Master's Thesis

A regularized limited memory BFGS method for unconstrained minimization problems

Guidance

Associate Professor Nobuo YAMASHITA

Shinji SUGIMOTO

Department of Applied Mathematics and Physics

Graduate School of Informatics

Kyoto University

February 2014

Abstract

The limited memory BFGS (L-BFGS) method is one of the popular methods for solving large scale unconstrained minimization problems. The L-BFGS method works with small memory, and it is more efficient than the steepest descent method. However, since L-BFGS method needs line search with the Wolfe condition, it may require many function evaluations for ill-posed problems. To overcome the difficulty, the L-BFGS method combined with the trust region method has been proposed, and it has nice performances in terms of the number of function evaluations. However, the trust region method solves a nonconvex subproblem at each iteration, and hence it takes much time for the large scale problems.

In this paper, we propose a method which combines the L-BFGS method with the regularized Newton method instead of the trust region method. The computational cost for the single iteration of the proposed method is same as that of the original L-BFGS method. We show that the proposed method has global convergence under usual conditions. Moreover, we presents numerical results that show the rubustness of the proposed method.

Contents

1 Introduction

In this paper, we consider the following unconstrained minimization problem.

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathbb{R}^n, \end{array} \tag{1.1}$$

where f is a smooth function from $\mathbb{R}^n$ into $\mathbb{R}$.
We focus on the large scale case. Standard solution methods for (1.1) such as the steepest decent method, the Newton method and BFGS method [5, 12] are not suitable for large scale problems from the following reasons. The steepest decent method converges slowly in general. The Newton method needs to compute the Hessian matrix and solve a linear equation at each iteration. Furthermore, the BFGS method requires $O(n^2)$ memory to store the approximate Hessian of f.

One of the popular methods for large scale problems (1.1) is the limited memory BFGS method (L-BFGS) proposed by Nocedal [9, 11]. The L-BFGS method uses the m vector pairs, i.e., (s_{k-i}, y_{k-i}), $i = 0, \cdots, m-1$ for computing a search direction d_k, where

$$\begin{aligned} s_k &= x_k - x_{k-1}, \\ y_k &= \nabla f(x_k) - \nabla f(x_{k-1}), \end{aligned}$$

and k denotes the iteration number. While the standard BFGS method requires $O(n^2)$ memory, L-BFGS method needs only $O(mn)$ memory. Moreover, given $\nabla f(x_k)$, we can compute the search direction $d_k = -H^k \nabla f(x_k)$ in $O(mn)$ time. The L-BFGS method needs the line search satisfying the Wolfe condition to guarantee the global convergence. However the line search scheme may require a large number of function evaluations for ill-posed problems. Since the main burden of the L-BFGS method is function evaluation, it is preferable to reduce the number of function evaluations as possible.

The trust region method (TR-method) can guarantee global convergence without the line search. It is known that the TR-method needs less function evaluations than the line search [2, 3, 10]. In fact, the L-BFGS method combined with the TR-method [2] presents the nice performance for many benchmark problems in terms of the number of function evaluations. However, the TR-method must solve the nonconvex subproblem

$$\begin{array}{ll} \text{minimize} & f(x_k) + \nabla f(x_k)^T d + \dfrac{1}{2} d^T H_k^{-1} d \\ \text{subject to} & \|d\| \leq \Delta_k, \end{array} \tag{1.2}$$

in each step, where Δ_k is a positive parameter called a trust region radius. It takes much time to solve the subproblem (1.2) even if H_k is given by the L-BFGS scheme.

The regularized Newton method proposed by Ueda and Yamashita [19, 20, 21] implicitly solves (1.2). It computes the search direction d_k by solving the following linear equation.

$$(\nabla^2 f(x_k) + \mu_k I) d = -\nabla f(x_k), \tag{1.3}$$

where μ_k is a positive parameter called a regularized parameter. If the parameter μ_k equals to the Lagrange multiplier in the KKT condition of (1.2), then the search direction d given by (1.2) is the solution of the problem (1.2). Note that the linear equations (1.3) are much easier than the subproblem (1.2) of the TR-method. The regularized Newton method [19] controls the parameter μ_k insted of computing the step length to guarantee global convergence. However, since the regularized Newton method in [19] is based on the Newton method, it must compute the Hessian matrix of f. Until now, the regularized Newton method with the L-BFGS method has not been considerd.

In this paper, we propose a novel method that combines the L-BFGS method with the regularized Newton method. We call the proposed method the regularized L-BFGS method. It might be natural to adopt a solution of the linear equation $(B_k + \mu I)d =$

$-\nabla f(x)$ as the search direction, where B_k is the inverse of H_k computed by L-BFGS method. However, we cannot get B_k explicitly. Therefore, we construct an inverse approximate matrix of $(B_k + \mu I)^{-1}$ by the L-BFGS method using $(s_k, \hat{y}_k(\mu))$, where $\hat{y}_k(\mu) = y_k + \mu s_k$. The term μs_k in $\hat{y}_k(\mu)$ plays a role of the regularization. Then, the search direction d_k can be computed in the same order $O(mn)$ of the usual L-BFGS method. We also control the regularized parameter μ_k for global convergence as the regularized Newton method. To this end, we use the idea of updating trust region radius Δ_k in the TR-method. We show that the proposed algorithm has global convergence property. Furthermore, we conducts the numerical experiments for CUTEst [7].

The paper is organized as follows. In Section 2, we propose a regularized L-BFGS method which controls the regularized parameter at each iteration. In Section 3, we show its global convergence under some conditions. In Section 4, we discuss some implementation issues for practical use. In Section 5, some numerical results are presented, and Section 6 concludes the paper.

Throughout the paper, we use the following notations. For a vector $x \in \mathbb{R}^n$, $\|x\|$ denotes the Euclidean norm defined by $\|x\| := \sqrt{x^T x}$. For a symmetric matrix $M \in \mathbb{R}^{n\times n}$, we denote the maximum eigenvalue and the minimum eigenvalue of M as $\lambda_{\max}(M)$ and $\lambda_{\min}(M)$, respectively. Moreover, $\|M\|$ denotes the l_2 norm of M defined by $\|M\| := \sqrt{\lambda_{\max}(M^T M)}$. If M is the symmetric positive semidefinite matrix, then $\|M\| = \lambda_{\max}(M)$. Furthermore, if the matrix $M \in \mathbb{R}^{n\times n}$ is the positive (semi) definite, the minimum eigenvalue of M is positive (non-negative) , i.e., $\lambda_{\min}(M) > (\geq)\ 0$.
Next, we give the definitions of Lipschitz continuity.

Definition 1.1 *Let S be a subset of $\mathbb{R}^n$ and $f : S \to \mathbb{R}$.*

(i) *The function f is said to be Lipschitz continuous on S if there exists a positive constant L_f such that*

$$|f(x) - f(y)| \leq L_f \|x - y\|,\ \forall x, y \in S.$$

(ii) *Suppose that the function f is differentiable. ∇f is said to be Lipschitz continuous on S if there exists a positive constant L_g such that*

$$|\nabla f(x) - \nabla f(y)| \leq L_g \|x - y\|,\ \forall x, y \in S.$$

The constants L_f and L_g are called Lipschits constant.

2 The regularized L-BFGS method

In this section, we propose a regularized L-BFGS method that controls the regularized parameter at each iteration. In what follows, x_k denotes the k-th iterative point, B_k denotes the approximate Hessian of $f(x_k)$ and $H_k^{-1} = B_k$.

We consider to combine the L-BFGS method with the regularized Newton method (1.3). For that purpose, we may replace the Hessian $\nabla^2 f(x_k)$ in the equation (1.3) with the apporoximate Hessian B_k. That is, we define a search direction d_k as a solution of

$$(B_k + \mu I)^{-1} d_k = -\nabla f(x_k). \tag{2.1}$$

However, since the L-BFGS method updates H_k, it is not easy to construct B_k explicitly. Furthermore, even if we get B_k, it takes much time to solve the linear equation (2.1) in large scale case.

