is Newton's method, because of its fast convergence properties. However, as is well-known, Newton's method must be safeguarded to ensure that it converges to a solution even from starting points that are far from the solution, a feature that is not automatic otherwise. Various safeguarding techniques are known, including the use of linesearches (see Ortega and Rheinboldt, 1970, Dennis and Schnabel, 1983, Toint, 1986, Toint, 1987, ...) or trust regions (see Moré and Sorensen, 1984, Nocedal, 1984, or Chapter 16 of Conn, Gould and Toint, 2000). More recently, Gould, Leyffer and Toint (2003*a*) and Gould and Toint (2003) have proposed a method that combines the basic trust-region mechanism with filter techniques: not only did they prove global convergence for the algorithm, but they also reported very encouraging numerical experience. The objective of this note is to outline this method.

2 The filter algorithm and its algorithmic options

2.1 The objective function, its models and the step

We consider an algorithm which aims at minimizing

$$f(x) = \tfrac{1}{2}\|\theta(x)\|^2.$$

For simplicity of exposition, we assume that the problem only contains nonlinear equations ($q = 0$). In this case, we may build two distinct local quadratic models of $f(x)$ in the neighbourhood of a given iterate x_k. The first is the Gauss-Newton model, and is given by

$$m_k^{\rm GN}(x_k + s) = \tfrac{1}{2}\|c_{\mathcal{E}}(x_k) + J_{\mathcal{E}}(x_k)s\|^2, \tag{2.1}$$

where $J_{\mathcal{E}}(x_k)$ is the Jacobian of $c_{\mathcal{E}}(x)$ at x_k. The second is the full second-order Newton model

$$m_k^{\rm N}(x_k + s) = m_k^{\rm GN}(x_k + s) + \tfrac{1}{2}\sum_{j\in\mathcal{E}} c_j(x_k)\langle s, \nabla^2 c_j(x_k)s\rangle, \tag{2.2}$$

which includes an additional term involving the curvature of the equality constraints.

In our method, we have chosen to compute the step s_k by minimizing one of these models in some region surrounding the current iterate x_k, defined by the constraint

$$\|s_k\|_k \le \tau_k\Delta_k, \tag{2.3}$$

where Δ_k is a trust-region radius which is updated in the usual trust-region manner (see Chapters 6 and 17 of Conn et al., 2000, for instance). The parameter $\tau_k \ge 1$ allows for steps that potentially extend much beyond the limit of the trust region itself, in the case where convergence seems satisfactory. The precise mechanism for determining τ_k will be discussed in more detail below. The $\|\cdot\|_k$ norm appearing in (2.3) is a preconditioned Euclidean norm, that is $\|s\|_k^2 = \langle s, P_k^{-1}s\rangle$, where P_k is a symmetric positive-definite preconditioning matrix that is used at the k-th iteration. The solution of the subproblem of minimizing $m_k^{\rm GN}(x_k+s)$ or $m_k^{\rm N}(x_k+s)$ subject to (2.3) is computed approximately using the Generalized Lanczos Trust-Region (GLTR) method of Gould, Lucidi, Roma and Toint (1999) as implemented in the GLTR module of GALAHAD (see Gould, Orban and Toint, 2003*b*). Besides using $P_k = I$ (i.e. no preconditioning at all), our package, named FILTRANE, can also be instructed to use a diagonal preconditioning which is obtained by extracting the diagonal of the matrix $H_k \overset{\text{def}}{=} J_{\mathcal{E}}(x_k)J_{\mathcal{E}}(x_k)^T$, or a banded preconditioning matrix of semi-bandwidth 5 obtained by extracting the corresponding part of H_k and modifying it if necessary to ensure its positive definiteness (see Conn, Gould and Toint, 1992 for details of that procedure). It also allows a user-defined preconditioning via its reverse communication interface.

2.2 The filter-trust-region mechanism

Once the step s_k has been computed, we define the *trial point* $x_k^+ = x_k + s_k$ and consider the question of deciding whether or not it is acceptable as our next iterate x_{k+1}. We use a so-called filter to answer this question.

