Proceedings of the 9th WSEAS International Conference on Applied Mathematics, Istanbul, Turkey, May 27-29, 2006 (pp372-378)

A Combined Filter Line Search and Trust Region Method for Nonlinear Programming

CHOONG MING CHIN
Asian Research Centre
British Telecom
Cyberjaya 63000
MALAYSIA

ABDUL HALIM ABDUL RASHID
University of Malaya
Institute of Mathematical Sciences
Pantai Valley, Kuala Lumpur 50603
MALAYSIA

KHALID MOHAMED NOR
Technological University of Malaysia
Department of Electrical Engineering
Skudai 81310, Johor
MALAYSIA

Abstract: A framework for solving a class of nonlinear programming problems via the filter method is presented. The proposed technique first solve a sequence of quadratic programming subproblems via line search strategy and to induce global convergence, trial points are accepted provided there is a sufficient decrease in the objective function or constraints violation function. In the event when the step size has reached a minimum threshold such that the trial iterate is rejected by the filter, the algorithm temporarily exits to a trust region based algorithm to generate iterates that approach the feasible region and also acceptable to the filter. Computational results on selected large scale CUTE problems on the prototype code filLS are very encouraging and numerical performance with LOQO and SNOPT show that the algorithm is efficient and reliable.

Key–Words: nonlinear programming, line search, trust region, SQP, SNQP.

1 Introduction

This paper concerns the development of an alternative filter algorithm for finding a local solution of the following Nonlinear Programming (NLP) problem

$$P \left\{ \begin{array}{lll} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} & f(\mathbf{x}) & \\ \text{subject to} & c_i(\mathbf{x}) \le 0 & i = 1, 2, \ldots, m \end{array} \right.$$

where we assume $f : \mathbb{R}^n \mapsto \mathbb{R}$ and $\mathbf{c} : \mathbb{R}^n \mapsto \mathbb{R}^m$ are twice differentiable.

In the paper by Fletcher and Leyffer [6] the authors have proposed solving Problem P using a filter method as an alternative to traditional merit function approaches. The underlying concept is fairly simple where trial points generated from solving a sequence of trust region quadratic programming (QP) subproblems are accepted provided there is a sufficient decrease in the objective function or constraints violation function. Furthermore extension of the filter technique to accommodate trust region Sequential Linear Programming (SLP) methods have also been reported in [3] where methods have been adapted to allow the possibility of taking equality based quadratic programming subproblem (EQP) steps. Besides trust region methods, extension of optimization techniques using the filter strategy can be found in derivative-free optimization approach [1], bundle method for nonsmooth optimization [5], interior point approach [9] and line search approach [10].

In view of the latest development in using the filter strategy in nonlinear programming we also dispense with the idea of using merit functions to induce global convergence in algorithms for NLP. Instead of purely focussing on trust region methods in NLP, we on the other hand propose and analyze an alternative filter strategy framework based on line search and trust region methods. It has been known that line search methods incorporating merit functions can converge to singular non-stationary points if the Jacobian matrix of active linearized constraints are linearly dependent at non-stationary points. In the context of a filter line search method, if the trial steps become too small when utilizing backtracking strategy then the filter algorithm will temporarily exit and enter into *feasibility restoration phase* [6]. The main objective of entering the feasibility restoration phase is to get closer to the feasible region by minimizing the constraints violation function. By using such a strategy it is hoped that the point generated from the feasibility restoration phase is acceptable to the filter and for which the QP subproblem is feasible. Take note that the purpose of this paper is to deal with the aspects of the proposed method as they relate to the algorithmic implementation. As for addressing the global as well as local convergence aspects of this method see [4].

To give a detailed treatment of this study, we organize this paper in the following manner. In Section 2 we begin by introducing the filter method and we will also show how it can be adapted in a line search SQP algorithm. In addition we will discuss the "slanting" filter technique which first featured in [3] so that stronger statements about convergence to a feasible point can be made. In this section also we will discuss the role of the feasibility restoration phase and how it is augmented in the filter algorithm. In Section 3 the implementation details of the prototype code filLS are discussed and we also present the numerical results by comparing the performance of filLS, LOQO and SNOPT on selected test problems. Finally in Section 4 we conclude the paper.