Now, we may regard $B_k + \mu I$ as an approximation of $\nabla^2 f(x) + \mu I$. Since B_k is the approximate Hessian of $f(x_k)$, the matrix $B_k + \mu I$ is an approximate Hessian of $f(x) + \frac{\mu}{2}\|x\|^2$. The L-BFGS method uses the vector pair (s_k, y_k) to construct the approximate Hessian, where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$. Note that y_k consists of the

gradients of f. Therefore, when we compute the approximate Hessian of $f(x)+\frac{\mu}{2}\|x\|^2$, we use the gradients of $f(x)+\frac{\mu}{2}\|x\|^2$. That is, we adopt the following $\hat{y}_k(\mu)$ instead of y_k.

$$\hat{y}_k(\mu) = (\nabla f(x_{k+1}) + \mu x_{k+1}) - (\nabla f(x_k) + \mu x_k) = y_k + \mu s_k.$$

Let $\hat{H}_k(\mu)$ be the matrix constructed by using the L-BFGS method with vector pair $(s_k, \hat{y}_k(\mu))$ and an appropriate initial matrix $\hat{H}_k^{(0)}(\mu)$. Then the search direction $d_k = -\hat{H}_k(\mu)\nabla f(x_k)$ is calculated in $O(mn)$ time, which is the same as that for the original L-BFGS.

Note that the conditions that $s_k^T\hat{y}_k(\mu) > 0$ and $\hat{H}_k^{(0)}$ is positive definite are necessary for the positive definiteness of $\hat{H}_k(\mu)^{-1}$. When the condition $s_k^T\hat{y}_k(\mu) > 0$ is not satisfied, we may replace $\hat{y}_k(\mu)$ by $\tilde{y}_k(\mu)$,

$$\tilde{y}_k(\mu) = y_k + \left(\max\left\{0, \frac{-s_k^T y_k}{||s_k||^2}\right\} + \mu\right) s_k.$$

Then, the condition $s_k^T\tilde{y}_k(\mu) > 0$ is satisfied, because

$$s_k^T\tilde{y}_k(\mu) = \max\{0, s_k^T y_k\} + \mu\|s_k\|^2 > 0.$$

In what follows, $\hat{H}_k(\mu)$ is the matrix constructed by L-BFGS method using the initial matrix $\hat{H}_k^{(0)}(\mu)$ and the vector pairs $(s_{k-i}, \hat{y}_{k-i}(\mu)),\ i = 1, \cdots, m$ and the serch direction is $d_k(\mu) = -\hat{H}_k(\mu)\nabla f(x_k)$. The L-BFGS method uses $\gamma_k I$ as the initial matrix $H_k^{(0)}$. Since $(B_k^{(0)})^{-1} = H_k^{(0)}$ and $\hat{H}_k^{(0)}$ is an approximation of $(B_k^{(0)} + \mu I)^{-1}$, we can set the initial matrix $\hat{H}_k^{(0)}(\mu)$ as,

$$\hat{H}_k^{(0)}(\mu) = (B_k^{(0)} + \mu I)^{-1} = \left(\frac{1}{\gamma_k} + \mu\right)^{-1} I = \frac{\gamma_k}{1+\gamma_k\mu} I.$$

The proposed method sets the next step $x_{k+1} = x_k + d_k(\mu)$ without a step length. It controls the parameter μ to guarantee the global convergence as in [19]. Note that the regularized parameter μ and $d_k(\mu)$ have following relations.

$$\begin{cases} d_k(\mu) \to -B_k^{-1}\nabla f(x_k) & (\mu \to 0) \\ d_k(\mu) \to -\frac{1}{\mu}\nabla f(x_k) & (\mu \to \infty). \end{cases}$$

These directions correspond to the steepest decent direction and the L-BFGS direction, respectively. Figure 2.1 shows the relations of these search directions. Namely, the large μ gives the decent direction of f, while the small one gives the L-BFGS direction. We use the ideas of updating trust region radius in the TR-method to control μ. In order to find an appropriate search direction, we use the ratio of the reduction of the objective function value to that of the model function value like the TR-method. We define a ratio function $r_k(d_k(\mu), \mu)$ by

$$r_k(d_k(\mu), \mu) = \frac{f(x_k) - f(x_k + d_k(\mu))}{f(x_k) - q_k(d_k(\mu), \mu)}, \tag{2.2}$$

where $q_k : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is following model function at x_k,

$$q_k(d_k(\mu), \mu) = f(x_k) + \nabla f(x_k)^T d_k(\mu) + \frac{1}{2} d_k(\mu)^T \hat{H}_k(\mu)^{-1} d_k(\mu).$$

Note that we do not have to compute the matrix $\hat{H}_k(\mu)^{-1}$ explicitly in $q_k(d_k, \mu)$. Since $d_k(\mu) = -\hat{H}_k(\mu)\nabla f(x_k)$, we have $d_k(\mu)^T\hat{H}_k(\mu)^{-1}d_k(\mu) = -d_k(\mu)^T\nabla f(x_k)$. If the ratio $r_k(d_k(\mu), \mu)$ is large, i.e., the reduction of the objective function f is sufficiently large as compared to that of the model function, we adopt $d_k(\mu)$ and decrease the parameter μ. On

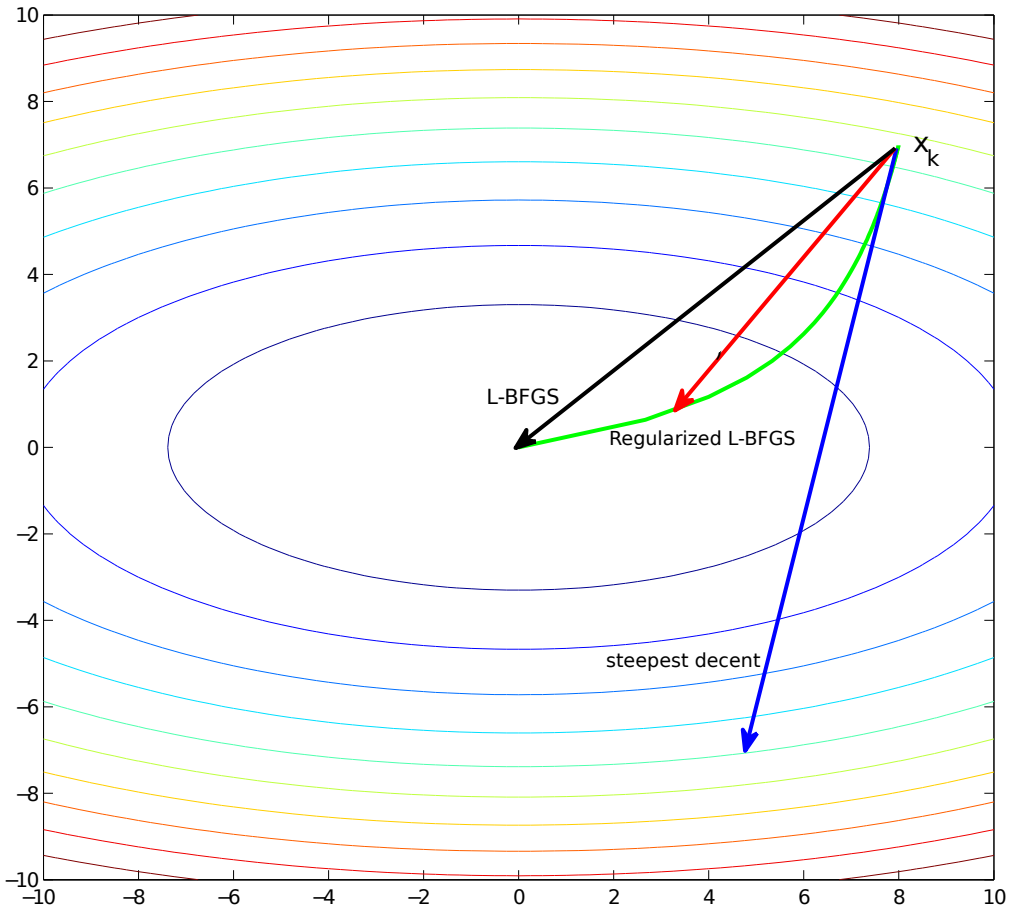

Figure 2.1: The implementations of μ and $d_k(\mu)$.

the other hand, if $r_k(d_k, \mu)$ is small, i.e., the reduction $f(x_k) - f(x_k + d_k)$ is not enough, we increase μ and compute $d_k(\mu)$ once again.

