PRECONDITIONED SPECTRAL PROJECTED GRADIENT METHOD ON CONVEX SETS *1)

Lenys Bello
(Dpto. de Matemática, Facultad de Ciencias (FACYT), Universidad de Carabobo, Valencia, Venezuela)

Marcos Raydan
(Dpto. de Computación, Universidad Central de Venezuela, Ap. 47002, Caracas 1041-A, Venezuela)

Abstract

The spectral gradient method has proved to be effective for solving large-scale unconstrained optimization problems. It has been recently extended and combined with the projected gradient method for solving optimization problems on convex sets. This combination includes the use of nonmonotone line search techniques to preserve the fast local convergence. In this work we further extend the spectral choice of steplength to accept preconditioned directions when a good preconditioner is available. We present an algorithm that combines the spectral projected gradient method with preconditioning strategies to increase the local speed of convergence while keeping the global properties. We discuss implementation details for solving large-scale problems.

Mathematics subject classification: 49M, 90C, 65K.

Key words: Spectral gradient method, Projected gradient method, Preconditioning techniques, Nonmonotone line search.

1. Introduction

We consider the optimization problem

$$\min \{ f(x) : x \in \Omega \},$$

where $\Omega$ is a nonempty closed and convex set in $\mathbb{R}^n$, $n$ is large, $f$ is continuously differentiable, $g(x) = \nabla f(x)$ is available, and $G(x) \approx \nabla^2 f(x)$ is also available and will be considered as a preconditioner. Our main objective is to develop a preconditioned and projected extension of the spectral gradient method to solve this problem.

Spectral gradient methods are nonmonotone schemes that have recently received considerable attention in the numerical analysis and optimization literature. They were introduced by Barzilai and Borwein [1], the convergence for quadratics was established by Raydan [17], and more recently, a proof of the R-linear rate of convergence for convex quadratics was given by Dai and Liao [10]. A complete review is presented by Fletcher [11], and the asymptotic behavior is studied by Dai and Fletcher [9].

The spectral gradient methods are nonmonotone schemes that have recently received considerable attention in the numerical analysis and optimization literature. They were introduced by Barzilai and Borwein [1], the convergence for quadratics was established by Raydan [17], and more recently, a proof of the R-linear rate of convergence for convex quadratics was given by Dai and Liao [10]. A complete review is presented by Fletcher [11], and the asymptotic behavior is studied by Dai and Fletcher [9].

The spectral gradient methods have been applied successfully to find local minimizers of large scale problems ([4, 5, 7, 8, 12, 18, 20]), to solve box-constrained quadratic optimization (Bielschowsky et al. [2]), and also to minimize general smooth functions on convex sets [6]. Preconditioned spectral gradient versions have also been developed ([16, 13, 15]). However, the combination of preconditioning techniques to accelerate the process and projected techniques on convex sets, for robustness and regularity, has not been studied.

---

* Received December 10, 2003; final revised July 17, 2004.
1) This author was partially supported by UCV-PROJECT 97-003769.
The paper is divided into sections as follows. In Section 2 we describe briefly the preconditioned spectral gradient method, and we recall some of the important aspects of the spectral projected gradient method on convex sets. In Section 3 we present the new algorithm that combines both ideas taking into account the difficult cases. In Section 4 we present preliminary and encouraging numerical results, and some final remarks.

2. Previous Extensions

We describe briefly the most important properties of the preconditioned spectral gradient method and the spectral projected gradient method on convex sets.

2.1 Preconditioned spectral gradient method

The iterates of the Preconditioned Spectral Gradient (PSG) method presented by Glunt, Hayden, and Raydan [13] are defined by

\[ x_{k+1} = x_k - \alpha_k z_k, \]

where \( z_k = G_k^{-1} g_k \), \( G_k \) is a nonsingular approximation to the Hessian of \( f \) at \( x_k \) and the scalar \( \alpha_k \) is given by

\[ \alpha_k = \left( \frac{\|z_k\|^2}{\|x_k\|^2} - 1 \right), \]

where \( x_0 \) and \( \alpha_0 \) are given initial data (see also [16]).