2 The Filter Line Search Algorithm

We consider solving Problem P iteratively and in order for our algorithm to obtain second order convergence of the iterates, one of the most attractive choices is to use Sequential Quadratic Programming (SQP) method as the basic iterative method. At the current iterate $\mathbf{x}_k$, where k is the iteration number, the QP subproblem in our algorithm is defined by

$$QP(\mathbf{x}_k)\left\{\begin{array}{ll} \underset{\mathbf{d}\in\mathbb{R}^n}{\text{minimize}} & \nabla f(\mathbf{x}_k)^T\mathbf{d}+\frac{1}{2}\mathbf{d}^T\mathbf{W}_k\mathbf{d} \\ \text{subject to} & \nabla c_i(\mathbf{x}_k)^T\mathbf{d}+c_i(\mathbf{x}_k)\le 0, \\ & i=1,2,\ldots,m \end{array}\right.$$

and we denote the local minimizer, the QP step as $\mathbf{d}_k$ (if it exists). After $\mathbf{d}_k$ has been computed, a step size $\alpha\in(0,1]$ is determined in order to obtain a trial iterate

$$\mathbf{x} \;=\; \mathbf{x}_k+\alpha\mathbf{d}_k.$$

As in line search strategy the step size α is chosen via a backtracking strategy so that $\mathbf{x}$ satisfies the filter requirements. If $\mathbf{x}$ satisfies filter conditions we then set $\mathbf{x}_{k+1}=\mathbf{x}$ and $\alpha_k=\alpha$.

On the other hand if the QP step is rejected by the filter for some $\alpha\in(0,1]$ and to overcome the difficulties associated with the Maratos effect, we then construct a second order correction (SOC) step. By adapting of an idea used by [8] we first calculate a step $\widetilde{\mathbf{d}}_k$ (if it exists) from solving the following modified QP subproblem

$$\widetilde{QP}(\mathbf{x}_k)\left\{\begin{array}{ll} \underset{\widetilde{\mathbf{d}}\in\mathbb{R}^n}{\text{minimize}} & \mathbf{g}_k^T\widetilde{\mathbf{d}}+\frac{1}{2}\widetilde{\mathbf{d}}^T\mathbf{W}_k\widetilde{\mathbf{d}} \\ \text{subject to} & \nabla c_i(\mathbf{x}_k)^T\widetilde{\mathbf{d}}+c_i(\mathbf{x}_k+\mathbf{d}_k) \\ & =-\|\mathbf{d}_k\|^\nu, i\in\mathcal{A}(\mathbf{x}_k) \end{array}\right.$$

where $\mathbf{g}_k=\nabla f(\mathbf{x}_k)+\mathbf{W}_k\mathbf{d}_k$, $\nu\in(2,3)$ and $\mathcal{A}(\mathbf{x}_k)=\{i:\nabla c_i(\mathbf{x}_k)^T\mathbf{d}_k+c_i(\mathbf{x}_k)=0\}$. Using the inherited step size value $\alpha\in(0,1]$ from the previous trial iterate $\mathbf{x}_k+\alpha\mathbf{d}_k$, we then utilize the SOC step $\mathbf{d}_k+\widetilde{\mathbf{d}}_k$ where for some $\alpha\in(0,1]$ we set

$$\mathbf{x} \;=\; \mathbf{x}_k+\alpha\mathbf{d}_k+\alpha^2\widetilde{\mathbf{d}}_k$$

and then subject $\mathbf{x}$ to the required filter test. If the new trial iterate $\mathbf{x}$ satisfies all the filter conditions we then set $\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha\mathbf{d}_k+\alpha^2\widetilde{\mathbf{d}}_k$ and $\alpha_k=\alpha$. However, if the new trial iterate fails to be accepted by the filter we then reduce the step size α via a backtracking strategy and return to test the QP and subsequently SOC steps again until either $\mathbf{x}_k+\alpha\mathbf{d}_k$ or $\mathbf{x}_k+\alpha\mathbf{d}_k+\alpha^2\widetilde{\mathbf{d}}_k$ satisfies the filter requirements or temporarily exits from the main filter algorithm to the feasibility restoration phase.