Based on the above ideas, we propose the following method. We call the proposed algorithm the regularized L-BFGS method.

Algorithm 2.1 Regularized L-BFGS method

Step 0 Choose the parameters $\mu_0, \mu_{\min}, \gamma_1, \gamma_2, \eta_1, \eta_2, m$ such that $0 < \mu_{\min} \leq \mu_0, 0 < \gamma_1 \leq 1 < \gamma_2, 0 < \eta_1 < \eta_2 \leq 1$ and $m > 0$. Choose an initial point $x_0 \in \mathbb{R}^n$ and an initial matrix $\hat{H}_0^{(0)}$. Set $k := 0$.

Step 1 If some stopping criteria are satisfied, then terminate. Otherwise, go to Step 2.

Step 2

Step 2-0 Set $l_k := 0$ and $\bar{\mu}_{l_k} = \mu_k$.

Step 2-1 Compute $d_k(\bar{\mu}_{l_k})$ by using the L-BFGS method.

Step 2-2 Compute $r_k(d_k(\bar{\mu}_{l_k}), \bar{\mu}_{l_k})$. If $r_k(d_k(\bar{\mu}_{l_k}), \bar{\mu}_{l_k}) < \eta_1$, then update $\bar{\mu}_{l_k+1} = \gamma_2 \bar{\mu}_{l_k}$, set $l_k = l_k + 1$, and go to Step 2-1. Otherwise, go to Step 3.

Step 3 If $\eta_1 \leq r_k(d_k(\bar{\mu}_{l_k}), \bar{\mu}_{l_k}) < \eta_2$, then update $\mu_{k+1} = \bar{\mu}_{l_k}$.
If $r_k(d_k(\bar{\mu}_{l_k}), \bar{\mu}_{l_k}) \geq \eta_2$, then update $\mu_{k+1} = \max[\mu_{\min}, \gamma_1 \bar{\mu}_{l_k}]$.
Update $x_{k+1} = x_k + d_k(\bar{\mu_{l_k}})$. Set $k = k + 1$, and go to Step 1.

In Step 2-1 we compute $d_k(\mu)$ from $(s_k, \hat{y}_k(\mu))$ by the L-BFGS updating scheme in [11], The details of Step 2-1 are written as follows.

Algorithm 2.2 Regularized L-BFGS update at kth iteration with parameter μ

Step 0 Set $p \leftarrow \nabla f(x_k)$.

Step 1 Repeat the following process with $i = k-1, k-2, \cdots, k-t$:

$$\begin{aligned} r_i &\leftarrow \tau_i s_i^T p \\ p &\leftarrow p - r_i(y_k + \mu s_k), \end{aligned}$$

where $\tau_i = (s_i^T(y_k + \mu s_k))^{-1}$.

Step 2 Set $q \leftarrow \hat{H}_k^{(0)}(\mu) p$.

Step 3 Repeat the following process with $i = k-t, k-t+1, \cdots, k-1$:

$$\begin{aligned} \beta &\leftarrow \tau_i (y_k + \mu s_k)^T q \\ q &\leftarrow q + (r_i - \beta) s_i. \end{aligned}$$

Step 4 Get the search direction by $d_k(\mu) = -q$.

Note that, since μ_k may vary in each iteration, it is impractical to store $y_k(\mu)$ for all μ_k. Fortunately, the Algorithm 2.2 uses y_k and s_k. Thus we do not have to store $\hat{y}_k(\mu)$ when the regularized parameter is changed, we get $\hat{y}_k(\mu)$ from s_k and y_k, immediately.

3 Global convergence

In this section, we show global convergence of the proposed algorithm. To this end, we need the following assumptions.

Assumption 3.1

(i) *The objective function f is twice continuously defferentiable.*

(ii) *The level set of f at the initial point x_0 is compact, i.e., $\Omega = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ is compact.*

(iii) *There exists positive constants M_1 and M_2 such that*

$$M_1\|z\|^2 \leq z^T\nabla^2 f(x)z \leq M_2\|z\|^2, \ \forall x \in \Omega.$$

(iv) *There exists the minimum $f_{\min}$ of f.*

(v) *There exists a constant $\underline{\gamma}$ such that $\gamma_k \geq \underline{\gamma} > 0$, for all k.*

The above assumptions are same as those for the global convergence of the original L-BFGS method.

Under the assumptions, we have the following properties. First, we define

$$G(x) = \nabla^2 f(x), \quad G_k = G(x_k), \quad \bar{G}_k = \int_0^1 G(x_k + \tau s_k)d\tau.$$

It then follows from the Taylor's theorem that

$$f(x_k + d_k(\mu)) = f(x_k) + \nabla f(x_k)^T d_k(\mu) + \frac{1}{2}\int_0^1 d_k(\mu)^T G(x_k + \tau d_k(\mu))d_k(\mu)d\tau.$$

Furthermore, since $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$, we have

$$y_k = \bar{G}_k s_k. \tag{3.1}$$

$$\hat{y}_k(\mu) = y_k + \mu s_k = (\bar{G}_k + \mu I)s_k. \tag{3.2}$$

It follows from Assumption 3.1 (iii) that $\lambda_{\min}(\bar{G}_k) \geq M_1$ and $\lambda_{\max}(\bar{G}_k) \leq M_2$. Therefore, we have that

$$\begin{aligned} M_1\|s\|^2 &\leq s_k^T y_k \leq M_2\|s_k\|^2, \\ \frac{1}{M_2}\|y_k\|^2 &\leq s_k^T y_k \leq \frac{1}{M_1}\|y_k\|^2, \\ (M_1 + \mu)\|s_k\|^2 &\leq s_k^T \hat{y}_k(\mu) \leq (M_2 + \mu)\|s_k\|^2. \end{aligned} \tag{3.3}$$

Since the sequence $\{x_k\}$ is included in the compact set Ω and f is twice continuously differentiable under Assumption 3.1 (ii), there exists a positive constant L_f such that

$$\|\nabla f(x_k)\| \leq L_f, \quad \text{for all } k. \tag{3.4}$$

Now, we investigate the behavior of the eigenvalues of $\hat{B}_k(\mu)$, where $\hat{B}_k(\mu)$ is the inverse of $\hat{H}_k(\mu)$. Note that the matrix $\hat{B}_k(\mu)$ is constructed by the BFGS formula with the vector pairs $(s_k, \hat{y}_k(\mu))$ and the initial matrix $\hat{B}_k^{(0)}(\mu) = \hat{H}_k^{(0)}(\mu)^{-1}$. Thus, we have

$$\begin{aligned} \hat{B}_k(\mu) &= \hat{B}_k^{(\tilde{m}_k)}(\mu) \\ \hat{B}_k^{(l+1)}(\mu) &= \hat{B}_k^{(l)}(\mu) - \frac{\hat{B}_k^{(l)}(\mu)s_{j_l}s_{j_l}^T\hat{B}_k^{(l)}(\mu)}{s_{j_l}^T\hat{B}_k^{(l)}(\mu)s_{j_l}} + \frac{y_{j_l}y_{j_l}^T}{y_{j_l}^T s_{j_l}}, \quad l = 0, \cdots, \tilde{m}_k - 1 \end{aligned} \tag{3.5}$$

where $\tilde{m}_k = \min\{k+1, m\}$ and $j_l = k - \tilde{m} + l$. Note that these expressions are used in [4, 9].

