Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 10, No. 1, pp. 229–242, January 2014

A PROJECTED GRADIENT FILTER TRUST-REGION ALGORITHM FOR BOUND CONSTRAINED OPTIMIZATION*

M. Fatemi and N. Mahdavi-Amiri†

Abstract: We present a filter trust-region algorithm for solving bound constrained optimization problems. The algorithm is an extension of our method recently proposed for unconstrained optimization to consider the bound constraints. The new algorithm combines the gradient projection method with the filter technique to generate non-monotone iterations. In contrast with the earlier filter algorithms, the new algorithm has the novelty to ensure the finiteness of the filter size. Global convergence to at least one first order critical point is established. Comparative numerical results on a set of test problems from the CUTEr collection show the algorithm is competitive and more efficient solely with respect to the filter size.

Key words: *bound constrained optimization , filter methods, trust-region algorithms, gradient projection method*

Mathematics Subject Classification: *90C52, 65K05, 49M37, 26B25*

1 Introduction

We consider the bound constrained minimization problem,

$$\begin{aligned} &\min f(x) \\ &s.t. \quad l \le x \le u, \end{aligned} \tag{1.1}$$

where, $f(x)$ is a twice continuously differentiable function and $l,\ u \in \mathbb{R}^n$ with $-\infty \le l_i \le u_i \le +\infty$, for $i = 1, \ldots, n$. There are several effective algorithms for solving this problem. The most famous of them are active set methods [6, 8] and gradient projection methods [3, 5, 28]. In the active set methods, the working set changes slowly and this leads to a large number of iterations specially on large scale problems. In order to overcome the disadvantage of the active set methods, several authors proposed combining of the active set strategy with the gradient projection methods [1, 2, 4, 9, 10, 12, 18, 19, 23]. The gradient projection methods allow for the working set to change rapidly, reducing the number of iteration required to convergence. See [13] for a great survey on simple bound optimization.

The idea of the use of filter was first introduced by Fletcher and Leyffer [17], and later extended by others [7,16,26,27]. More recently, this idea has been used by Gould et al. [20,21] to solve nonlinear least squares and unconstrained minimization problems. An extension of

*The first and second authors respectively thank the Research Council of K. N. Toosi University of Technology and Sharif University of Technology for supporting this work.

†Corresponding author.

the method, given in [21], to simple bound constrained problems was subsequently proposed by Sainvitu and Toint in [25].

Here, we combine the gradient projection method with the filter technique and present a filter trust-region algorithm for bound constrained optimization. We extend the unconstrained method introduced by Fatemi and Mahdavi-Amiri [14] to the case of bound constrained optimization problems. The main feature of our algorithm is in fact using a new filter technique to ensure global convergence. This technique guarantees the finiteness of the filter size [15]. The remainder of our work is organized as follow. Section 2 gives a brief description of the approximating model. Section 3 explains some basic concepts of the multidimensional filter. Our new algorithm is given in Section 4. The global convergence of the algorithm to at least one first order critical point is established in Section 4.1. Some comparative computational results are illustrated in Section 5. Finally, we conclude in Section 6.

2 Step Length Calculation

As is common in trust-region algorithms, a trial step length s is computed by first choosing an approximating model of the objective function and then minimizing the model in the presence of a trust-region constraint so that the trial point $x_k^+ = x_k + s$ is a feasible point for the original problem. By this trust-region constraint, we aim to trust the model to be an adequate representation of the objective function. We choose the quadratic approximation model as follow

$$m_k(x_k + s) = f(x_k) + g_k^T s + \frac{1}{2} s^T H_k s, \tag{2.1}$$

where, $g_k = \nabla f(x_k)$ and H_k is a symmetric approximation of the Hessian of the objective function in the current iterate x_k. To find a trial step length, we minimize (2.1) subject to the following constraints

$$l \le x_k + s \le u, \tag{2.2}$$

$$\|s\| \le \Delta_k, \tag{2.3}$$

where, $\|.\|$ is an arbitrarily chosen norm and Δ_k is the trust-region radius. We note that (2.2) ensures the feasibility of the trial point $x_k + s$. Here, similar to Gould et al. [20, 21], we do not necessarily need to uphold (2.3) for every iteration.

It is convenient to choose the infinity norm for the trust-region constraint (2.3), because the shape of the trust-region is aligned with the simple bound constraint (2.2) and we can replace (2.2) and (2.3) by the box constraints

$$(l_k)_i \stackrel{\text{def}}{=} \max(l_i, (x_k)_i - \Delta_k) \le (x_k + s)_i \le \min(u_i, (x_k)_i + \Delta_k) \stackrel{\text{def}}{=} (u_k)_i, \tag{2.4}$$

for $i = 1, \ldots, n$. The approximate solution of the trust-region subproblem, as is common in the trust-region algorithms, needs to provide a sufficient decrease in the model in the sense of the following inequality

$$m_k(x_k) - m_k(x_k + s_k) \ge \kappa_{mdc} \chi_k \min(\frac{\chi_k}{\beta_k}, \Delta_k), \tag{2.5}$$

where, $\kappa_{mdc} \in (0, 1)$, $\beta_k = 1 + \|H_k\|$ and χ_k is a first order criticality measure; see [8, chapter 8]. In the context of unconstrained optimization, $\chi_k = \|g_k\|$ is obviously one of the many suitable criticality measures. A possible criticality measure for problem (1.1) is

$$\chi_k \stackrel{\text{def}}{=} \|\bar{g}(x_k)\|_\infty, \tag{2.6}$$

where,

$$\bar{g}(x_k) = x_k - p[x_k - g_k, l, u] \tag{2.7}$$

is called the projected gradient of the objective function into the feasible box and $p[x, l, u]$ is the projection operator defined by

$$p[x, l, u]_i = \begin{cases} l_i & \text{if } x_i \leq l_i \\ x_i & \text{if } x_i \in (l_i, u_i) \\ u_i & \text{if } x_i \geq u_i, \end{cases} \tag{2.8}$$

(see e.g., [8, chapter 8] and [9]).