Using the technique as in trust region methods we let the *actual reduction* in $f(\mathbf{x}_k)$ as

$$\Delta f=f(\mathbf{x}_k)-f(\mathbf{x}_k+\alpha\mathbf{d}_k)$$

if QP step is used or

$$\Delta f=f(\mathbf{x}_k)-f(\mathbf{x}_k+\alpha\mathbf{d}_k+\alpha^2\widetilde{\mathbf{d}}_k)$$

if SOC step is used. Furthermore we let

$$\Delta l^{(\alpha)} \;=\; -\alpha\nabla f(\mathbf{x}_k)^T\mathbf{d}_k$$

as the *linear reduction* in $f(\mathbf{x}_k)$. Our sufficient reduction condition for $f(\mathbf{x}_k)$ then takes the form

$$\Delta f \;\ge\; \sigma\Delta l^{(\alpha)}$$

where $\sigma\in(0,\frac{1}{2})$ is a pre-assigned parameter. In some ways the sufficient reduction test resembles the use of Armijo line search condition for unconstrained optimization problems.

We now turn our attention to the definition of an NLP filter introduced in [6]. Basically in nonlinear programming there are two competing aims to satisfy that is to minimize $f(\mathbf{x})$ and to minimize $h(\mathbf{c}(\mathbf{x}))$ where

$$h(\mathbf{c}(\mathbf{x}))=\sum_{i=1}^{m}\max\{0,c_i(\mathbf{x})\}.$$

Note that in this paper we would use the above formulation as a measure of constraint infeasibility. The pair $(h(\mathbf{c}(\mathbf{x}_i)),f(\mathbf{x}_i))$ is said to be acceptable for inclusion in the filter if either

$$h(\mathbf{c}(\mathbf{x}_i))<h(\mathbf{c}(\mathbf{x}_j)) \text{ or } f(\mathbf{x}_i)<f(\mathbf{x}_j)$$

for all $j\in\mathcal{F}^{(k)}$ where $\mathcal{F}^{(k)}$ denotes the set of iterations indices j $(j\le k)$ such that $(h(\mathbf{c}(\mathbf{x}_j)),f(\mathbf{x}_j))$

is an entry in the current filter. Hence a filter is a list of pairs $(h(\mathbf{c}(\mathbf{x}_i)),\, f(\mathbf{x}_i))$ such that no pair dominates any other where domination in our definition means $h(\mathbf{c}(\mathbf{x}_i)) \le h(\mathbf{c}(\mathbf{x}_j))$ and $f(\mathbf{x}_i) \le f(\mathbf{x}_j)$ for $i, j \in \mathcal{F}^{(k)}$.

For the purpose of proving convergence [4], the present definition of a filter is inadequate as it allows points to accumulate in the neighbourhood of filter entries that are not Kuhn-Tucker points. This is readily corrected by defining a small "slanting" envelope around the current filter entries. In our test, the trial iterate $\mathbf{x}_i$ is acceptable to the filter if either

$$h(\mathbf{c}(\mathbf{x}_i)) \le (1-\eta)h(\mathbf{c}(\mathbf{x}_j)) \qquad (2.1)$$

or

$$f(\mathbf{x}_i) \le f(\mathbf{x}_j) - \gamma h(\mathbf{c}(\mathbf{x}_i)) \qquad (2.2)$$

for all $j \in \mathcal{F}^{(k)}$ where $\eta \in (0,1)$, $\gamma \in (0,1)$ are parameters close to zeroes. This idea is illustrated in Figure 1 using values $\eta = \gamma = 0.1$ although in practice η and γ are close to zeroes.

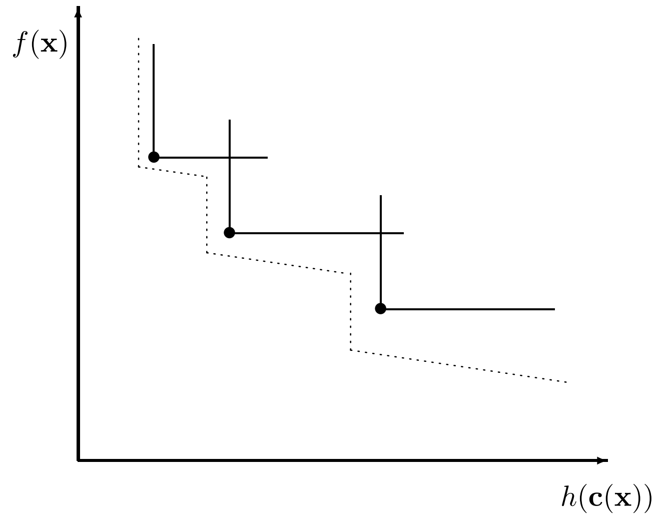

Figure 1: An NLP filter with "slanting" envelope strategy

Apart from using conditions (2.1)-(2.2) we also impose an upper bound criteria

$$h(\mathbf{c}(\mathbf{x})) \le (1-\eta)u$$

where $u > 0$ on any filter entries and this is readily implemented by initializing the filter with the entry $(u, -\infty)$.