We now focus on the trace and determinant of $\hat{B}_k(\mu)$. At first, we show that the trace of $\hat{B}_k^{(l)}(\mu)$ is $O(\mu)$.

Lemma 3.1 *Suppose that Assumption 3.1 holds. Then,*

$$\mathrm{tr}(\hat{B}_k^{(l)}(\mu)) \le M_3 + (2m+n)\mu, \quad l = 0, \cdots, \ \tilde{m}_k$$

where

$$M_3 = \frac{n}{\underline{\gamma}} + mM_2.$$

Proof We have from Assumption 3.1, (3.1) and (3.3) that

$$\begin{aligned}
\frac{\|\hat{y}_k(\mu)\|^2}{s_k^T \hat{y}_k(\mu)} &= \frac{\|y_k\|^2 + 2\mu s_k^T y_k + \mu^2 \|s_k\|}{s_k^T y_k + \mu \|s_k\|^2} \\
&= \frac{\|y_k\|^2 + \mu s_k^T y_k}{s_k^T y_k + \mu \|s_k\|^2} + \frac{\mu(s_k^T y_k + \mu \|s_k\|^2)}{s_k^T y_k + \|s_k\|^2} \\
&\le \frac{\|y_k\|^2 + \mu s_k^T y_k}{s_k^T y_k} + \mu \\
&\le \frac{\|y_k\|^2}{\frac{1}{M_2}\|y_k\|^2} + 2\mu \\
&= M_2 + 2\mu. \qquad (3.6)
\end{aligned}$$

From updating fomula (3.5) of the matrix $\hat{B}_k(\mu)$,

$$\mathrm{tr}(\hat{B}_k^{(l)}(\mu)) = \mathrm{tr}(\hat{B}_k^{(0)}(\mu)) + \sum_{t=0}^{l-1} \left(-\frac{\|\hat{B}_k^{(t)}(\mu) s_{j_t}\|^2}{s_{j_t}^T \hat{B}_k^{(t)}(\mu) s_{j_t}} + \frac{\|\hat{y_{j_t}}(\mu)\|^2}{s_{j_t}^T \hat{y}_{j_t}(\mu)} \right),$$

where

$$\frac{\|\hat{B}_k^{(t)}(\mu) s_{j_t}\|^2}{s_{j_t}^T \hat{B}_k^{(t)}(\mu) s_{j_t}} \ge 0.$$

It then follows from (3.6) that

$$\begin{aligned}
\mathrm{tr}(\hat{B}_k^{(l)}(\mu)) &\le \mathrm{tr}(\hat{B}_k^{(0)}(\mu)) + \sum_{t=0}^{l-1} \frac{\|\hat{y_{j_t}}(\mu)\|^2}{s_{j_t}^T \hat{y}_{j_t}(\mu)} \\
&\le \mathrm{tr}(\hat{B}_k^{(0)}(\mu)) + l(M_2 + 2\mu) \\
&= n\left(\frac{1}{\gamma_k} + \mu\right) + l(M_2 + 2\mu) \\
&\le n\left(\frac{1}{\underline{\gamma}} + \mu\right) + m(M_2 + 2\mu) \\
&\le M_3 + (2m+n)\mu.
\end{aligned}$$

This completes the proof. □

The next lemma gives a lower bound of the determinant of $\hat{B}_k(\mu)$.

Lemma 3.2 *Suppose that Assumption 3.1 holds. Then,*

$$\det(\hat{B}_k(\mu)) \ge M_4 \mu^n,$$

where

$$M_4 = \left(\frac{1}{2m+n}\right)^m.$$

Proof Note that the determinant of the approximate matrix updated by the BFGS updating scheme has the following property [14, 15].

$$\det(\hat{B}_k^{(l+1)}(\mu)) = \det(\hat{B}_k^{(l)}(\mu)) \frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{s_{j_l}^T \hat{B}_{k-1}^{(l)}(\mu) s_{j_l}}.$$

Then we have

$$\begin{aligned}
\det(\hat{B}_k(\mu)) &= \det(\hat{B}_k^{(0)}(\mu)) \prod_{l=0}^{\tilde{m}-1} \frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{s_{j_l}^T \hat{B}_{k-1}^{(0)}(\mu) s_{j_l}} \\
&= \det(\hat{B}_k^{(0)}(\mu)) \prod_{l=0}^{\tilde{m}-1} \frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{s_{j_l}^T s_{j_l}} \frac{s_{j_l}^T s_{j_l}}{s_{j_l}^T \hat{B}_k^{(l)}(\mu) s_{j_l}} \\
&\geq \det(\hat{B}_k^{(0)}(\mu)) \prod_{l=0}^{\tilde{m}-1} \frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{s_{j_l}^T s_{j_l}} \frac{s_{j_l}^T s_{j_l}}{\lambda_{\max}(\hat{B}_k^{(l)}(\mu)) s_{j_l}^T s_{j_l}} \\
&= \det(\hat{B}_k^{(0)}(\mu)) \prod_{l=0}^{\tilde{m}-1} \frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{\|s_{j_l}\|^2} \frac{1}{\lambda_{\max}(\hat{B}_k^{(l)}(\mu))}.
\end{aligned}$$

Since $B_k^{(0)}(\mu)$ is symmetric positive definite, Lemma 3.1 implies that $\lambda_{\max}(\hat{B}_k^{(l)}(\mu)) \geq M_3 + (2m+n)\mu$. Furthermore, we have $\frac{s_{j_l}^T \hat{y}_{j_l}(\mu)}{\|s_{j_l}\|^2} \geq M_1 + \mu$ from (3.3). Therefore, it follows that

$$\begin{aligned}
\det(\hat{B}_k(\mu)) &\geq \det(\hat{B}_k(\mu)^{(0)}) \left(\frac{M_1 + \mu}{M_3 + (2m+n)\mu} \right)^{\tilde{m}} \\
&= \det\left(\frac{1 + \gamma_k \mu}{\gamma_k} I \right) \left(\frac{M_1 + \mu}{M_3 + (2m+n)\mu} \right)^{\tilde{m}} \\
&\geq \left(\frac{1}{\gamma_k} + \mu \right)^n \left(\frac{1}{2m+n} \right)^{\tilde{m}} \\
&\geq \left(\frac{1}{2m+n} \right)^m \mu^n \\
&= M_4 \mu^n.
\end{aligned}$$

This completes the proof. □

From the above two lemmas we have $\lambda_{\max}(\hat{H}_k(\mu)) \to 0 \propto \mu \to \infty$.

Lemma 3.3 *Suppose that Assumption 3.1 holds. Then, for all $k \geq 0$,*

$$\lambda_{\max}(\hat{H}_k(\mu)) \leq M_5 \frac{1}{\mu}, \ \forall \mu \in [\mu_{\min}, \infty),$$

where

$$M_5 = \frac{1}{M_4 n^{n-1}} \left(\frac{M_3}{\mu_{\min}} + (2m+n) \right)^{n-1}.$$

Furthermore, $\lim_{\mu \to \infty} \lambda_{\max}(\hat{H}_k(\mu)) = 0$.