The PSG method requires no line search during the process but does not guarantee monotonic descent in the objective function. As a consequence, Raydan [18] proposed a globalization scheme for the spectral gradient algorithm that fits nicely with the nonmonotone behavior of this family of methods. Roughly speaking, the algorithm forces the step to satisfy this weak condition:

\[ f(x_{k+1}) \leq \max_{0 \leq j \leq M} f(x_{k-j}) + \gamma g_k^T (x_{k+1} - x_k), \]

where \( M \) is a nonnegative integer and \( \gamma \) is a small positive number. When \( M > 0 \) this condition allows the objective function to increase at some iterations and still guarantees global convergence. This globalization strategy is based on the nonmonotone line search technique of Grippo, Lampariello and Lucidi [14].

A direct combination of the PSG method and the nonmonotone globalization strategy described above produces an algorithm fully described in Luengo et. al. [15].

2.2 Spectral projected gradient method

There have been many different variations of the projected gradient method that can be viewed as the constrained extensions of the optimal gradient method for unconstrained minimization. They all have the common property of maintaining feasibility of the iterates by frequently projecting trial steps on the feasible convex set. In particular, Birgin et al. [6, 3] combine the projected gradient method with recently developed ingredients in optimization, as follows. The algorithm starts with \( x_0 \in \mathbb{R}^n \) and is based on the spectral projected gradient direction \( d_k = P(x_k - \alpha_k g(x_k)) - x_k \), where \( \alpha_k \) is the spectral choice of steplength \( \frac{\|z_k\|^2}{\|x_k\|^2} - 1 \), and for \( z \in \mathbb{R}^n \), \( P(z) \) is the projection on \( \Omega \). In the case of rejection of the first trial point, \( x_k + d_k \), the next ones are computed along the same direction, i.e., \( x_{k+1} = x_k + \lambda d_k \), using a nonmonotone line search to force the following condition

\[ f(x_{k+1}) \leq \max_{0 \leq j \leq M} f(x_{k-j}) + \gamma \lambda (d_k, g(x_k)), \]

where \( M \geq 1 \) is a given integer. As a consequence, the projection operation must be performed only once per iteration. More details can be found in [6] and [3].
3. New Algorithm

We combine both versions, preconditioned and projected, to obtain a new algorithm. However, the combination cannot be obtained by simply connecting both ideas. The most important difficulty is that the preconditioned direction, even if it is a descent direction, might not be a descent direction after being projected on \( \Omega \). Therefore, additional controls need to be included to produce a robust and well-defined algorithm. In the rest of this document the notation \( \|z\| \) represents the Euclidean norm of the vector \( z \).

**PSPG:** Preconditioned Spectral Projected Gradient Algorithm

Given \( x_0 \in \Omega \), \( M > 0 \), \( \alpha_0 \in [\epsilon, 1/\epsilon] \), \( \gamma \in (0, 1) \), \( 0 < \sigma_1 < \sigma_2 < 1 \), a small tolerance \( tol > 0 \) for the stopping criterion and a tolerance \( tolpre > 0 \) for activating the preconditioner; \( 0 < \epsilon < 1 \) and \( 0 < c < 1 \).

Set \( k \leftarrow 0 \), \( \hat{d}_0 = P(x_0 - \alpha_0 g(x_0)) - x_0 \) and \( \text{precond} = \text{off} \)

while \( (\|\hat{d}_k\| > tol) \)

\[ \begin{align*}
  \text{if} \ (\|\hat{d}_k\| \leq tolpre) \ \text{then} \ (\text{precond} = \text{on}) \\
  \text{end if}
\end{align*} \]

\[ \begin{align*}
  \text{if} \ (\text{precond} = \text{on}) \ \text{choose} \ G_k \text{ and} \\
  \text{solve} \ G_k d_k = g_k \text{ for} \ d_k, \\
  \text{else} \ d_k = g_k \\
  \text{end if}
\end{align*} \]

set \( d_k = P(x_k - \alpha_k d_k) - x_k \),

\[ \begin{align*}
  \text{if} \ (\text{precond} = \text{on}) \ \text{and} \ (d_k^T g_k > -\epsilon \max(\|d_k\|, \|\hat{d}_k\|, \|d_k\|^2, \|g_k\|^2)) \ , \text{then} \\
  \text{precond} = \text{off}, \ tolpre = tolpre \cdot c \ \text{and} \ d_k = \hat{d}_k \\
  \text{end if}
\end{align*} \]

set \( \lambda \leftarrow 1 \) and set \( x_+ = x_k + d_k \)