In response to the new developments we now state the filter acceptability test in a clearer context. For a trial iterate $\mathbf{x} = \mathbf{x}_k + \alpha\mathbf{d}_k$ for some step size $\alpha > 0$, the point $\mathbf{x}$ is said to be *acceptable to the filter* if the two conditions given below hold true

(1) $h(\mathbf{c}(\mathbf{x})) \le (1-\eta)h(\mathbf{c}(\mathbf{x}_j))$ or $f(\mathbf{x}) \le f(\mathbf{x}_j) - \gamma h(\mathbf{c}(\mathbf{x}))$ for all filter entries $j \in \mathcal{F}^{(k)}$.

(2) $h(\mathbf{c}(\mathbf{x})) \le (1-\eta)u$.

Following the above definitions and explanation, we are now in a position to state the filter line search SQP algorithm by means of the following pseudo-code.

Filter Line Search SQP Algorithm

Given initial point $\mathbf{x}_0$, $t \in (0,1)$, set $\sigma \in (0, \frac{1}{2})$, $\eta \in (0,1)$, $\gamma \in (0,1)$ and set the iteration index $k := 0$. If $h(\mathbf{c}(\mathbf{x}_0)) \ne 0$ let $k \in \mathcal{F}^{(k)}$. Set $(u, -\infty)$ in the filter.

```
REPEAT
    Solve $QP(\mathbf{x}_k)$ subproblem to obtain a local minimizer $\mathbf{d}_k$.
    Set $\alpha = 1$ and
    $\alpha_{\min} \begin{cases} = 0 & \text{if } h(\mathbf{c}(\mathbf{x}_k)) = 0, \\ \in (0, h(\mathbf{c}(\mathbf{x}_k))^2) & \text{if } h(\mathbf{c}(\mathbf{x}_k)) \in (0,1), \\ \in (0,1) & \text{otherwise.} \end{cases}$
    IF $QP(\mathbf{x}_k)$ is infeasible THEN
      Goto Feasibility Restoration Phase to find $\mathbf{x}_{k+1}$
      so that it is acceptable to the filter and the
      $QP(\mathbf{x}_{k+1})$ subproblem is feasible.
    ELSE
      REPEAT
        IF $\mathbf{x}_k + \alpha\mathbf{d}_k$ is acceptable to the filter THEN
          IF $\Delta l^{(\alpha)} > 0$ and $\Delta f < \sigma\Delta l^{(\alpha)}$ THEN
            • Set $\alpha := \alpha t$.
          ELSE
            • Set $\alpha_k = \alpha$, $\Delta f_k = \Delta f$,
              $\Delta l_k^{(\alpha)} = \Delta l^{(\alpha)}$.
            • Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{d}_k$.
            • If $h(\mathbf{c}(\mathbf{x}_{k+1})) > 0$ then initially set
              $\mathcal{F}^{(k+1)} = \mathcal{F}^{(k)} \cup \{k+1\}$ and
              remove any points from the filter
              that are dominated by
              $(f(\mathbf{x}_{k+1}), h(\mathbf{c}(\mathbf{x}_{k+1})))$.
          ENDIF
        ELSE
          Solve $\widetilde{QP}(\mathbf{x}_k)$ subproblem to obtain a
          local minimizer step $\widetilde{\mathbf{d}}_k$ (provided the
          subproblem is solved for the first time).
          IF $\widetilde{QP}(\mathbf{x}_k)$ is infeasible THEN
            • Set $\alpha := \alpha t$.
          ELSE
            IF $\mathbf{x}_k + \alpha\mathbf{d}_k + \alpha^2\widetilde{\mathbf{d}}_k$ is acceptable to
            the filter THEN
              IF $h(\mathbf{c}(\mathbf{x}_k)) = 0$ THEN
                IF $\Delta l^{(\alpha)} > 0$ and
                  $\Delta f < \sigma\Delta l^{(\alpha)}$ THEN
                  • Set $\alpha := \alpha t$.
                ELSE
                  • Set $\alpha_k = \alpha$, $\Delta f_k = \Delta f$,
                $\Delta l_k^{(\alpha)} = \Delta l^{(\alpha)}$.
              • Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k + \alpha_k^2 \widetilde{\mathbf{d}}_k$.
              • If $h(\mathbf{c}(\mathbf{x}_{k+1})) > 0$ then set
                $\mathcal{F}^{(k+1)} = \mathcal{F}^{(k)} \cup \{k+1\}$
                and remove any points from
                the filter that are dominated
                by $(f(\mathbf{x}_{k+1}), h(\mathbf{c}(\mathbf{x}_{k+1})))$.
            ENDIF
          ELSE
            • Set $\alpha_k = \alpha$, $\Delta f_k = \Delta f$,
              $\Delta l_k^{(\alpha)} = \Delta l^{(\alpha)}$.
            • Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k + \alpha_k^2 \widetilde{\mathbf{d}}_k$.
            • If $h(\mathbf{c}(\mathbf{x}_{k+1})) > 0$ then set
              $\mathcal{F}^{(k+1)} = \mathcal{F}^{(k)} \cup \{k+1\}$
              and remove any points from
              the filter that are dominated by
              $(f(\mathbf{x}_{k+1}), h(\mathbf{c}(\mathbf{x}_{k+1})))$.
          ENDIF
        ENDIF
      ENDIF
    ENDIF
  UNTIL $\alpha < \alpha_{\min}$ or $\mathbf{x}_{k+1}$ is found.
ENDIF
IF $\alpha < \alpha_{\min}$ THEN
  Goto Feasibility Restoration Phase to find $\mathbf{x}_{k+1}$
  so that it is acceptable to the filter and the
  $QP(\mathbf{x}_{k+1})$ subproblem is feasible.
ENDIF
Set $k := k + 1$.
UNTIL convergence criterion is met.
```