In order to satisfy (2.5), we need to find the Generalized Cauchy Point (GCP); see [9,10]. This point is the first local minimizer of the univariate function

$$m_k(p[x_k - tg_k, l_k, u_k]).$$

There are several efficient algorithm for the GCP calculation; see [10, 22, 24]. In these algorithms, in order to provide a fast asymptotic rate of convergence, after computing GCP, the variables non-active at the GCP is determined and then the model m_k is further minimized in the box (2.4) over the subspace corresponding to the non-active variables. This process can efficiently be done by a conjugate gradient based algorithm.

3 The Multidimensional Filter

A filter is simply a data structure whose mission is to store pertinent information on the past iterates for use in determining the new iterates. Before giving a formal definition, we briefly recall some facts from [20].

If we focus temporarily on the finding of a stationary point for problem (1.1), then we can compute such a point by the following minimization problem:

$$\min \sum_{j=1}^{p} \theta_j(x), \tag{3.1}$$

where,

$$\theta_j(x) = \|\bar{g}_{\mathcal{I}_j}(x)\|, \quad \text{for } j = 1, \ldots, p,$$

and the $\bar{g}_{\mathcal{I}_j}(x)$ form a partition of $\bar{g}(x)$ (not necessarily disjoint) into sets $\{\bar{g}_i(x)\}_{i \in \mathcal{I}_j}$, with $\bar{g}_i(x)$ as the ith component of $\bar{g}(x)$ and $\mathcal{I}_1 \cup \mathcal{I}_2 \ldots \cup \mathcal{I}_p = \{1, \ldots, n\}$.

Let $\theta(x_k) = (\theta_1(x_k), \ldots, \theta_p(x_k))$. Then, it is easy to see that for an arbitrarily chosen norm, there exist some positive constants κ_l and κ_u such that

$$\kappa_l \|\bar{g}_k\|_\infty \leq \|\theta(x_k)\| \leq \kappa_u \|\bar{g}_k\|_\infty. \tag{3.2}$$

The minimization problem (3.1) can be seen as a p-criteria optimization problem. Hence, we define filter as a list $\mathcal{F}$ of p-tuples $\theta(x_k)$ so that if $\theta(x_k)$ and $\theta(x_l)$, with $k \neq l$, belong to the filter, then for at least one $j \in \{1, \ldots, p\}$,

$$\theta_j(x_k) < \theta_j(x_l).$$

As a data structure, we may wish to add a point to the filter or remove a point from it. Our strategy is inspired by that of [14]: we say that a new trial point x_k^+ is acceptable for the filter if for all $\theta(x_l) \in \mathcal{F}$, there exists $j \in \{1, \ldots, p\}$ such that

$$\theta_j(x_k^+)^{\mu_2} + \lambda_2 \|\theta(x_k^+)\|^{\mu_1} \leq \theta_j(x_l)^{\mu_2} + \lambda_1 \|\theta(x_l)\|^{\mu_1}, \tag{3.3}$$

where, $\|.\|$ is a Euclidean norm and $\lambda_1, \lambda_2, \mu_1$ and μ_2 are constants so that

$$0 < \lambda_1 < \lambda_2 < \frac{1}{\sqrt{p}} \quad \text{and} \quad 0 < \mu_1 < \mu_2.$$

Knowing this, we add a point to the filter only if it is acceptable for the filter. As we will show in Section 4.1 (Lemma 4.7), this adding strategy guarantees the finiteness of the filter size, a property that can not be guaranteed by the earlier filter algorithms.

It is also possible to remove a point from the filter. We remove $\theta(x_l)$ from the filter and replace it with $\theta(x_k^+)$ if (3.3) is satisfied for all index $j \in \{1, \ldots, p\}$. In this case, we say that $\theta(x_k^+)$ dominates $\theta(x_l)$ from the filter.

Finally, note that we can also use the extra removing procedure introduced in [14] to further control the filter size. Here, we skip the details and refer the interested reader to [14].

4 The Algorithm

Here, we combine the filter technique with the gradient projection strategy to introduce the following gradient projection filter trust-region algorithm. This algorithm is a modification of the algorithm proposed in [14].

Algorithm 4.1. Gradient Projection Filter Trust-Region Algorithm (GPFTRA).