Proof We have from Lemma 3.1 and 3.2 that

$$\begin{aligned}
\mathrm{tr}(\hat{B}_k(\mu)) &\leq M_3 + (2m+n)\mu, \\
\det(\hat{B}_k(\mu)) &\geq M_4 \mu^n.
\end{aligned}$$

Since $\hat{B}_k(\mu)$ is symmetric positive definite, we have

$$\begin{aligned} \mathrm{tr}(\hat{B}_k(\mu)) &\geq n\lambda_{\min}(\hat{B}_k(\mu)) \\ \det(\hat{B}_k(\mu)) &\leq \lambda_{\min}(\hat{B}_k(\mu))\{\lambda_{\max}(\hat{B}_k(\mu))\}^{n-1}. \end{aligned}$$

Therefore, we have

$$\begin{aligned} \lambda_{\min}(\hat{B}_k(\mu)) &\geq \frac{\det(\hat{B}_k(\mu))}{\{\lambda_{\max}(\hat{B}_k(\mu))\}^{n-1}} \\ &\geq \frac{M_4\mu^n}{\{\frac{1}{n}(M_3+(2m+n)\mu)\}^{n-1}}. \end{aligned}$$

It then follows from Assumption 3.1 (v) that

$$\begin{aligned} \lambda_{\max}(\hat{H}_k(\mu)) &= \frac{1}{\lambda_{\min}(\hat{H}_k^{-1}(\mu))} \\ &= \frac{1}{\lambda_{\min}(\hat{B}_k(\mu))} \\ &\leq \frac{\{\frac{1}{n}(M_3+(2m+n)\mu)\}^{n-1}}{M_4\mu^n} \\ &= \frac{1}{M_4 n^{n-1}}\left(\frac{M_3+(2m+n)\mu}{\mu}\right)^{n-1}\frac{1}{\mu}. \end{aligned} \tag{3.7}$$

Since $\mu \geq \mu_{\min}$, we have

$$\frac{M_3+(2m+n)\mu}{\mu} = \frac{M_3}{\mu} + (2m+n) \leq \frac{M_3}{\mu_{\min}} + (2m+n).$$

It then follows from (3.7) that

$$\begin{aligned} \lambda_{\max}(\hat{H}_k(\mu)) &\leq \frac{1}{M_4 n^{n-1}}\left(\frac{M_3}{\mu_{\min}} + (2m+n)\right)^{n-1}\frac{1}{\mu} \\ &= M_5\frac{1}{\mu}. \end{aligned}$$

and hence we have

$$\lim_{\mu\to\infty} \lambda_{\max}(\hat{H}_k(\mu)) = 0.$$

This completes the proof. □

Now we give an upper bound of $\|d_k(\mu)\|$.

Lemma 3.4 *Suppose that Assumption 3.1 holds. Then,*

$$\|d_k(\mu)\| \leq U_d,$$

where

$$U_d = \frac{L_f M_5}{\mu_{\min}}.$$

Proof We have from the definition of $d_k(\mu)$, (3.4) and Lemma 3.3 that

$$\begin{aligned} \|d_k(\mu)\| &= \|\hat{H}_k(\mu)\nabla f(x_k)\| \\ &\leq \|\hat{H}_k(\mu)\|\|\nabla f(x_k)\| \\ &= \lambda_{\max}(\hat{H}_k(\mu))\|\nabla f(x_k)\| \\ &\leq \lambda_{\max}(\hat{H}_k(\mu))L_f \\ &\leq \frac{L_f M_5}{\mu} \\ &\leq \frac{L_f M_5}{\mu_{\min}} = U_d. \end{aligned}$$

This completes the proof. □

Lemma 3.4 implies that

$$x_k + \nu d_k(\mu) \in \Omega + \mathrm{B}(0, U_d), \quad \forall \nu \in [0,1], \quad \forall \mu \in [\mu_{\min}, \infty), \quad \forall k \geq 0.$$

Moreover, since $\Omega + \mathrm{B}(0, U_d)$ is compact and f is twice continuously differentiable, $\nabla f(x_k)$ is Lipshitz continuous on $\Omega + \mathrm{B}(0, U_d)$. That is, there exists a positive constant L_g such that

$$\|\nabla^2 f(x_k)\| \leq L_g, \quad \forall x_k \in \Omega + \mathrm{B}(0, U_d). \tag{3.8}$$

Next, we investigate the condition of μ to satisfy the terminate condition $r_k(d_k(\mu), \mu) \geq \eta_1$ of inner iterations at Step 2-2 of Algorithm 2.1.

Lemma 3.5 *Suppose that Assumption 3.1 holds. Then we have*

$$f(x_k) - f(x_k + d_k(\mu)) - \eta_1(f(x_k) - q_k(d_k(\mu), \mu)) \geq \frac{1}{2}((2-\eta_1)\lambda_{\min}(\hat{H}_k(\mu)^{-1}) - L_g)\|d_k(\mu)\|^2.$$

Proof We have from the Taylor's theorem that

$$\begin{aligned} f(x_k + d_k(\mu)) &= f(x_k) + \int_0^1 \nabla f(x_k + \tau d_k(\mu))^T d_k(\mu) d\tau \\ &= f(x_k) + \nabla f(x_k)^T d_k(\mu) + \int_0^1 (\nabla f(x_k + \tau d_k(\mu)) - \nabla f(x_k))^T d_k(\mu) d\tau. \end{aligned}$$

From the Lipschitz continuously of $\nabla f(x_k)$ of (3.8), we get

$$\begin{aligned} f(x_k) &- f(x_k + d_k(\mu)) - \eta_1(f(x_k) - q_k(d_k(\mu), \mu)) \\ &= -\nabla f(x_k)^T d_k(\mu) - \int_0^1 (\nabla f(x_k + \tau d_k(\mu)) - \nabla f(x_k))^T d_k(\mu) d\tau - \frac{\eta_1}{2} d_k(\mu)^T (\hat{H}_k(\mu)^{-1}) d_k(\mu) \\ &= \frac{(2-\eta_1)}{2} d_k(\mu)^T (\hat{H}_k(\mu)^{-1}) d_k(\mu) - \int_0^1 (\nabla f(x_k + \tau d_k(\mu)) - \nabla f(x_k))^T d_k(\mu) d\tau \\ &\geq \frac{(2-\eta_1)}{2} \lambda_{\min}(\hat{H}_k(\mu)^{-1}) \|d_k(\mu)\|^2 - \int_0^1 L_g \tau \|d_k(\mu)\|^2 d\tau \\ &= \frac{1}{2}((2-\eta_1)\lambda_{\min}(\hat{H}_k(\mu)^{-1}) - L_g)\|d_k(\mu)\|^2. \end{aligned}$$

This completes the proof. □

From Lemma 3.5, if μ satisfies

$$\lambda_{\min}(\hat{H}_k^{-1}(\mu)) \geq \frac{L_g}{2-\eta_1}, \tag{3.9}$$

then we have

$$r_k(d_k(\mu), \mu) \geq \eta_1, \tag{3.10}$$

that is, the inner loops of Algorithm 2.1 must terminate.

Next, we give the upper bound of the parameter μ_k.

Lemma 3.6 *Suppose that Assumption 3.1 hold. Then, for any $k \geq 0$,*

$$\mu_k^* \leq U_\mu,$$

where

$$U_\mu = \gamma_2 M_5 \frac{L_g}{2-\eta_1}.$$

Proof If $\bar{\mu}_{l_k}$ satisfies (3.9), then $r_k(d_k(\bar{\mu}_{l_k}), \bar{\mu}_{l_k}) \geq \eta_1$ from Lemma 3.5. Therefore, the inner loops must terminate and we set $\mu_k^* = \bar{\mu}_{l_k}$.