From the pseudo-code, at every iteration k there is an inner loop in which backtracking strategy is used, where decreasing values of α are generated. The inner loop terminates when the algorithm satisfies either one of the following scenarios:

(a) the trial point $\mathbf{x}_k + \alpha \mathbf{d}_k$ or $\mathbf{x}_k + \alpha \mathbf{d}_k + \alpha^2 \widetilde{\mathbf{d}}_k$ is accepted to be a new iterate;

(b) the step size $\alpha < \alpha_{\min}$.

Furthermore our algorithm also provides an outlet if the current $QP(\mathbf{x}_k)$ subproblem is infeasible or if the backtracking strategy fails to improve either the objective function or the constraints violation function values. We do this by exiting the algorithm temporarily and enter into a feasibility restoration phase where the main purpose is to reduce the constraints infeasibility. The whole process terminates if the restoration phase finds a point that is both acceptable to the filter, and for which the QP subproblem is feasible.

The main crux of our feasibility restoration phase is to solve a norm minimization problem of the form

$$H \left\{ \begin{array}{ll} \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} & h(\mathbf{c}(\mathbf{x})) \end{array} \right.$$

where Problem H is a non-smooth optimization problem. To promote global convergence and also fast local convergence of iterates, following [7] in the development of a Sequential Non-Smooth Programming (SNQP) method, our strategy solves the following QP subproblem at every iteration

$$\widehat{QP}(\mathbf{x}_k) \left\{ \begin{array}{ll} \underset{\mathbf{d} \in \mathbb{R}^n}{\text{minimize}} & l(\mathbf{d}) + \frac{1}{2}\mathbf{d}^T \mathbf{B}_k \mathbf{d} \\ \text{subject to} & \|\mathbf{d}\|_\infty \le \rho \end{array} \right.$$

where $\mathbf{x}_k$ denotes the k-th iterate, $\mathbf{d}$ is a displacement vector, $l(\mathbf{d}) = h(\mathbf{c}(\mathbf{x}_k) + \nabla \mathbf{c}(\mathbf{x}_k)^T \mathbf{d})$, $\mathbf{B}_k$ is an approximation of the Hessian of the Lagrangian and ρ is a trust region radius.