Step 0:{**Initialization**}

Give an initial point x_0, a small tolerance $\epsilon > 0$, an initial symmetric matrix H_0, an initial trust-region radius $\Delta_0 > 0$, the constants $\lambda_1, \lambda_2, \mu_1$ and μ_2 satisfying

$$0 \leq \lambda_1 < \lambda_2 < \frac{1}{\sqrt{p}}, \quad 0 < \mu_1 < \mu_2,$$

with $\eta_1, \eta_2, \gamma_1, \gamma_2$ and γ_3 satisfying

$$0 < \eta_1 < \eta_2 < 1, \quad 0 < \gamma_1 < \gamma_2 < 1 \leq \gamma_3.$$

Choose $f_{sup} \geq f(x_0)$.
$k = 0$.
RESTRICT=**false**.
$\mathcal{F} = \emptyset$.
repeat
  Step 1:{**Computing a trial step**}
  if RESTRICT=**true then**
    {**minimize within trust-region**}
    Compute s_k by approximately minimizing (2.1) such that $x_k + s_k$ satisfies (2.4).
  else
    Start minimizing (2.1) subject to (2.2).
    if negative curvature is discovered **then**
      {**minimize within trust-region**}
      Compute s_k by approximately minimizing (2.1) such that $x_k + s_k$ satisfies (2.4).
    else
      {**minimize disregarding trust-region**}
      Compute s_k by approximately minimizing (2.1) subject to (2.2).
    end if

end if
$x_k^+ = x_k + s_k$.
Compute the ratio,

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

Step 2:{**Test trial step for acceptance**}
if $f(x_k^+) \leq f_{sup}$ **then**
  if $\rho_k \geq \eta_1$ **and** $\|s_k\|_\infty \leq \Delta_k$ **then**
    $x_{k+1} = x_k^+$.
    RESTRICT = **false**.
    $f_{sup} = f(x_{k+1})$;
  else if x_k^+ is acceptable for the filter **then**
    Add $\theta(x_k^+)$ to the filter.
    $x_{k+1} = x_k^+$.
    RESTRICT = **false**.
  else
    $x_{k+1} = x_k$.
    RESTRICT=**true**.
  end if
else
  $x_{k+1} = x_k$.
  RESTRICT=**true**.
end if
Step 3:{**Updating the trust-region radius**}
if $\|s_k\|_\infty \leq \Delta_k$ **then**
  Choose

$$\Delta_{k+1} \in \begin{cases} [\gamma_1\Delta_k, \gamma_2\Delta_k] & \text{if } \rho_k < \eta_1 \\ [\gamma_2\Delta_k, \Delta_k] & \text{if } \rho_k \in [\eta_1, \eta_2) \\ [\Delta_k, \gamma_3\Delta_k] & \text{if } \rho_k \geq \eta_2 \end{cases}$$

else
  $\Delta_{k+1} = \Delta_k$.
end if
$k = k + 1$.
until ($\|\bar{g}_k\|_\infty < \epsilon$ **or** k is larger than a user defined value)

As can be seen in Step 1 of Algorithm 4.1, if a negative curvature is discovered, we must recompute s_k by considering the trust-region constraint (2.3).

4.1 Global Convergence Analysis

Here, we analyze the global convergence properties of Algorithm 4.1 as applied to problem (1.1).

The following assumptions are commonly used in convergence analysis of the filter algorithms.
A1. f is a twice continuously differentiable function on $\mathbb{R}^n$.
A2. The sequence $\{x_k\}$ generated by the algorithm is contained in a compact subset of $\mathbb{R}^n$.
A3. There exists a constant $\kappa_{umh} \geq 1$ such that for all k,

$$\|H_k\| = \beta_k - 1 \leq \kappa_{umh} - 1.$$

Assumptions A1, A2 imply that there exists $\kappa_{ufh} \geq 1$ such that

$$\|\nabla^2_{xx} f(x_k)\| \leq \kappa_{ufh},$$

for all k. Consequently, assumption A3 can be satisfied for the choice $H_k = \nabla^2_{xx} f(x_k)$.

Let

$$\mathcal{S} \stackrel{\text{def}}{=} \{k \mid x_{k+1} = x_k + s_k\}$$

be the set of successful iterations,

$$\mathcal{D} \stackrel{\text{def}}{=} \{k \mid \rho_k \geq \eta_1 \text{ and } \|s_k\|_\infty \leq \Delta_k\}$$

be the set of sufficient decrease iterations, and

$$\mathcal{A} \stackrel{\text{def}}{=} \{k \mid \theta(x_k^+) \text{ is added to the filter}\}$$

be the set of the filter iterations. It is easy to see that

$$\mathcal{S} = \mathcal{D} \cup \mathcal{A}. \tag{4.1}$$

We first characterize the first order stationary conditions for problem (1.1).

Theorem 4.1. *A feasible point x^* is a first order critical point for problem (1.1) if and only if*

$$\bar{g}(x^*) = 0.$$

Proof. See theorems 12.1.2 and 12.1.3 in [8]. □

In the next step for our convergence analysis, we need to recall a basic result concerning the trust-region radius.

Lemma 4.2. *Suppose that assumptions A1-A3 hold and that there exists a constant $\kappa_{lbg} > 0$ such that $\chi_k \geq \kappa_{lbg}$, for all k. Then, there exists a constant $\kappa_{lbd} > 0$ such that*

$$\Delta_k \geq \kappa_{lbd}.$$

Proof. It is easy to see that Lemma 3.2 and Lemma 3.3 of [25] still hold. Thus, the proof is identical to the proof of Lemma 3.4 in [25]. □

We now consider the case of $|\mathcal{S}| = \infty$ and restrict our attention to the case of infinitely many filter iterations.