while \( (f(x_+) > \max_{0 \leq j \leq \min\{k,m-1\}} \{f(x_{k-j})\} + \gamma \lambda(d_k, g(x_k))) \)

\[ \begin{align*}
  \text{choose} \ \sigma \in [\sigma_1, \sigma_2], \ \text{set} \ \lambda = \sigma \lambda \ \text{and} \ \text{set} \ x_+ = x_k + \lambda d_k \\
  \text{endwhile}
\end{align*} \]

set \( x_{k+1} \leftarrow x_+ \), set \( s_k = x_{k+1} - x_k \), set \( \lambda_k = \lambda \),

set \( y_k = g(x_{k+1}) - g(x_k) \) and set \( b_k = (d_k, y_k) \).

\[ \begin{align*}
  \text{If} \ (b_k \leq \epsilon) \ \text{then set} \ \alpha_{k+1} = \frac{1}{\epsilon} \\
  \text{else, set} \ \alpha_k = s_k^T g_k \ \text{and set} \ \alpha_{k+1} = \min\{1, \max\{\frac{1}{\epsilon}, \epsilon, \frac{\alpha_k}{\epsilon}\}\} \\
  \text{end if}
\end{align*} \]

set \( k \leftarrow k + 1 \) and \( \hat{d}_k = P(x_k - \alpha_k g_k) - x_k \)

\[ \begin{align*}
  \text{endwhile}
\end{align*} \]

set \( x \leftarrow x_k \).

**Remarks.**

1) Notice that if \( \Omega = \mathbb{R}^n \) then the PSPG algorithm reduces to the PSG algorithm in [6]; and if \( G_k = I \) for all \( k \) then it reduces to the SPG method described in [15].

2) The preconditioner should be activated when the iterations are “close” to a solution ([15]). In general, depending on the specific application, this information is supplied by the user. In the PSPG algorithm we use a default criterion, i.e., \( \|\hat{d}_k\| \leq tolpre \), where \( tolpre \) is given by the user. Notice that we do not impose any conditions on the matrix \( G_k \).

3) In order to save computations involved by invoking the preconditioner too early and often,
Theorem 3.1. Assume that \( \Omega_0 = \{ x \in \Omega : f(x) \leq f(x_0) \} \) is a bounded set. Then Algorithm PSPG is well defined, and either \( d_j = 0 \) for some finite \( j \) or any accumulation point of the sequence \( \{x_k\} \) that it generates is a constrained stationary point.

Proof. In Algorithm PSPG the search direction is either \( d_k = P(x_k - \alpha_k G_k^{-1} g_k) - x_k \) or \( \hat{d}_k = P(x_k - \alpha_k g_k) - x_k \). When the preconditioner is activated we try \( d_k \), but if it does not satisfy the sufficient decrease condition (1), then we use \( \hat{d}_k \). If the preconditioner is not activated we also use \( d_k \). Since we force \( \alpha_k \in [\epsilon, 1/\epsilon] \) for all \( k \), then by Lemma 2.1 in [6], when using \( \hat{d}_k \) the sufficient decrease condition (1) is satisfied. Therefore, for all \( k \) the search direction satisfies (1), and the nonmonotone line search procedure terminates in a finite number of trials, which implies that the algorithm is well defined.

Now, we make use of the first part of the proof of the convergence theorem in [14, p. 709]. Let us define \( m(k) = \min(k, M) \). Clearly, \( m(0) = 0 \) and

\[
0 \leq m(k) \leq \min(m(k-1) + 1, M) \quad \text{for } k \geq 1.
\]

Using the fact that all search directions \( d_k \) satisfies (1), then we can obtain positive numbers \( c_1 \) and \( c_2 \) such that they satisfy \( g_k^T d_k \leq -c_1 \| g_k \|^2 \), and \( \| d_k \| \leq c_2 \| g_k \| \) for all \( k \). In fact, from (1) we have \( d_k^T g_k \leq -\epsilon \| g_k \|^2 \) for all \( k \), and \( c_1 = \epsilon \). Once again using (1) we obtain \( d_k^T g_k \leq -\epsilon \| d_k \|^2 \), and by the Cauchy-Schwarz inequality it follows that

\[
\epsilon \| d_k \|^2 \leq |d_k^T g_k| \leq \| d_k \| \| g_k \|,
\]

which implies that

\[
\| d_k \| \leq \frac{1}{\epsilon} \| g_k \|,
\]

and hence \( c_2 = 1/\epsilon \). Finally, in our algorithm the trial steps are all constant \( (a = 1) \). Therefore, repeating the same arguments in [14, p. 710-711] we obtain:

\[
\lim_{k \to \infty} \lambda_k (g_k^T d_k) = 0. \tag{2}
\]