As in all trust region based methods we define

$$\Delta h = h(\mathbf{c}(\mathbf{x}_k)) - h(\mathbf{c}(\mathbf{x}_k + \mathbf{d}))$$

as the *actual reduction* in $h(\mathbf{c}(\mathbf{x}_k))$, and let

$$\Delta q = h(\mathbf{c}(\mathbf{x}_k)) - h(\mathbf{c}(\mathbf{x}_k) + \nabla \mathbf{c}(\mathbf{x}_k)^T \mathbf{d}) - \frac{1}{2}\mathbf{d}^T \mathbf{B}_k \mathbf{d}$$

be the *quadratic predicted reduction* in $h(\mathbf{c}(\mathbf{x}_k))$. In order for the trial step $\mathbf{x}_k + \mathbf{d}$ to be accepted by the algorithm, we require it to satisfy the simple sufficient reduction condition

$$\Delta h \ge \zeta \Delta q$$

where $\Delta q \ge 0$ for a suitable positive-definite $\mathbf{B}_k$ and $\zeta \in (0,1)$ is a pre-assigned parameter. Given all the necessary definitions, we are now in a position to state the trust region feasibility restoration phase by means of the following pseudo-code.

Feasibility Restoration Phase

Given $\mathbf{x}_k$, set $\rho_{\min} > 0$, $\zeta \in (0,1)$ and $\rho \ge \rho_{\min}$. Set $\mathcal{V}(\mathbf{x}_k) = \{i : c_i(\mathbf{x}_k) > 0\}$ and $\mathcal{V}^\perp(\mathbf{x}_k) = \{i : c_i(\mathbf{x}_k) \le 0\}$. Stop if $\mathcal{V}(\mathbf{x}_k) = \emptyset$ (all constraints are feasible) and return to filter line search SQP algorithm.

```
REPEAT
  Solve $\widehat{QP}(\mathbf{x}_k, \rho)$ subproblem to obtain a local
  minimizer $\mathbf{d}$.
  IF $\Delta h < \zeta \Delta q$ THEN
    • Set $\rho := \frac{1}{2}\rho$.
  ELSE
    • Set $\widehat{\mathbf{d}}_k = \mathbf{d}$, $\rho_k = \rho$, $\Delta h_k = \Delta h$, $\Delta q_k = \Delta q$.
```

Proceedings of the 9th WSEAS International Conference on Applied Mathematics, Istanbul, Turkey, May 27-29, 2006 (pp372-378)

```
        • Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$.
        • Set $\mathcal{V}(\mathbf{x}_{k+1}) = \{i : c_i(\mathbf{x}_{k+1}) > 0\}$ and
          $\mathcal{V}^{\perp}(\mathbf{x}_{k+1}) = \{i : c_i(\mathbf{x}_{k+1}) \le 0\}$.
        • Set $\rho = \begin{cases} \rho_k & \text{if } \|\widehat{\mathbf{d}}_k\|_\infty < \rho_k, \\ 2\rho_k & \text{otherwise.} \end{cases}$
        • Set $\rho \ge \rho_{\min}$ if $\rho < \rho_{\min}$.
        • Set $k := k + 1$.
    ENDIF
UNTIL $\mathbf{x}_k$ is acceptable to the filter or convergence criterion is met.
```

Note that in the restoration phase there is always a possibility that the restoration phase might fail to terminate and converge to an infeasible point. An example of this behaviour could happened if there exists a non-zero local minimum of $h(\mathbf{c}(\mathbf{x}))$ which indicates that the original problem P is locally infeasible. On the other hand, if the restoration phase is converging to a feasible point then due to the filter acceptance test it is usually likely that the restoration phase will terminate and returns back to the main filter algorithm.

3 Numerical Performance of filLS

In this section we describe the performance of our code filLS and the following are the details regarding the implementation of the algorithm:

- The code filLS has been implemented in C++ with double precision. In addition, the code is also interfaced with CPLEX version 9.0 and AMPL.

- The Hessian matrices W_k and B_k are derived from the Hessian of the Lagrangian function of Problem P and Problem H respectively. In the event the matrices are indefinite we then perturb them by using a modified Cholesky factorization method [2] to make them positive-definite.