Lemma 4.3. *If A1-A3 hold and $|\mathcal{A}| = \infty$, then*

$$\liminf_{k\to\infty} \chi_k = 0. \tag{4.2}$$

Proof. The proof is similar to the first part of the proof of Theorem 1 in [14] except that $g(x_k)$ is replaced by $\theta(x_k)$ and that we use the new filter acceptance criterion (3.3). We can then show that

$$\liminf_{k\to\infty} \|\theta(x_k)\| = 0.$$

Thus, equality (4.2) is a straightforward result of (2.6) and (3.2). □

We now concentrate on the case of finitely many filter iterations and show that the number of sufficient decrease iterations should be finite unless a first order critical point is approached.

Lemma 4.4. *Suppose that $|\mathcal{A}| < \infty$ and A1-A3 hold and there exists a constant $\kappa_{lbg} > 0$ such that $\chi_k \geq \kappa_{lbg}$, for all k. Then, there can be only finitely many sufficient decrease iterations; i.e., $|\mathcal{D}| < \infty$.*

Proof. The proof is identical to the proof of Theorem 2 in [14] except that $\|g_k\|$ is replaced by χ_k. □

Now, we consider the case of $|\mathcal{S}| < \infty$ and prove the first order criticality of the limit points of the sequence of the iterates.

Theorem 4.5. *Suppose that A1-A3 hold and that there are only finitely many successful iterations; i.e., $|\mathcal{S}| < \infty$. Then, $x_k = x^*$, for all sufficiently large k, and x^* is first order critical.*

Proof. The proof is the same as the proof of Theorem 3.6 in [21] except that $\|g_k\|$ is replaced by χ_k. □

Now, we have the following main result.

Theorem 4.6. *Suppose that assumptions A1-A3 hold. Then, either $\chi_k = 0$ for some finite k, or*

$$\liminf_{k\to\infty} \chi_k = 0.$$

Proof. The proof follows from lemmas 4.3 and 4.4, Theorem 4.5 and using (4.1). □

The above result implies that at least one of the limit points of the sequence of the iterates generated by Algorithm 4.1 is a first order critical point. Furthermore, as Example 4.1 in [14] indicated, we can not improve this result to the case of the first order criticality of all the limit points without modifying the filter mechanism.

As a final result, we show that our filter acceptance criterion based on (3.3) ensures the finiteness of the filter size.

Lemma 4.7. *Let $\{\theta_{k+1}\}$ with $k \in \mathcal{A}$ be the subsequence of all the points added to the filter,*

$$0 < \mu_1 < \mu_2,\ 0 < \lambda_1 < \lambda_2 < \frac{1}{\sqrt{p}},$$

and m be the smallest integer satisfying

$$(1 + \lambda_2)\epsilon^m < \lambda_1,$$

for some $\epsilon \in (0, 1)$. Then, the filter size is finite.

Proof. The proof is similar to the proof of Lemma 4 in [14] except that g_k, θ_1 and θ_2 are respectively replaced by θ_k, λ_1 and λ_2. □

Remark 4.8. The idea of introducing the reference iteration f_{sup} (see Step 2 of Algorithm 4.1) is inspired by Algorithm 2.1 in [25]. In contrast with the algorithm in [25], we can remove the process of determining the reference iteration from Algorithm 4.1 without sacrificing the convergence guarantee. In other words, both the first " **if** ... **else** ... **endif** " statement and also the term " $f_{sup} = f(x_{k+1})$ " can be removed from the Step 2 of Algorithm 4.1. By the reference iteration, we only ensure the existence of a decreasing subsequence of $\{f(x_k)\}$, increasing the chance to converge to a minimal point. We must note that although this strategy seems to be useful, it does not guarantee convergence to a second order critical point. Hence, a modified version of Algorithm 4.1 without the definition of the reference iteration can be used simply to solve for example convex problems, because a zero criticality measure is both necessary and sufficient for second order criticality of the convex problems. In the next section, we investigate the effect of removing the reference iteration on our numerical results.

5 Numerical experiments

Here, we test our algorithm on a set of 108 simple bound constrained problems from the CUTEr collection. We chose test problems with the same names and dimensions as specified in Table 4.1 in [25]. In our numerical tests, we compared our algorithm with the filter trust-region algorithm (FTRA) of [25]. The codes were written in MATLAB and run on a PC with a 2.4 GHz Intel Core 2Duo CPU and 2 GB of memory under ubuntu 10.04 Linux operating system. In order to find a suitable starting point, we project the initial point supplied by the problem onto the feasible region. All attempts to solve the test problems were limited to a maximum of 5000 iterations or 1 hour of CPU time.