Let \( \bar{x} \in \Omega \) be an accumulation point of \( \{x_k\} \), relabel \( \{x_k\} \) a subsequence converging to \( \bar{x} \), and \( \{\lambda_k\} \), \( \{d_k\} \), and \( \{\hat{d}_k\} \) the corresponding subsequences generated in the algorithm. We consider two cases:

**Case 1.** Assume that \( \inf \lambda_k = 0 \). Suppose, by contradiction, that \( \bar{x} \) is not a constrained stationary point. Hence, by continuity and compactness, there exists \( \delta > 0 \) such that

\[
\| P(\bar{x} - \alpha g(\bar{x})) - \bar{x} \| > \frac{\delta}{\epsilon} > 0 \quad \text{for all } \alpha \in [\epsilon, 1/\epsilon].
\]

Since the condition (1) is satisfied by all iterates of algorithm PSPG, then

\[
\left\langle g(x_k), \frac{\hat{d}_k}{\| d_k \|} \right\rangle < -\epsilon \| d_k \| < -\frac{\delta}{2} \quad \text{for all } \alpha \in [\epsilon, 1/\epsilon], \tag{3}
\]

and \( k \) large enough on the subsequence that converges to \( \bar{x} \).
Since \( \inf \lambda_k = 0 \), there exists a subsequence \( \{x_k\}_K \) such that
\[
\lim_{k \to K} \lambda_k = 0.
\]
In that case, from the way \( \lambda_k \) is chosen in the nonmonotone line search, there exists an index \( k \) sufficiently large such that for all \( k \geq \hat{k} \), \( k \in K \), there exists \( \rho_k \), \( 0 < \sigma_1 \leq \rho_k \leq \sigma_2 \), for which \( \lambda_k / \rho_k > 0 \) fails to satisfy the nonmonotone line search condition, i.e.,
\[
f(x_k + \frac{\lambda_k}{\rho_k} d_k) > \max_{0 \leq j \leq M-1} f(x_{k-j}) + \gamma \frac{\lambda_k}{\rho_k} (g(x_k), d_k) \geq f(x_k) + \gamma \left( \frac{\lambda_k}{\rho_k} (g(x_k), d_k) \right) .
\]
Hence,
\[
\frac{f(x_k + \frac{\lambda_k}{\rho_k} d_k) - f(x_k)}{\lambda_k / \rho_k} \geq \gamma (g(x_k), d_k).
\]
By the mean value theorem, this relation can be written as
\[
\langle g(x_k + t_k d_k), d_k \rangle > \gamma 
\]
where \( t_k \) is a scalar in the interval \([0, \lambda_k / \rho_k]\) that goes to zero as \( k \to K \) goes to infinity.

Taking a convenient subsequence such that \( d_k / \|d_k\| \) is convergent to \( d \), and taking limits in (4) we deduce that \( (1 - \gamma) \langle g(\bar{x}), d \rangle \geq 0 \). (In fact, observe that \( \{\|d_k\|\}_K \) is bounded and so \( t_k \|d_k\| \to 0 \).) Since \( (1 - \gamma) > 0 \) and \( \langle g(x_k), d_k \rangle < 0 \) for all \( k \), then \( \langle g(\bar{x}), d \rangle = 0 \).

By continuity and the definition of \( d_k \) this implies that for \( k \) large enough on that subsequence we have that
\[
\langle g(x_k), \frac{d_k}{\|d_k\|} \rangle > -\delta / 2,
\]
which contradicts (3).

**Case 2.** Assume that \( \inf \lambda_k \geq \rho > 0 \). Hence, by (2) we obtain
\[
\lim_{k \to \infty} g^*_k d_k = 0,
\]
which implies, using (1), that \( \lim_{k \to \infty} g^*_k = 0 \). Therefore, \( g(\bar{x}) = 0 \), and
\[
\|P(\bar{x} - \alpha g(\bar{x})) - \bar{x}\| = 0 \text{ for all } \alpha \in [e, \frac{1}{e}],
\]
which implies that \( \bar{x} \) is a constrained stationary point.