- For parameter values in the main filter algorithm we set $t = 0.99$, $\sigma = 10^{-4}$, $\eta = 10^{-3}$, $\gamma = 10^{-3}$ and $\alpha_{\min} = 10^{-5}$ for $h(\mathbf{c}(\mathbf{x})) \neq 0$. As for the feasibility restoration phase, we choose the initial trust region $\rho_{\text{ini}} = 5$, $\rho_{\min} = 10^{-4}$ and $\zeta = 10^{-4}$. In addition we set the tolerance level $\epsilon_{\text{tol}} = 10^{-6}$ and the maximimum iterations permitted is $k_{\max} = 500$.

- As for the convergence criteria, the KKT error, the constraint violation $h(\mathbf{c}(\mathbf{x}))$ and the ℓ_∞ norm of the QP or SOC step are computed. We terminate the iteration when the above conditions are satisfied to an accuracy of ϵ_{tol} or $k = k_{\max}$.

- The algorithm may also terminate in the feasibility restoration phase and this occurs when the restoration phase can no longer reduce the constraint infeasibility function. Hence, the algorithm stops if the iteration $k = k_{\max}$ or the KKT error of Problem H and the ℓ_∞ of the step are less then ϵ_{tol}

To analyze the performance of our algorithm we will compare it with LOQO version 6.06 and SNOPT version 6.1 where both of these solvers utilize backtracking line search techniques and also merit functions to promote global convergence. The selection of large scale CUTE test problems are listed in Table 1.

Selected CUTE Problems		
$80 \le n < 500$	$500 \le n < 1000$	$n \ge 1000$
AIRPORT	BDVALUE	CATENARY
BRATU2D	CBRATU2D	CHEMRCTA
BRATU3D	CBRATU3D	CHEMRCTB
BRITGAS	CORKSCREW	GILBERT
CHANDHEQ	EIGENA2	MANNE
CLNBEAM	GAUSSELM	MINPERM
GROUPING	GPP	OPTMASS
HVYCRASH	HADAMARD	READING1
MINC44	OPTCDEG2	SVANBERG
NGONE	OPTCDEG3	UBH5

Table 1 Selection of 30 large scale CUTE test problems

In this paper we only discuss the statistics concerning the performance of the three codes and to simplify the presentation, if filLS, LOQO or SNOPT terminates before reaching a local solution of Problem P, the following notations are used:

-E- An arithmetic error occurred causing the code to fail

-I- The nonlinear problem was found to be infeasible

-F- Nonlinear contraints were found to be locally infeasible

-M- The run was terminated after reaching the maximum number of iterations.

By using the failure type notation, Table 2 gives a breakdown of the results of our runs.

	Number of Problems		
Failure type	fiILS	LOQO	SNOPT
-E-	0	0	1
-I-	0	0	5
-F-	4	0	0
-M-	0	5	0
Total number of failures	4	5	6
Total number of successful runs	26	25	24

Table 2 Types of failure for fiILS, LOQO and SNOPT on 30 large scale CUTE test problems

As regards on the performance of fiILS on NLP problems we feel it is very encouraging as the code is able to solve 26 out of the 30 test problems and furthermore it outperforms both LOQO and SNOPT. The test problems which our code failed to solve are HVYCRASH, CORKSCRW, GAUSSELM and CATENARY while for LOQO the test problems which caused it to exceed its maximum number of iterations ($k_{\max} = 500$) are MINC44, GAUSSELM, CHEMRCTA, CHEMRCTB and MANNE. Finally for SNOPT, the test problems which it failed to find local solutions are CBRATU3D, HADAMARD, GILBERT, UBH5, SVANBERG and CATENARY. Although more tests are needed to reach a more rigorous conclusion, the preliminary results do show that the filter concept together with line search and trust region strategies can be an attractive choice.