In our tests, we chose the exact Hessian of the objective function as the Hessian of the approximating model. A step length s is computed by approximately minimizing the model with the algorithm presented in [10]. This algorithm is terminated at the first s for which,

$$\|\nabla m_k(x_k + s)_{free}\| \leq \min(0.1, \sqrt{\|\bar{g}_k\|})\|\bar{g}_k\|,$$

where, $\nabla m_k(x_k + s)_{free}$ denotes the restricted gradient of the model corresponding to the free variables. The initial parameters were chosen to be: $\gamma_1 = 0.0625$, $\gamma_2 = 0.25$, $\gamma_3 = 2$, $\eta_1 = 0.01$, $\mu_1 = \eta_2 = 0.9$, $\mu_2 = 1$, $\Delta_0 = 1$, $\epsilon = 10^{-6}$, $\lambda_2 = 2\lambda_1$ and

$$\lambda_1 = \min(0.001, \frac{1}{2\sqrt{p}}).$$

Moreover, we chose $p = n$ and $\mathcal{I}_j = \{j\}$ for our tests.

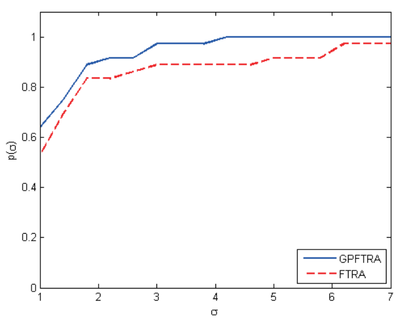

Fig. 1 *Filter size performance profile.*

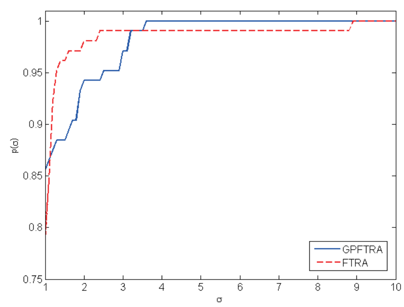

Fig. 2 *Iteration performance profile.*

The two algorithms successfully solved 103 problems and failure occurred on BIGGSB1, MINSURFO, PALMER7A, QRTQUAD and SCOND1LS, because the maximal iteration

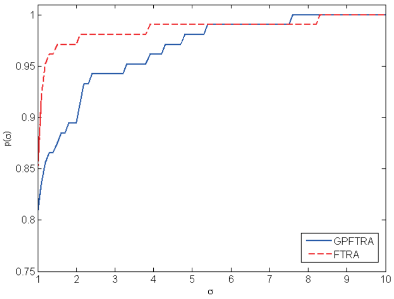

Fig. 3 *Conjugate gradient iteration performance profile.*

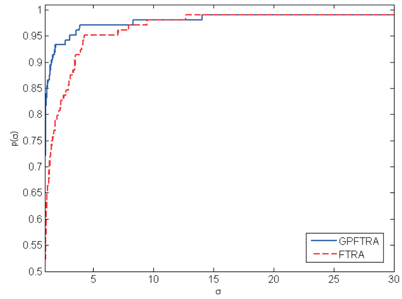

Fig. 4 *CPU performance profile.*

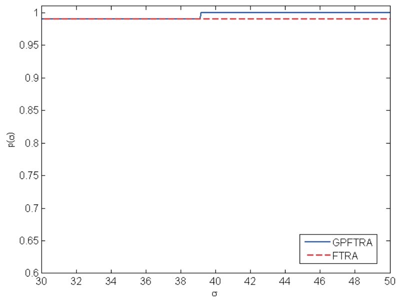

Fig. 5 *CPU performance profile.*

count was reached before having the chance to detect convergence. In Table 1 and Table 2, we report the results obtained by applying FTRA and GPFTRA on the whole set of the test problems. In the tables, we let *iter* denote the number of iterations, CPU denote the CPU time, CG denote the number of conjugate gradient iterations and f^* denote the final objective function value.

As Table 1 and Table 2 show the efficiency of the two algorithms is averagely the same. We have better results of FTRA over GPFTRA on problems EXPQUAD, NCVXBQP2, NCVXBQP3, PAMER1A, PALMER2A and YFIT. The FTRA algorithm has also produced the better result with respect to the total amount of the conjugate gradient iteration than the GPFTRA algorithm on problems PALMER2B, PALMER3, PALMER5E and QR3DLS. We note that on these problems, GPFTRA has a better total number of iterations than FTRA. The better result of GPFTRA over FTRA can be seen on problems PALMER4A, PALMER5A, PALMER5B, PSPDOC, QR3DLS, WEEDS. We should note that GPFTRA has produced better final objective function values than FTRA on problems HS1, HS38, QR3DLS and YFIT.

We have also used the performance profile of Dolan and Moré [11] to compare the efficiency of the two algorithms. As Fig. 1, indicates, GPFTRA is significantly more efficient than FTRA with respect to the filter size. Its efficiency is also better than FTRA with respect to the CPU time factor; see Fig. 4 and Fig. 5. Furthermore, Fig. 2 and Fig. 3 give the performance profiles for the number of iterations and total amount of conjugate gradient iterations. As these figures indicate, FTRA is more efficient than GPFTRA with respect to to these factors.

We finally investigated the effect of removing the reference iteration as described in Remark 4.8. In this case, we observed that on problems PSPDOC and SINEALI, the iterates finally converged to a point with the final objective function values 4.8×10^{16} and 8.5×10^{4} in 24 and 4 iterations, respectively. We also observed that on the average, the restriction to reference iteration undermines the efficiency of GPFTRA.