## 4. Numerical Experiments

We compare the Spectral Projected Gradient Method (SPG) and the Preconditioned Spectral Projected Gradient Method (PSPG) on 8 standard test problems that can be found in the literature. A description of the functions and the starting points can be found in [18] and references therein. All the experiments were run on a PC Pentium III, 800 MHz and 256 Mbytes RAM, Fortran 77 (double precision), we used \( \gamma = 10^{-4} \), \( M = 10 \), \( \sigma_1 = 0.1 \), \( \sigma_2 = 0.6 \), \( \epsilon = 10^{-20} \), \( \alpha_0 = 1 / \|g_0\| \) and we stop when \( \|d_k\| \leq 10^{-6} \). For the PSPG we used the tridiagonal part of the Hessian as a preconditioner. In this case, solving the preconditioned linear system requires \( O(n) \) flops and the storage for the matrix \( G_k \) is only 2 \( n \)-dimensional vectors. The preconditioner is activated (“local test”) whenever \( \|P_0(x_k - \alpha g_k) - x_k\| \leq \text{tolpre} \), where \( \text{tolpre} \) is a tolerance factor. If the preconditioning strategy is deactivated (\precond = off\), then we set \( \text{tolpre} = \text{tolpre} \times 10^{-1} \). In our particular experiments, \( \Omega \) is a box, i.e.,
\[
\Omega = \{ x : l \leq x \leq u \}
\]
where the vectors \( l \) and \( u \) are given and they can have infinite values in some of the positions. Table 1 lists the 8 functions and the structure of their Hessians. However, due to space restrictions, we only report three cases from this table. We report the best case, the worst case, and an average case that we observed in our experiments. In all the experiments we verified that the two methods converged to the same point. We report in tables 2, 3, and 4 the numerical results for the three functions. In particular, we report the dimension of the problem \( (n) \), the tolerance factor for the “local test” \( (tolpre) \), the number of times that the preconditioner was activated \( (\text{Precond}(t)) \) where \( t \) is the iteration at which it was activated for the last time, the number of iterations required for convergence \( (iter) \), the number of function evaluations \( (F) \) the number of gradient evaluations \( (G) \), the vectors \( (l) \) and \( (u) \), the required CPU time in seconds \( (\text{Time}) \), and whether the unconstrained solution \( x^* \) is in \( \Omega \) or not.

We observe that the global PSPG method is a robust method to find local minimizers of nonquadratic functions subject to bound constraints. It outperforms the SPG method in number of iterations and computational work.

For the Strictly Convex 2 function the preconditioning matrix is the exact Hessian (diagonal) and so the global PSPG method shows superlinear convergence in the last few iterations. This fact explains the excellent behavior of the PSPG for that particular test problem. For the Penalty 1 function, that is not convex, is highly nonlinear, and the exact Hessian is dense, the tridiagonal part of the Hessian is not a very accurate preconditioner. As a consequence, we observe the worst behavior of the PSPG method when compared with the SPG method on the 8 standard test problems. Even in that case, the PSPG method is competitive in number of iterations and computational work. Finally, for the Extended Powell Singular function we observed what we have called the average behavior. In that case, the PSPG method converges in fewer number of iterations and less computational work than the SPG method. Indeed, in the average, when the solution is in \( \Omega \) the PSPG method is approximately six time faster than the SPG method, and when the solution is not in \( \Omega \), it is approximately twice as fast as the SPG method.

In general, we observe that it is important to activate the preconditioner at the right iteration by using correctly the parameter \( tolpre \), which plays an important role during the convergence process. Of course, in order to establish definite conclusions it is necessary to run experiments on real and larger problems, using more realistic preconditioners. These are all interesting topics that deserve further investigation in the near future.

In summary, our preliminary numerical results indicate that the PSPG method combines in a suitable way the preconditioning technique and the projected scheme to produce a promising idea that accelerates the convergence while adding robustness and regularity to the process, in the sense of [19].