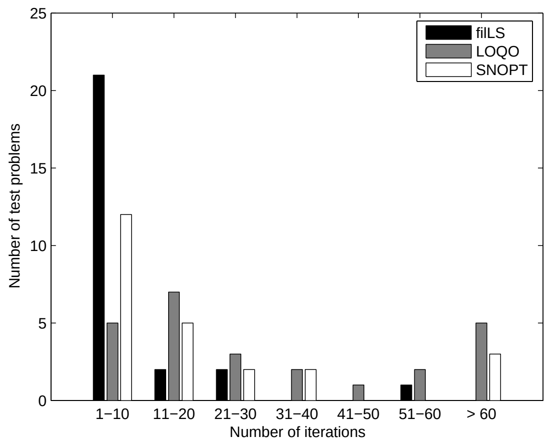

Figure 2: Comparing the number of iterations for selected large scale CUTE test problems

Another encouraging aspect of fiILS can be seen by comparing the number of solved problems for a certain range of number of iterations and function evaluations. In Figure 2, we can see a majority of the problems solved by fiILS required less than 10 iterations while only a small proportion of the problems solved by LOQO belong to that category. The same efficiency is observed when comparing our code with SNOPT where the latter tends to outperform LOQO in some categories. In addition such a scenario is also repeated in Figure 3 if we were to compare the number of problems solved for a certain range of function evaluations. Overall, from an efficiency point of view, we feel our code is as efficient as either LOQO and SNOPT to solve large scale NLP problems.

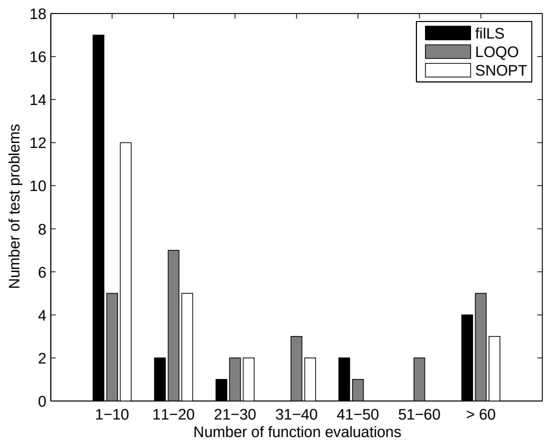

Figure 3: Comparing the number of function evaluations for selected large scale CUTE test problems

4 Conclusion

A prototypical algorithm of applying filter strategy in line search SQP methods has been described, demonstrating the fact that convergence for NLP can be achieved without the need to maintain sufficient descent in a traditional penalty type merit function approach. From the numerical tests, the code fiILS is relatively suitable for solving nonlinear optimization problems including large scale problems. In terms of number of successful run problems, the code fiILS outperforms LOQO and SNOPT and we believe this is due to the filter strategy which is more flexible in accepting iterates, and also to the used of feasibility restoration phase to generate iterates which are close to the feasible region. As for efficiency and reliability, the code fiILS in our view is therefore suitable for large scale optimization and is well-suited to provide the basis of a commercial NLP code.

References:

[1] C. Audet and J.E. Dennis Jr, A pattern search filter method for nonlinear programming without derivatives, Technical Report, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2000.

[2] S.H. Cheng and N.J. Higham, A modified Cholesky algorithm based on a symmetric indefinite factorization, *SIAM Journal on Matrix Analysis and Applications*, 19, 1998, pp. 1097 - 1110.

[3] C.M. Chin and R. Fletcher, On the global convergence of an SLP–filter algorithm that takes EQP steps, *Mathematical Programming*, 96(1), 2003, pp. 161 – 177.

[4] C.M. Chin, A.H.A. Rashid and K.M. Nor, Global and local convergence of a filter line search method for nonlinear programming, *Optimization Methods and Software*, 2006 (to appear).

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

[6] R. Fletcher and S. Leyffer, Nonlinear programming without a penalty function, *Mathematical Programming*, 91(1), 2002, pp. 239 – 269.

[7] R. Fletcher and E. Sainz de la Maza, Nonlinear programming and nonsmooth optimization by successive linear programming, *Mathematical Programming*, 43, 1989, pp. 235 – 256.

[8] E.R. Panier, A.L. Tits, A superlinear convergent feasible method for the solution of inequality constrained optimization problems, *SIAM Journal on Control and Optimization*, 25(4), 1987, pp. 934 – 950.

[9] M. Ulbrich, S. Ulbrich and L.N. Vicente, A globally convergent primal-dual interior-point filter method for nonlinear programming, *Mathematical Programming*, 100(2), 2004, pp. 379 – 410.

[10] A. Wächter and L.T. Biegler, Global and local convergence of line search filter methods for nonlinear programming, CAPD Technical Report B-01-09, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, 2001.