Now, we give the terminate condition on μ of the inner loop. We have from Lemma 3.3 that

$$\begin{aligned} \lambda_{\min}(\hat{H}_k^{-1}(\mu)) &= \frac{1}{\lambda_{\max}(\hat{H}_k(\mu))} \\ &\geq \frac{\mu}{M_5}. \end{aligned} \tag{3.11}$$

It then follows from (3.9) that the terminate condition of inner loop holds when

$$\mu \geq M_5 \frac{L_g}{2-\eta_1}. \tag{3.12}$$

Note that, if the inner loop terminates at l_k, then (3.12) does not hold with $\mu = \mu_{l_k-1}$, that is,

$$\bar{\mu}_{l_k-1} < M_5 \frac{L_g}{2-\eta_1}.$$

Since $\mu_k^* = \bar{\mu}_{l_k} = \gamma_2 \bar{\mu}_{l_k-1}$, we have

$$\mu_k^* = \gamma_2 \bar{\mu}_{l_k-1} < \gamma_2 M_5 \frac{L_g}{2-\eta_1} = U_\mu. \tag{3.13}$$

This completes the proof. □

Next, we give a lower bound of the reduction of the model function q_k.

Lemma 3.7 *Suppose that Assumption 3.1 holds. Then we have*

$$f(x_k) - q_k(d_k(\mu), \mu) \geq M_6 \|\nabla f(x_k)\|^2,$$

where

$$M_6 = \frac{1}{2(M_3 + (2m+n)\mu_{\min})}.$$

Proof It follows from the definition of the model function $q_k(d_k(\mu), \mu)$ and Lemma 3.1, 3.6 that

$$\begin{aligned} f(x_k) - q_k(d_k(\mu), \mu) &= -\frac{1}{2} d_k(\mu)^T (\hat{H}_k^{-1}(\mu)) d_k(\mu) - \nabla f(x_k)^T d_k(\mu) \\ &= -\frac{1}{2} \nabla f(x_k)^T \hat{H}_k(\mu) \nabla f(x_k) + \nabla f(x_k)^T \hat{H}_k(\mu) \nabla f(x_k) \\ &= \frac{1}{2} \nabla f(x_k)^T \hat{H}_k(\mu) \nabla f(x_k) \\ &\geq \frac{1}{2} \lambda_{\min}(\hat{H}_k(\mu)) \|\nabla f(x_k)\|^2 \\ &= \frac{\|\nabla f(x_k)\|^2}{2\lambda_{\max}(\hat{H}_k^{-1}(\mu))} \\ &= \frac{\|\nabla f(x_k)\|^2}{2\lambda_{\max}(\hat{B}_k(\mu))} \\ &\geq \frac{\|\nabla f(x_k)\|^2}{2\mathrm{tr}(\hat{B}_k(\mu))} \\ &\geq \frac{\|\nabla f(x_k)\|^2}{2(M_3 + (2m+n)\mu)} \\ &= \frac{1}{2(M_3 + (2m+n)U_\mu)} \|\nabla f(x_k)\|^2 \\ &= M_6 \|\nabla f(x_k)\|^2. \end{aligned}$$

This completes the proof. □

From this lemma, we give a lower bound of the reduction of the objective function value when x_k is not a stationary point.

Lemma 3.8 *Suppose that Assumption 3.1 holds. If there exits a positive constant ϵ_g such that $\|\nabla f(x_k)\| \geq \epsilon_g$, then we have $f(x_k) - f(x_{k+1}) \geq \rho\epsilon_g^2$, where $\rho = \eta_1 M_6$.*

Proof It follows from Lemma 3.6 and 3.7 that

$$\begin{aligned} f(x_k) - f(x_{k+1}) &\geq \eta_1(f(x_k) - q_k(d_k(\mu_k^*), \mu_k^*)) \\ &\geq \eta_1 M_6 \|\nabla f(x_k)\|^2 \\ &\geq \rho\epsilon_g^2. \end{aligned}$$

This completes the proof. □

Now, we are at the position to prove the main theorem of this section.

Theorem 3.1 *Suppose that Assumption 3.1 holds. Then, $\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0$ or there exists $K \geq 0$ such that $\|\nabla f(x_K)\| = 0$.*

Proof Suppose the contrary, i.e., there exists a positive constant ϵ_g such that $\|\nabla f(x_k)\| \geq \epsilon_g$ for all $k \geq 0$.
It follows from Lemma 3.8 that

$$\begin{aligned} f(x_0) - f(x_k) &= \sum_{j=0}^{k-1} (f(x_j) - f(x_{j+1})) \\ &\geq \sum_{j=0}^{k-1} \rho\epsilon_g^2 \\ &= \rho\epsilon_g^2 k. \end{aligned}$$

Taking $k \to \infty$, the right hand side of the last inequality goes to infinity, and hence

$$\lim_{k\to\infty} f(x_k) = -\infty.$$

This contradicts the existence of $f_{\min}$ in Assumption 3.1 (iv). This completes the proof. □

4 Implementation issues

In this section, we discuss some implementation issues for practical use of the regularized L-BFGS method. **Scaling parameter γ_k in the initial matrix.**
The regularized L-BFGS method uses the following initial matrix in each iteration.

$$\hat{H}_k^{(0)}(\mu) = \frac{\gamma_k}{1 + \gamma_k \mu} I.$$

The parameter γ_k represents a scale of $\nabla^2 f(x)$. Thus, We exploits the ideas on the scaling parameter γ_k used in [1, 3, 11, 16, 17], that is, we set

$$\gamma_k = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2}.$$

It is known that the L-BFGS method with this scaling initial matrix has an efficient performance in [3, 11]. Note that $\gamma_k > 0$ for the positive definiteness of $\hat{H}_k^{(0)}(\mu)$. If $s_{k-1}^T y_{k-1} < \alpha\|s_{k-1}\|^2$, then we set $\gamma_k = \alpha\frac{\|s_{k-1}\|^2}{\|y_{k-1}\|^2}$, where α is a quite small positive constant.
Nonmonotone decresing technique.

In Algorithm 2.1, we control the regularized parameter μ to satisfy the decent condition $f(x_{k+1}) < f(x_k)$. However, μ sometimes becomes quite large when we treat the ill-posed problems. In this situation, we need a large number of function evaluations. Therefore, we use the ideas of the nonmonotone line search technique [8, 18] to overcome the difficulty. We replace the ratio function $r_k(d_k(\mu), \mu)$ with the following new ratio function $\bar{r}_k(d_k(\mu), \mu)$.

$$\bar{r}_k(d_k(\mu), \mu) = \frac{\max_{0 \le j \le m(k)} f(x_{k-j}) - f(x_k + d_k(\mu))}{f(x_k) - q_k(d_k(\mu), \mu)},$$

where

$$m(0) = 0, \quad 0 \le m(k) \le \min\{m(k-1) + 1, M\},$$

and M is a nonnegative integer constant.This modification still keeps the global convergence of the regularized L-BFGS method.

In the numerical experiments reported in the next section, when $k < M$, then we use the original ratio function $r_k(d_k(\mu), \mu)$. On the other hand, if $k \ge M$, then we use the new ratio function $\bar{r}_k(d_k(\mu), \mu)$.

Use of the gradient information of the rejected steps.
We focus on the rejected steps. Here, the rejected steps occur in Step 2-2 of the regularized L-BFGS algorithm. Namely, if a l_k holds the condition $r_k(d_k(\mu_{l_k}), \mu_{l_k}) \le \eta_1$ in Step 2-2, then l_k is the rejected step and we may exploits the information on the gradient at the rejeceted step. These gradients contain important information about the behavior of the objective function. Therefore, we store (s_{l_k}, y_{l_k}) given at these rejected step to update the approximate Hessian. Here, we do not have to increase the number m of pairs. When we store the new pair (s_{l_k}, y_{l_k}) at the rejected step, we discard the oldest vector pair. It is known that the L-BFGS method with this modification has a nice performance in [2].