6 Conclusions

We proposed a projected gradient filter trust-region algorithm for solving bound constrained optimization problems. The new algorithm is a modified version of our recent algorithm proposed for unconstrained optimization. In contrast with the earlier filter algorithms, the new algorithm has the novelty to ensure the finiteness of the filter size. We showed, under standard assumptions, that at least one of the limit points of the sequence of the iterates generated by the algorithm was a first order critical point. Numerical comparative results on a set of bound constrained test problems from the CUTEr collection showed the algorithm is competitive and more efficient solely with respect to the filter size..

References

[1] M. Andretta, E.G. Birgin and J.M. Martínez, Practical active-set euclidian trust-region method with spectral projected gradients for bound-constrained minimization, *Optimization* 54 (2005) 305–325.

[2] E.G. Birgin and J.M. Martínez, Large-scale active-set box-constrained optimization method with spectral projected gradients, *Comput. Optim. Appl.* 23 (2002) 101–125.

[3] E.G. Birgin, J.M. Martínez and M. Raydan, Nonmonotone spectral projected gradient methods on convex sets, *SIAM J. Optim.* 10 (1999)1196–1211.

[4] E.G. Birgin and J.M. Martínez, A box constrained optimization algorithm with negative curvature directions and spectral projected gradients, *Computing Supplement* (2001). 49–60.

[5] E.G. Birgin, J.M. Martínez and M. Raydan, Inexact spectral projected gradient methods on convex sets, *J. Numer. Anal.* 23 (2003) 539–559.

[6] Z.W. Chen, J.Y. Han and D.C. Xu,A nonmonotone trust region method for nonlinear programming with simple bound constraints, *Appl Math Optim.* 43 (2001) 63–85.

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

[8] A.R. Conn, N.I.M. Gould and P.L. Toint, *Trust-Region Methods*, MPS/SIAM Series on Optimization, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[9] A.R. Conn, N.I.M. Gould and Ph. L. Toint, Global convergence of a class of trust region algorithms for optimization with simple bounds, *SIAM J. Numer. Analysis* 25 (1988) 764–767.

[10] A.R. Conn, N.I.M. Gould and Ph. L. Toint,, Testing a class of methods for solving minimization problems with simple bounds on the variables, *Mathematics of Computation* 50 (1988) 399–430.

[11] E.D. Dolan and J.J. Moré, Benchmarking optimization software with performance profiles, *Math. Program. Ser. A* 91 (2002) 201–213.

[12] Z. Dostál, Box constrained quadratic programming with proportioning and projections, *SIAM J. Optim.* 7 (1997) 871–887.

[13] Z. Dostál, *Optimal Quadratic Programming Algorithms*, Springer-Verlag, New York, 2009.

[14] M. Fatemi and N. Mahdavi-amiri, A filter trust-region algorithm for unconstrained optimization with strong global convergence properties, *Computational Optimization and Applications* 52 (2012) 239–266.

[15] M. Fatemi and N. Mahdavi-amiri, A non-monotone trust region algorithm for unconstrained optimization with dynamic reference iteration updates using filter, *Optimization* 61 (2012) 733–763.

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

[17] R. Fletcher and S. Leyffer, Nonlinear programming without a penalty function, Math. Program.Ser. A 91 (2002) 239–269.

[18] A. Friedlander and J.M. Martínez, On the maximization of a concave function with box constraints, *SIAM J. Optim.* 4 (1994) 177–192.

[19] A. Friedlander, J.M. Martínez and S.A. Santos, A new trust region algorithm for bound constrained minimization, *Applied Mathematics and Optimization* 30 (1994) 235–266.

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

[21] N.I.M. Gould, C. Sainvitu and P.L. Toint, A filter-trust-region method for unconstrained optimization,*SIAM J. Optim.* 16 (2005) 341–357.

[22] C. Lin and J.J. Moré, Newton's method for large bound-constrained optimization problems, *SIAM J. Optim.* 9 (1999) 1100–1127.

[23] J.J. Moré and G. Toraldo, On the solution of large quadratic programming problems with bound constraints, *SIAM J. Optim.* 1 (1991) 93–113.

[24] J. Nocedal and S.J. Wright, *Numerical Optimization*, second ed., Springer Series in Operations Research and Financial Engineering, Springer, New York, 2006.

[25] C. Sainvitu and P.L. Toint, A filter-trust-region method for simple-bound constrained optimization, *Optimization Methods Software* 22 (2007) 835–848.

[26] M. Ulbrich, S. Ulbrich and L.N. Vicente, A globally convergent primal-dual interior-point filter method for nonlinear programming, *Math. Program. Ser. A* 100 (2004) 379–410.

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

[28] Z. Yu, Solving bound constrained optimization via a new nonmonotone spectral projected gradient method, *Applied Numerical Mathematics* 58 (2008) 1340–1348.

Manuscript received 17 August 2011
revised 22 September 2012
accepted for publication 16 January 2013

Masoud Fatemi
Department of Mathematics, K. N. Toosi University of Technology, Tehran, Iran
Scientific Computations in Optimization and System Engineering (SCOPE)
K. N. Toosi University of Technology, Tehran, Iran
E-mail address: smfatemi@kntu.ac.ir

Nezam Mahdavi-Amiri
Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran
E-mail address: nezamm@sharif.edu

Table 1: Numerical comparisons on a subset of test problems.