In order to define our filter, we first say that a point x_1 *dominates* a point x_2 whenever

$$|\theta_i(x_1)| \le |\theta_i(x_2)| \text{ for all } i \in \mathcal{E}.$$

Thus, if iterate x_{k_1} dominates iterate x_{k_2}, the latter is of no real interest to us since x_{k_1} is at least as good as x_{k_2} for each i. All we need to do now is to remember iterates that are not dominated by other iterates using a structure called a filter. A *filter* is a list $\mathcal{F}$ of m-tuples of the form $(\theta_{1,k}|, \ldots, |\theta_{m,k}|)$ such that, for $k \ne \ell$,

$$|\theta_{i,k}| < |\theta_{i,\ell}| \text{ for at least one } i \in \mathcal{E}.$$

Filter methods then accept the new trial iterate x_k^+ if it is not dominated by any other iterate in the filter. While the idea of not accepting dominated trial points is simple and elegant, it needs to be refined a little in order to provide an efficient algorithmic tool. In particular, we do not wish to accept a new point x_k^+ if $\theta_k^+ \overset{\text{def}}{=} \theta(x_k^+)$ is too close to being dominated by another point already in the filter. To avoid this situation, we slighly strengthen our acceptability condition. More formally, we say that a new trial point x_k^+ is *acceptable for the filter* $\mathcal{F}$ if and only if

$$\forall \theta_\ell \in \mathcal{F} \quad \exists\, i \in \mathcal{E} \quad |\theta_i(x_k^+)| < \Big[|\theta_{i,\ell}| - \gamma_\theta \|\theta_\ell\|\Big]_+ \tag{2.4}$$

where $\gamma_\theta \in (0, 1/\sqrt{m})$ is a small positive constant and $[w]_+ = \max[0, w]$.

In order to avoid cycling, and assuming the trial point is acceptable in the sense of (2.4), we may wish to add it the to the filter, so as to avoid other iterates that are worse, that is we perform the simple operation $\mathcal{F} \leftarrow \mathcal{F} \cup \{\theta_k\}$. This may however cause an existing filter value θ_ℓ to be *strongly dominated* in the sense that

$$\exists\, \theta_q \in \mathcal{F} \quad \forall j \in \{1, \ldots, p\} \quad |\theta_{j,\ell}| \ge |\theta_{j,q}| - \gamma_\theta \|\theta_\ell\|. \tag{2.5}$$

If this happens, we simplify later comparisons by removing θ_ℓ from the filter.

If the trial point is not acceptable for the filter, it may nevertheless be acceptable for the usual trust-region mechanism. This requires that $\|s_k\| \le \Delta_k$ and that

$$\rho_k = \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)} \tag{2.6}$$

is sufficiently positive. Our algorithm therefore combines the filter and trust-region acceptability criteria to allow a potentially larger set of trial points to be accepted.

Inequality constraints are treated in way entirely similar to that used for equalities: as already mentioned in (1.4) we define θ to measure the violation of the inequality constraints. Although the ℓ_2-penalty function (1.3) has discontinuous second derivatives on the boundary of the set of vectors satisfying the inequality constraints, this does not seem to create numerical difficulties when using the Gauss-Newton model.

2.3 An outline of the algorithm and a glimpse on numerical performance

We now outline the FILTRANE algorithm using the ideas developed above. This outline is presented as Algorithm 2.1, page 77. Clearly, this description leaves a number of points unspecified and not fully explained. A more detailed discussion is beyond the scope of this note, and we refer the reader to Gould and Toint (2003) for an in-depth analysis.

We simply attempt to motivate the reader to investigate further by showing performance profiles comparing the classical trust-region framework to that including the filter technique (using the Gauss-Newton model (2.1)) introduced above on a set of 122 problems from the CUTEr collection (see Gould et al., 2003*b*). Given a set of test problems and a set of competing algorithms, the i-th performance profiles $p_i(\alpha)$ indicates the fraction of problems for which the i-th algorithm is within a factor α of the best for a given metric (see Dolan and Moré, 2002 for a formal definition of performance profiles and a discusion of their properties) Comparisons of both iteration counts and CPU times indicate that the filter technique results in a considerably improved performance compared to the more classical technique, and this is true both in reliability (on the right of the profiles) and in efficiency (on their left).

3 Conclusion

We have briefly outlined the main ideas and the algorithm that are behing the FILTRANE package, which is available in the GALAHAD library of optimization programs (see Gould et al., 2003*b* and `http://galahad.rl.ac.uk/galahad-www/`). We have also indiacted that FILTRANE appears to be remarkably robust and efficient. We hope the supplied pointers will encourage the reader to pursue this subject of research, maybe with the help of the two more detailed papers on the subject, namely Gould et al. (2003*a*) and Gould and Toint (2003).