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
<th>Hessian</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Brown Almost Linear</td>
<td>dense</td>
</tr>
<tr>
<td>2</td>
<td>Broyden Tridiagonal</td>
<td>tridiagonal</td>
</tr>
<tr>
<td>3</td>
<td>Oren’s Power</td>
<td>dense</td>
</tr>
<tr>
<td>4</td>
<td>Penalty 1</td>
<td>dense</td>
</tr>
<tr>
<td>5</td>
<td>Extended Powell Singular</td>
<td>pentadiagonal</td>
</tr>
<tr>
<td>6</td>
<td>Extended Rosenbrock</td>
<td>tridiagonal</td>
</tr>
<tr>
<td>7</td>
<td>Variably Dimensioned</td>
<td>dense</td>
</tr>
<tr>
<td>8</td>
<td>Strictly Convex 2</td>
<td>diagonal</td>
</tr>
</tbody>
</table>
Table 2: PSPG with and without preconditioning for Strictly Convex ($x^* = 0$) for different values of $n$ and different values of $l$ and $u$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>topre</th>
<th>Precond($t$)</th>
<th>$\text{iter}$</th>
<th>$F$</th>
<th>$G$</th>
<th>$l$</th>
<th>$u$</th>
<th>Time</th>
<th>$x^* \in \Omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1.0d-20</td>
<td>off</td>
<td>83</td>
<td>99</td>
<td>84</td>
<td>-10</td>
<td>10</td>
<td>0.23</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+10</td>
<td>1(1)</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>-10</td>
<td>10</td>
<td>0.01</td>
<td>yes</td>
</tr>
<tr>
<td>500</td>
<td>1.0d-20</td>
<td>off</td>
<td>214</td>
<td>286</td>
<td>215</td>
<td>$-\infty$</td>
<td>0.5</td>
<td>0.93</td>
<td>yes</td>
</tr>
<tr>
<td>500</td>
<td>1.0d+10</td>
<td>1(1)</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>$-\infty$</td>
<td>0.5</td>
<td>0.01</td>
<td>yes</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d-20</td>
<td>off</td>
<td>366</td>
<td>549</td>
<td>367</td>
<td>$-\infty$</td>
<td>0.5</td>
<td>2.12</td>
<td>yes</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+10</td>
<td>1(1)</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>$-\infty$</td>
<td>0.5</td>
<td>0.02</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d-20</td>
<td>off</td>
<td>78</td>
<td>82</td>
<td>79</td>
<td>-40</td>
<td>10($u_1 = -3,u_n = 6$)</td>
<td>0.17</td>
<td>no</td>
</tr>
<tr>
<td>100</td>
<td>1.10+10</td>
<td>1(1)</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>-40</td>
<td>10($u_1 = -3,u_n = 6$)</td>
<td>0.02</td>
<td>no</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d-20</td>
<td>off</td>
<td>347</td>
<td>475</td>
<td>348</td>
<td>-40</td>
<td>10($u_1 = -3,u_n = 6$)</td>
<td>1.92</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d-20</td>
<td>off</td>
<td>1466</td>
<td>2253</td>
<td>1467</td>
<td>-40</td>
<td>10($u_1 = -3,u_n = 6$)</td>
<td>57.07</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+10</td>
<td>1(1)</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>-40</td>
<td>10($u_1 = -3,u_n = 6$)</td>
<td>0.21</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 3: PSPG with and without preconditioning for Penalty 1 ($x^* \approx 0.02$) for different values of $n$ and different values of $l$ and $u$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>topre</th>
<th>Precond($t$)</th>
<th>$\text{iter}$</th>
<th>$F$</th>
<th>$G$</th>
<th>$l$</th>
<th>$u$</th>
<th>Time</th>
<th>$x^* \in \Omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1.0d+20</td>
<td>off</td>
<td>154</td>
<td>398</td>
<td>155</td>
<td>-10</td>
<td>10</td>
<td>0.34</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+2</td>
<td>4(65)</td>
<td>78</td>
<td>133</td>
<td>79</td>
<td>-10</td>
<td>10</td>
<td>0.19</td>
<td>yes</td>
</tr>
<tr>
<td>500</td>
<td>1.0d+20</td>
<td>off</td>
<td>50</td>
<td>69</td>
<td>51</td>
<td>-10</td>
<td>10</td>
<td>0.20</td>
<td>yes</td>
</tr>
<tr>
<td>500</td>
<td>1.0d+3</td>
<td>3(51)</td>
<td>56</td>
<td>86</td>
<td>57</td>
<td>-10</td>
<td>10</td>
<td>0.25</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+20</td>