Name	GPFTRA iter	CPU	CG	f^*	FTRA iter	CPU	CG	f^*
3PK	88	0.1482	594	1.72E+00	88	0.1602	594	1.72E+00
ALLINIT	9	0.0133	14	1.67E+01	9	0.0267	14	1.67E+01
BDEXP	2	0.9241	2	2.65E-123	2	0.9036	2	2.65E-123
BLEACHNG	5	24.3918	1	9.18E+03	5	26.0591	1	9.18E+03
BQP1VAR	2	0.0009	0	0.00E+00	2	0.0631	0	0.00E+00
BQPGABIM	4	0.0063	17	-3.79E-05	4	0.0190	17	-3.79E-05
BQPGASIM	5	0.0135	26	-5.52E-05	5	0.0251	26	-5.52E-05
BQPGAUSS	533	194.2316	6320	-3.63E-01	533	193.2728	6320	-3.63E-01
CAMEL6	7	0.0064	6	-1.03E+00	7	0.0224	6	-1.03E+00
CHEBYQAD	24	13.6575	72	1.01E+02	24	13.9096	72	1.01E+02
CHENHARK	8	29.2582	1190	-2.00E+00	8	29.2351	1190	-2.00E+00
CVXBQP1	2	4.1307	5	2.25E+04	2	4.2049	5	2.25E+04
DECONVB	26	0.0844	67	3.41E-09	26	0.0985	67	3.41E-09
EG1	6	0.0063	5	-1.13E+00	6	0.0195	5	-1.13E+00
EXPLIN	50	15.5694	113	-7.19E+07	50	15.4188	113	-7.19E+07
EXPLIN2	25	13.1176	52	-7.20E+07	25	13.0891	52	-7.20E+07
EXPQUAD	132	256.690	281	-3.68E+009	70	71.537	139	-3.68E+009
GRIDGENA	5	20.8589	200	2.35E+04	5	21.1741	200	2.35E+04
HADAMALS	10	0.7603	6	7.31E+03	10	0.8069	6	7.31E+03
HART6	309	1816.3125	774	2.51E+00	309	1817.5970	774	2.51E+00
HATFLDA	24	0.0296	58	8.07E-19	24	0.0440	58	8.07E-19
HATFLDB	20	0.0366	36	5.57E-03	20	0.0449	36	5.57E-03
HATFLDC	6	0.0135	38	1.70E-17	6	0.0223	38	1.70E-17
HIMMELP1	6	0.0090	5	-2.39E+01	6	0.0207	5	-2.39E+01
HS1	27	0.033	36	5.49E-028	16	0.016	17	4.02E-015
HS110	8	0.0076	2	-4.58E+01	9	0.0210	2	-4.58E+01
HS2	8	0.0053	3	4.94E+00	8	0.0187	3	4.94E+00
HS25	3	0.0054	2	3.28E+01	3	0.0181	2	3.28E+01
HS3	2	0.0014	3	3.16E-35	2	0.0136	3	3.16E-35
HS38	57	3.5620	179	2.86E-25	54	0.0910	173	6.12E-15
HS3MOD	3	0.0058	4	0.00E+00	3	0.0147	4	0.00E+00
HS4	2	0.0011	3	2.67E+00	2	0.0136	3	2.67E+00
HS45	2	0.0042	4	1.00E+00	2	0.0165	4	1.00E+00
HS5	13	0.0138	9	-1.91E+00	13	0.0297	9	-1.91E+00
JNLBRNG1	110	120.0741	238	-1.81E-01	110	117.8933	238	-1.81E-01
JNLBRNG2	8	5.7573	255	-4.15E+00	8	5.8375	255	-4.15E+00
JNLBRNGA	8	6.0473	303	-2.91E-01	8	6.0616	303	-2.91E-01
JNLBRNGB	9	7.7468	553	-6.46E+00	9	7.8323	553	-6.46E+00
LINVERSE	46	82.5600	51	6.82E+02	46	78.8241	51	6.82E+02
LOGROS	2	0.0018	3	1.93E+02	2	0.0142	3	1.93E+02
MAXLIKA	276	1.1937	337	1.14E+03	277	1.2462	346	1.14E+03
MCCORMCK	15	28.7382	35	-4.57E+03	8	10.6754	17	-4.57E+03
MDHOLE	53	0.0860	77	0.00E+00	42	0.0760	50	0.00E+00
NCVXBQP1	2	4.0910	5	-1.99E+08	2	3.9777	5	-1.99E+08
NCVXBQP2	32	5.7840	121	-1.33E+08	9	4.6880	16	-1.33E+08
NCVXBQP3	33	8.5250	38	-6.58E+07	11	7.3520	16	-6.58E+07
NOBNDTOR	23	53.9472	687	-4.50E-01	23	84.5287	687	-4.50E-01
NONSCOMP	16	23.6340	43	3.76E-10	16	17.1380	43	3.76E-10
OBSTCLAE	122	177.1975	158	1.06E+01	122	248.4091	158	1.06E+01
OBSTCLAL	11	5.5947	74	1.06E+01	11	7.8374	74	1.06E+01
OBSTCLBL	309	1816.3125	774	2.51E+00	309	1817.5970	774	2.51E+00
OBSTCLBM	30	73.3985	67	3.38E+01	30	97.7889	67	3.38E+01

Table 2: Numerical comparisons on a subset of test problems.