Acknowledgements

The second author is indebted to the Belgian National Fund for Scientific Research for its support during his 2002-2003 sabbatical leave. Some of this research was sponsored

```
Algorithm 2.1: Outline of the Filter-Trust-Region Algorithm
Step 0: Initialization.
    An initial point $x_0$ and an initial trust-region radius $\Delta_0 > 0$ are given, as
    well as constants $0 < \gamma_0 \leq \gamma_1 < 1 \leq \gamma_2$, $\gamma_\theta \in (0, 1/\sqrt{p})$, $0 < \eta_1 < \eta_2 < 1$.
    Compute $c_0 = c(x_0)$ and $\theta_0$. Set $k = 0$, $\mathcal{F} = \emptyset$, and select $\tau_0 \geq 1$.

Step 1: Test for termination.
    If either $\theta_k$ or $\|\nabla f(x_k)\|$ is sufficiently small, stop.

Step 2: Choose a model and a norm.
    Choose a norm $\|\cdot\|_k$ for (2.3). Set $m_k$ to be either $m_k^{\mathrm{GN}}$ or $m_k^{\mathrm{N}}$.

Step 3: Determine a trial step.
    Compute a step $s_k$ using the GLTR algorithm. If the model is found to be
    nonconvex and $\tau_k > 1$, reenter the GLTR algorithm with $\tau_k = 1$. Compute
    the trial point $x_k^+ = x_k + s_k$.

Step 4: Evaluate the residual at the trial step.
    Compute $c(x_k^+)$ and $\theta_k^+ = \theta(x_k^+)$. Define $\rho_k$ according to (2.6).

Step 5: Test to accept the trial step.
    • If $x_k^+$ is acceptable for the current filter:
        Set $x_{k+1} = x_k^+$, select $\tau_{k+1} \geq 1$ and add $\theta_k^+$ to $\mathcal{F}$ if either $\rho_k < \eta_1$ or
        $\|s_k\| > \Delta_k$.
    • If $x_k^+$ is not acceptable for the current filter:
        If $\|s_k\| \leq \Delta_k$ and $\rho_k \geq \eta_1$, set $x_{k+1} = x_k^+$ and select $\tau_{k+1} \geq 1$. Else, set
        $x_{k+1} = x_k$ and $\tau_{k+1} = 1$.

Step 6: Update the trust-region radius.
    If $\|s_k\| \leq \Delta_k$, update the trust-region radius by choosing
        $$\Delta_{k+1} \in \begin{cases} [\gamma_0\Delta_k, \gamma_1\Delta_k] & \text{if } \rho_k < \eta_1, \\ [\gamma_1\Delta_k, \Delta_k] & \text{if } \rho_k \in [\eta_1, \eta_2) \\ [\Delta_k, \gamma_2\Delta_k] & \text{if } \rho_k \geq \eta_2; \end{cases}$$
    otherwise, set $\Delta_{k+1} = \Delta_k$. Increment $k$ by one and go to Step 1.
```

by EPSRC grants GR/R46641 and GR/S02969/01.

References

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. LANCELOT*: a Fortran package for large-scale nonlinear optimization (Release A).* Number 17 *in* 'Springer Series

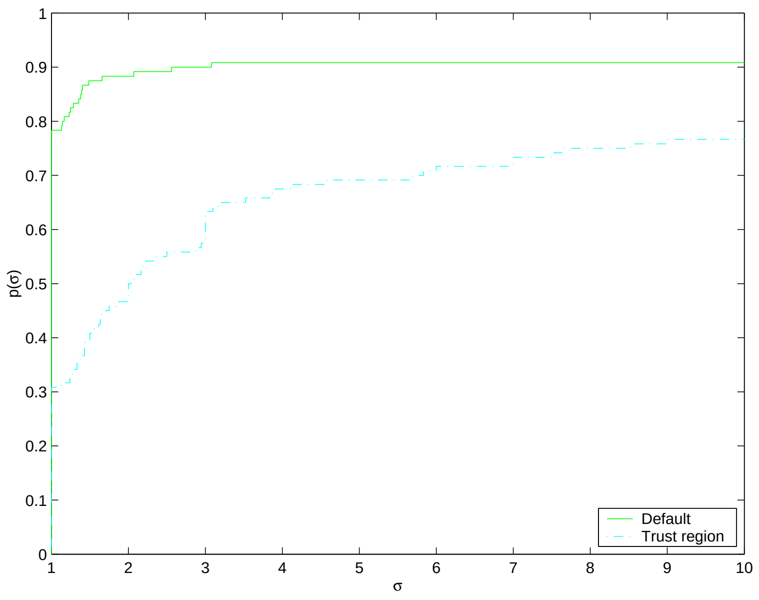

Figure 2.1: Iteration performance profiles $p(\alpha)$ for the default FILTRANE variant (including filter) and the pure trust-region variant (no filter)