5 Numerical results

In this section, we report some numerical results for the proposed algorithm from the following point of views.

1. The influences of some parameters contained in the Algorithm 2.1.
2. Comparison of the proposed algorithm with the original L-BFGS method[13].

In each experiment, benchmark problems were chosen from CUTEst [7]. All algorithms were coded in FORTRAN90 and run on a machine with Intel Core i5 1.7 GHz CPU and 4GB RAM. We used the original L-BFGS algorithm coded by Nocedal [13] without changing the default settings. We adopted an initial point x_0 given by CUTEst, and set the termination criteria as

$$\frac{\|f(x_k)\|}{\max(1, \|x_k\|)} < \epsilon \quad \text{and} \quad n_f > 10000,$$

where n_f is the number of function evaluations. These criteria are similar to that in [13]. In what follows, we regard the trial such that $n_f > 10000$ as failure.

We compare the algorithms by using the following distribution function proposed in [6]. Let $\mathcal{S}$ be a set of solvers, and let $\mathcal{P}_\mathcal{S}$ be a set of problems that can be solved by all methods in $\mathcal{S}$. We denote a measure for evaluation required to solve a problem p by a solver s as $t_{p,s}$, and the best $t_{p,s}$ for each p as t_p^*, i.e.,$t_p^* := \min\{t_{p,s} | s \in \mathcal{S}\}$. Then the distribution function $F_s^\mathcal{S}(\tau)$ for a method s is defined by

$$F_s^\mathcal{S}(\tau) = \frac{|\{p \in \mathcal{P}_\mathcal{S} | t_{p,s} \le \tau t_p^*\}|}{|\mathcal{P}_\mathcal{S}|}, \quad \tau \ge 1.$$

The alorithm whose $F_s^\mathcal{S}(\tau)$ is close to 1 is considered to be superior to the other algorithms in $\mathcal{S}$.

5.1 The influences of some parameters in Algorith2.1

The proposed algorithm has several parameters. Therefore, we need to investigate the influences of these parameters, and choose the favorable ones.

First of all, we focus on γ_1 and γ_2 used to control the regularized parameter μ. The other parameters are set as follows.

$$\eta_1 = 0.01,\ \eta_2 = 0.9,\ \mu_{\min} = 1.0 \times 10^{-3},\ m = 5.$$

The nonmonotone parameter M is set to 8. We compare the parameter sets $P_1, \cdots, P_9$ in Table 5.1. The table also shows the number of successes and the success rate for all 313 test problems.

Table 5.1: The number of successes and the rate of success at each (γ_1, γ_2).

	γ_1	γ_2	The number of successes	The rate of success(%)
P_1	0.1	2.0	266	85.0
P_2	0.1	5.0	267	85.3
P_3	0.1	10.0	268	85.6
P_4	0.2	2.0	268	85.6
P_5	0.2	5.0	267	85.3
P_6	0.2	10.0	266	85.0
P_7	0.5	2.0	267	85.3
P_8	0.5	5.0	268	85.6
P_9	0.5	10.0	265	84.7

Figure 5.1 show the distribution function of these parameter sets in terms of the CPU time.

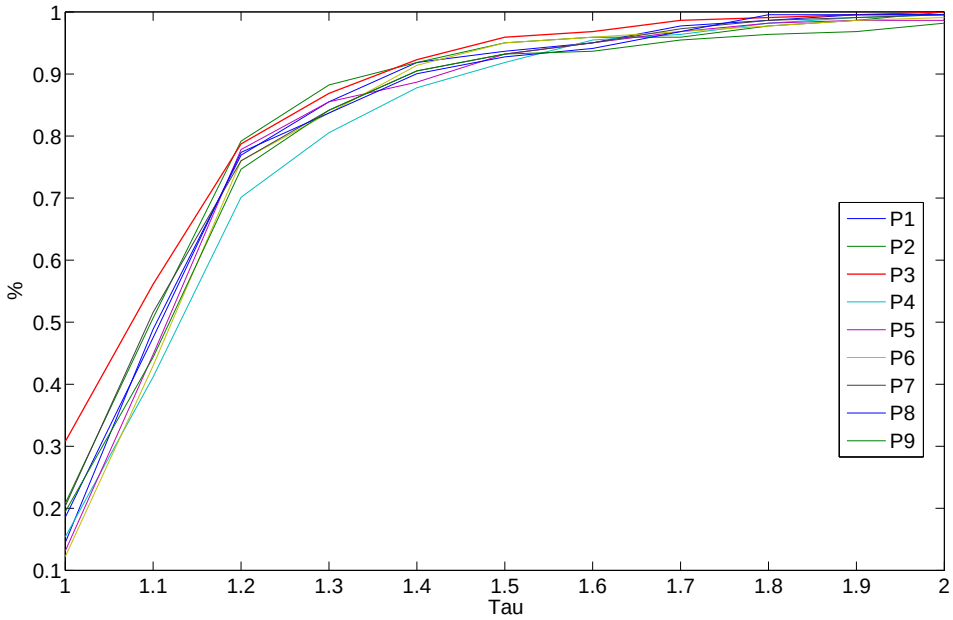

Figure 5.1: Comparison of (γ_1, γ_2).

We see that the parameter set P_3 is the best. Therefore, we use $\gamma_1 = 0.1$ and $\gamma_2 = 10.0$ in the subsequent experimentations.

Next, we focus on the number of memories m. In the original L-BFGS method, we usually set m in $3 \leq m \leq 7$ [11]. Then, we compare $m = 3,\ 5,\ 7$. The other parameters are as follows.

$$\gamma_1 = 0.1,\ \gamma_2 = 10.0,\ \eta_1 = 0.01,\ \eta_2 = 0.9,\ \mu_{\min} = 1.0 \times 10^{-3},\ M = 8.$$

Table 5.2 shows the number of successes and the rate of success for all 313 test problems.

Table 5.2: The number of successes and the rate of success at each m.

m	The number of successes	The rate of success (%)
3	267	85.3
5	268	85.6
7	266	85.0

Figure 5.2 shows the distribution function of the number of memories in terms of the CPU time.

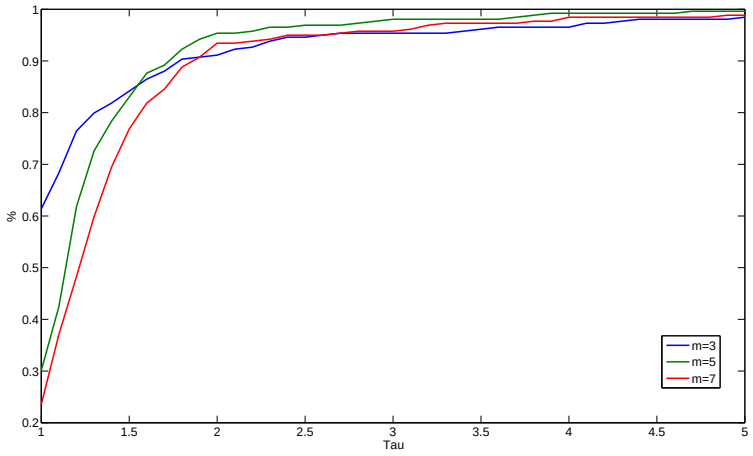

Figure 5.2: Comparison of $m = 3, 5, 7$.

We see that $m = 5$ is the best. Therefore, we use $m = 5$ in the subsequent experimentations.

Next, we focus on the nonmonotone parameter M. We compare $M = 4,\ 6,\ 8,\ 10,\ 12$ and M which coresponds to a usual monotone decreasing case. The other parameters are given as,

$$\gamma_1 = 0.1,\ \gamma_2 = 10.0,\ \eta_1 = 0.01,\ \eta_2 = 0.9,\ \mu_{\min} = 1.0 \times 10^{-3},\ m = 5.$$

Table 5.3 shows the number of successes and the rate of success for all 313 test problems.