<td>off</td>
<td>75</td>
<td>373</td>
<td>76</td>
<td>-10($l_1 = 5$)</td>
<td>10</td>
<td>0.20</td>
<td>no</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+2</td>
<td>1(26)</td>
<td>39</td>
<td>41</td>
<td>40</td>
<td>-10($l_1 = 5$)</td>
<td>10</td>
<td>0.13</td>
<td>no</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+20</td>
<td>off</td>
<td>69</td>
<td>71</td>
<td>70</td>
<td>-100($l_1 = 5$)</td>
<td>100($u_n = 10$)</td>
<td>0.08</td>
<td>no</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+3</td>
<td>2(56)</td>
<td>58</td>
<td>114</td>
<td>59</td>
<td>-100($l_1 = 5$)</td>
<td>100($u_n = 10$)</td>
<td>0.11</td>
<td>no</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+20</td>
<td>off</td>
<td>43</td>
<td>93</td>
<td>44</td>
<td>-10($l_1 = 5$)</td>
<td>10</td>
<td>0.23</td>
<td>no</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+4</td>
<td>1(36)</td>
<td>40</td>
<td>42</td>
<td>41</td>
<td>-10($l_1 = 5$)</td>
<td>10</td>
<td>0.18</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+20</td>
<td>off</td>
<td>83</td>
<td>85</td>
<td>84</td>
<td>-100($l_1 = 5$)</td>
<td>100($u_n = 10$)</td>
<td>1.7</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+5</td>
<td>1(55)</td>
<td>77</td>
<td>127</td>
<td>78</td>
<td>-100($l_1 = 5$)</td>
<td>100($u_n = 10$)</td>
<td>2.37</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 4: PSPG with and without preconditioning for Extended Powell Singular ($-0.5 < x_i^* < 0.5$ for all $i$) for different values of $n$ and different values of $l$ and $u$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>topre</th>
<th>Precond($t$)</th>
<th>$\text{iter}$</th>
<th>$F$</th>
<th>$G$</th>
<th>$l$</th>
<th>$u$</th>
<th>Time</th>
<th>$x^* \in \Omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1.0d+20</td>
<td>off</td>
<td>336</td>
<td>566</td>
<td>337</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>0.14</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+1</td>
<td>1(31)</td>
<td>46</td>
<td>49</td>
<td>47</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>0.02</td>
<td>yes</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+20</td>
<td>off</td>
<td>322</td>
<td>581</td>
<td>323</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>0.95</td>
<td>yes</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+2</td>
<td>1(31)</td>
<td>46</td>
<td>49</td>
<td>47</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>0.12</td>
<td>yes</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+20</td>
<td>off</td>
<td>206</td>
<td>356</td>
<td>207</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>6.83</td>
<td>yes</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+1</td>
<td>1(31)</td>
<td>46</td>
<td>49</td>
<td>47</td>
<td>-1($l_1 = -10$)</td>
<td>1000($u_1 = 30$)</td>
<td>1.46</td>
<td>yes</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+20</td>
<td>off</td>
<td>274</td>
<td>337</td>
<td>275</td>
<td>$-\infty$</td>
<td>0</td>
<td>0.09</td>
<td>no</td>
</tr>
<tr>
<td>100</td>
<td>1.0d+3</td>
<td>1(143)</td>
<td>157</td>
<td>222</td>
<td>158</td>
<td>$-\infty$</td>
<td>0</td>
<td>0.07</td>
<td>no</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+20</td>
<td>off</td>
<td>269</td>
<td>335</td>
<td>270</td>
<td>$-\infty$</td>
<td>0</td>
<td>0.66</td>
<td>no</td>
</tr>
<tr>
<td>1000</td>
<td>1.0d+3</td>
<td>1(142)</td>
<td>157</td>
<td>223</td>
<td>158</td>
<td>$-\infty$</td>
<td>0</td>
<td>0.41</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+20</td>
<td>off</td>
<td>269</td>
<td>335</td>
<td>270</td>
<td>$-\infty$</td>
<td>0</td>
<td>7.33</td>
<td>no</td>
</tr>
<tr>
<td>10000</td>
<td>1.0d+3</td>
<td>1(142)</td>
<td>157</td>
<td>222</td>
<td>158</td>
<td>$-\infty$</td>
<td>0</td>
<td>4.52</td>
<td>no</td>
</tr>
</tbody>
</table>

References