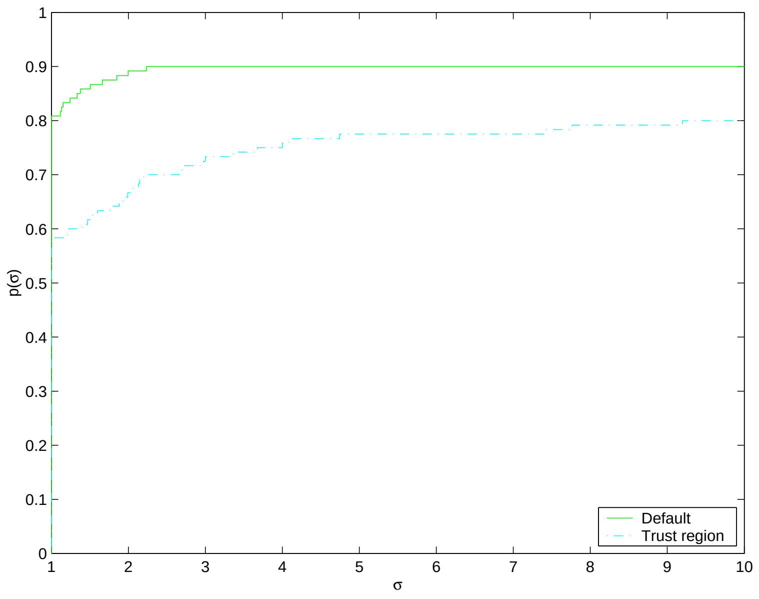

Figure 2.2: CPU time performance profiles $p(\alpha)$ for the default FILTRANE variant (including filter) and the pure trust-region variant (no filter)

in Computational Mathematics'. Springer Verlag, Heidelberg, Berlin, New York, 1992.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Number 01 *in* 'MPS-SIAM Series on Optimization'. SIAM, Philadelphia, USA, 2000.

J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization* *and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1983. Reprinted as *Classics in Applied Mathematics 16*, SIAM, Philadelphia, USA, 1996.

E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, **91**(2), 201–213, 2002.

R. Fletcher and S. Leyffer. User manual for filterSQP. Numerical Analysis Report NA/181, Department of Mathematics, University of Dundee, Dundee, Scotland, 1998.

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

R. Fletcher, N. I. M. Gould, S. Leyffer, Ph. L. Toint, and A. Wächter. Global convergence of trust-region SQP-filter algorithms for nonlinear programming. *SIAM Journal on Optimization*, **13**(3), 635–659, 2002*a*.

R. Fletcher, S. Leyffer, and Ph. L. Toint. On the global convergence of a filter-SQP algorithm. *SIAM Journal on Optimization*, **13**(1), 44–59, 2002*b*.

C. C. Gonzaga, E. Karas, and M. Vanti. A globally convergent filter method for nonlinear programming. Technical report, Department of Mathematics, Federal University of Santa Catarina, Florianopolis, Brasil, 2002.

N. I. M. Gould and Ph. L. Toint. FILTRANE, a Fortran 95 filter-trust-region package for solving systems of nonlinear equalities, nonlinear inequalities and nonlinear least-squares problems. Technical Report 03/15, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2003.

N. I. M. Gould, S. Leyffer, and Ph. L. Toint. A multidimensional filter algorithm for nonlinear equations and nonlinear least-squares. Technical Report TR-2003-004, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2003*a*.

N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM Journal on Optimization*, **9**(2), 504–525, 1999.

N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD—a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization. *Transactions of the ACM on Mathematical Software*, **29**(4), (to appear), 2003*b*.

J. J. Moré and D. C. Sorensen. Newton's method. *in* G. H. Golub, ed., 'Studies in Numerical Analysis', number 24 *in* 'MAA Studies in Mathematics', pp. 29–82, Providence, Rhode-Island, USA, 1984. American Mathematical Society.

J. Nocedal. Trust region algorithms for solving large systems of nonlinear equations. *in* W. Liu, T. Belytschko and K. C. Park, eds, 'Innovative Methods for Nonlinear Problems', pp. 93–102. Pineridge Press, 1984.

J. M. Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. Academic Press, London, 1970.

Ph. L. Toint. Numerical solution of large sets of algebraic nonlinear equations. *Mathematics of Computation*, **46**(173), 175–189, 1986.

Ph. L. Toint. On large scale nonlinear least squares calculations. *SIAM Journal on Scientific and Statistical Computing*, **8**(3), 416–435, 1987.