Name	GPFTRA iter	CPU	CG	f^*	FTRA iter	CPU	CG	f^*
OBSTCLBU	30	48.5622	74	3.38E+01	30	65.4013	74	3.38E+01
OSLBQP	2	0.1960	5	6.25E+00	2	0.0139	5	6.25E+00
PALMER1	31	0.1584	18	1.18E+04	26	0.1093	18	1.18E+04
PALMER1A	112	0.1530	466	8.99E-02	60	0.0920	145	8.99E-02
PALMER1B	35	0.1707	59	3.45E+00	42	0.1811	61	3.45E+00
PALMER1E	156	0.3070	1088	8.35E-04	192	0.4090	1285	8.35E-04
PALMER2	48	0.0750	42	3.65E+05	51	0.0780	40	3.65E+03
PALMER2A	219	0.3350	1029	1.71E-02	113	0.1830	487	1.71E-02
PALMER2B	23	0.0363	51	6.23E-01	27	0.1281	43	6.23E-01
PALMER2E	238	1.5916	1970	2.07E-04	99	0.1912	463	1.16E-01
PALMER3	21	0.0829	26	2.42E+03	27	0.0532	23	2.42E+03
PALMER3A	249	1.1845	1221	2.04E-02	80	0.3043	229	2.04E-02
PALMER3B	27	0.0345	51	6.23E-01	33	0.1471	56	4.23E+00
PALMER3E	181	0.3630	1499	5.07E-05	190	0.4020	1569	5.07E-05
PALMER4	22	0.0909	26	2.42E+03	26	0.0484	20	2.42E+03
PALMER4A	62	0.1610	173	4.06E-02	145	0.6712	668	4.06E-02
PALMER4B	32	0.0390	55	6.84E+00	21	0.0370	37	6.84+000
PALMER4E	113	0.2450	950	1.48E-04	127	0.2440	993	1.48E-04
PALMER5A	4264	7.5750	37774	5.18E-02	4493	7.9180	38853	5.18E-02
PALMER5B	2279	4.2490	20071	9.75E-03	2639	4.7700	23430	9.75E-03
PALMER5D	3	0.0101	7	8.73E+01	3	0.0175	7	8.73E+01
PALMER5E	2153	3.5620	15485	2.56E-02	2272	3.7380	16638	2.56E-02
PALMER6A	282	0.4149	1202	5.59E-02	95	0.1355	255	5.59E-02
PALMER6E	43	0.0750	295	2.24E-04	47	0.1390	349	2.24E-04
PALMER7E	601	1.0380	3966	6.68E+00	706	1.2600	5022	6.68E+00
PALMER8A	39	0.0520	107	7.40E-02	43	0.0530	102	7.40E-02
PALMER8E	32	0.0530	209	6.34E-03	61	0.1230	438	6.34E-03
PENTDI	2	1.1751	6	-7.50E-01	2	1.1899	6	-7.50E-01
PROBPENL	2	0.1145	3	3.99E-07	2	0.1286	3	3.99E-07
PSPDOC	26	0.0470	57	2.41E+00	230	0.3330	469	2.41E+02
QR3DLS	373	144.6730	99919	2.43E-07	521	133.6800	57126	8.47E-03
QUDLIN	2	758.9444	3	-7.19E+07	2	58.0068	3	-7.19E+07
S368	6	0.0274	4	-7.50E-01	6	0.0426	4	-7.50E-01
SIM2BQP	2	0.0040	3	0.00E+00	2	0.0164	3	0.00E+00
SIMBQP	2	0.0057	2	0.00E+00	2	0.0167	2	0.00E+00
SINEALI	8	0.7021	34	-9.99E+04	8	0.7532	34	-9.99E+04
SPECAN	12	0.6836	50	1.65E-13	12	0.7734	50	1.65E-13
TORSION1	24	63.3769	553	-4.30E-01	24	61.7812	553	-4.30E-01
TORSION2	93	315.9712	184	-4.30E-01	93	317.2696	184	-4.30E-01
TORSION3	12	23.4316	150	-1.22E+00	12	23.3387	150	-1.22E+00
TORSION4	12	34.5391	69	-1.22E+00	12	35.1008	69	-1.22E+00
TORSION5	7	13.2039	50	-2.86E+00	7	12.1168	50	-2.86E+00
TORSION6	6	17.7341	40	-2.86E+00	6	17.9586	40	-2.86E+00
TORSIONA	24	62.6318	553	-4.18E-01	24	62.7643	553	-4.18E-01
TORSIONB	60	218.1529	128	-4.18E-01	60	225.9029	128	-4.18E-01
TORSIONC	12	23.3875	150	-1.20E+00	12	23.3422	150	-1.20E+00
TORSIOND	11	35.9114	101	-1.20E+00	11	36.1875	101	-1.20E+00
TORSIONE	7	12.2407	50	-2.85E+00	7	12.2321	50	-2.85E+00
TORSIONF	6	17.2641	40	-2.85E+00	6	17.2753	40	-2.85E+00
WEEDS	37	0.0470	59	2.59E+00	57	0.0670	86	2.59E+00
YFIT	104	0.1480	271	6.74E-13	33	0.1030	70	3.16E-009