Table 5.3: The number of successes and the rate of success at each M.

M	The number of successes	The rate of success (%)
Monotone (M=0)	225	71.9
4	263	84.0
6	265	84.7
8	265	84.7
10	268	85.6
12	268	85.6

Figure 5.3 shows the distribution function of the nonmonotone parameter in terms of the CPU time.

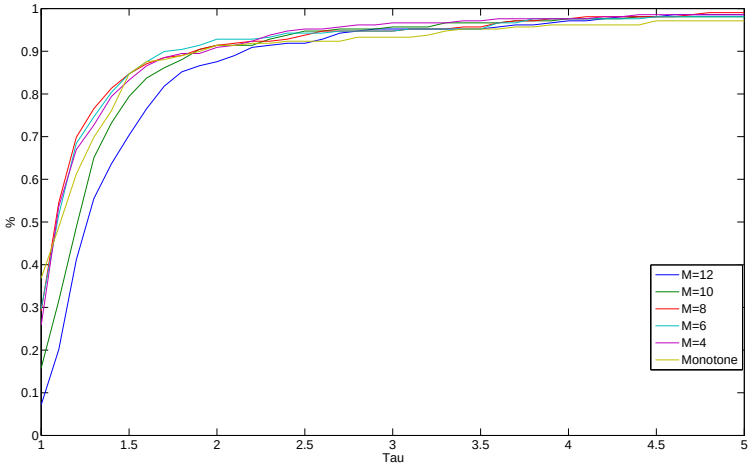

Figure 5.3: Comparison of $M = 4,\ 6,\ 8,\ 10,\ 12$ and monotone decreasing ($M = 0$).

We see that $M = 8$ is the best. Therefore, we use $M = 8$ at subsequent experimentations.

Finally, we compare the algorithm using the rejected information (Reject) and the basic one (Basic) introduced in section 4. The other parameters are as follows.

$$\gamma_1 = 0.1,\ \gamma_2 = 10.0,\ \eta_1 = 0.01,\ \eta_2 = 0.9,\ \mu_{\min} = 1.0 \times 10^{-3},\ m = 5,\ M = 8$$

Table 5.4 shows the number of successes and the rate of success for all 313 test problems.

Table 5.4: The number of successes and the rate of success of Reject and Basic.

Algorithm	The number of successes	The rate of success (%)
Reject	268	85.6
Basic	270	86.3

Figure 5.4 shows the distribution function of these two algorithms in terms of the CPU time.

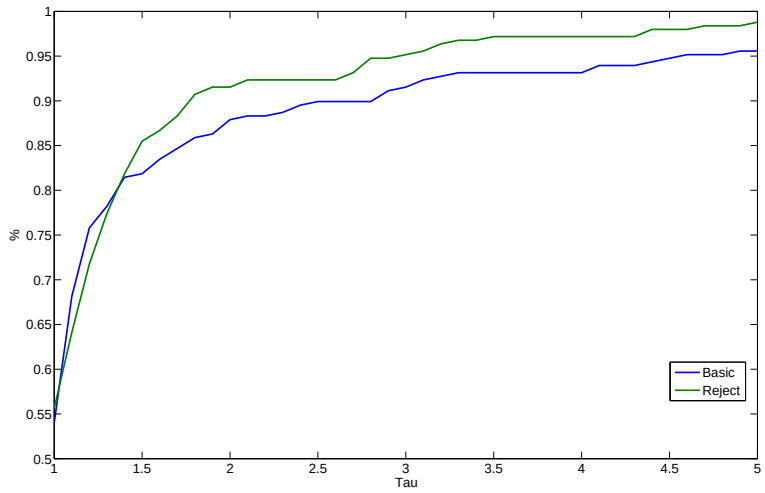

Figure 5.4: Comparison of Reject and Basic.

We see that the algorithm using the rejected information is better than the other. Therefore, we use the algorithm using the rejected information when we compare the regularized L-BFGS method and the L-BFGS method.

5.2 Comparisons of the regularized L-BFGS method and the L-BFGS method

We compare the regularized L-BFGS method (RLBFGS) and the L-BFGS method (LBFGS) in terms of the number of function evaluations and the CPU time.

Table 5.5 shows the number of successes and the rate of success for all 313 test problems. Figures 5.5 and 5.6 show the results for $\mathcal{P}_{\mathcal{S}}$, respectively. Furthermore, Figure 5.7 and 5.8 show the results only for the large scale problems. Here, we define the large scale problems as the one whose dimensions of their decision variables is over 1000. There are 147 large scale test problems. ${\mathcal{P}_{\mathcal{S}}}^{\text{large}}$ denotes the set which extract the large scale problems from $\mathcal{P}_{\mathcal{S}}$. Table 5.6 shows the number of successes and the rate of success for 147 large scale test problems. Moreover, we show the details of all results in Table A.1 in appendix.

The rate of success of Table 5.5 and 5.6 show that the regularized L-BFGS method can solve 85% for each case. However, the L-BFGS method can solve 71.9% of all test problems and 68.7% of large scale test problems. Therefore, we can argue that the regularized L-BFGS method can solve more problems than the L-BFGS method.

Figure 5.5 shows that the regularized L-BFGS method needs less number of function evaluations than that of the L-BFGS method. Furthermore, Figure 5.7 shows that the results for large scale test problems have a little improvement than that of the all test problems. Figure 5.6 shows that the regularized L-BFGS method is slightly fast than the L-BFGS method. However, Figure 5.8 shows that the results for large scale test problems do not has such improvement. In terms of that, since some large scale problems are easy to solve comparatively, we can consider that the line search technique of the L-BFGS method has a nice performance for these easy problems.

Table 5.5: The number of successes and the rate of success of RLBFGS and LBFGS for all 313 problems.

Algorithm	The number of successes	The rate of success (%)
RLBFGS	266	85.0
LBFGS	225	71.9

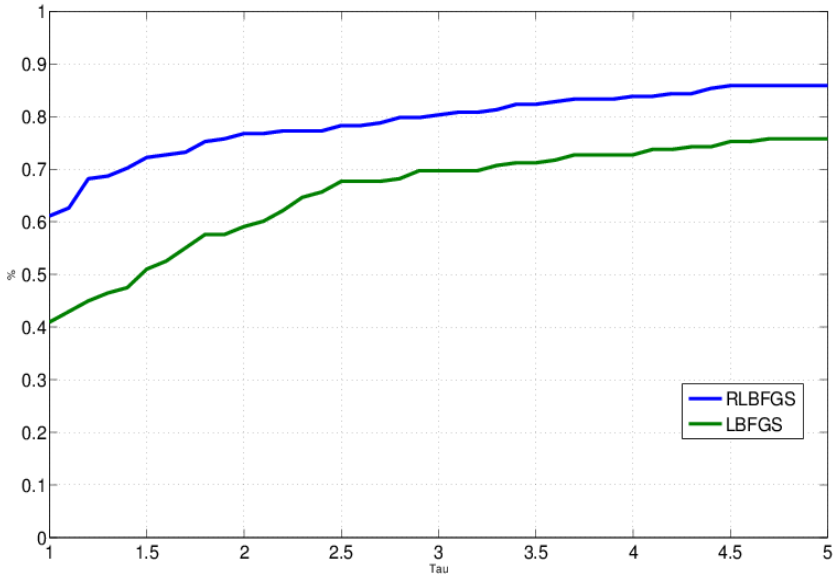

Figure 5.5: Comparison of RLBFGS and LBFGS in terms of n_f for $\mathcal{P}_\mathcal{S}$.

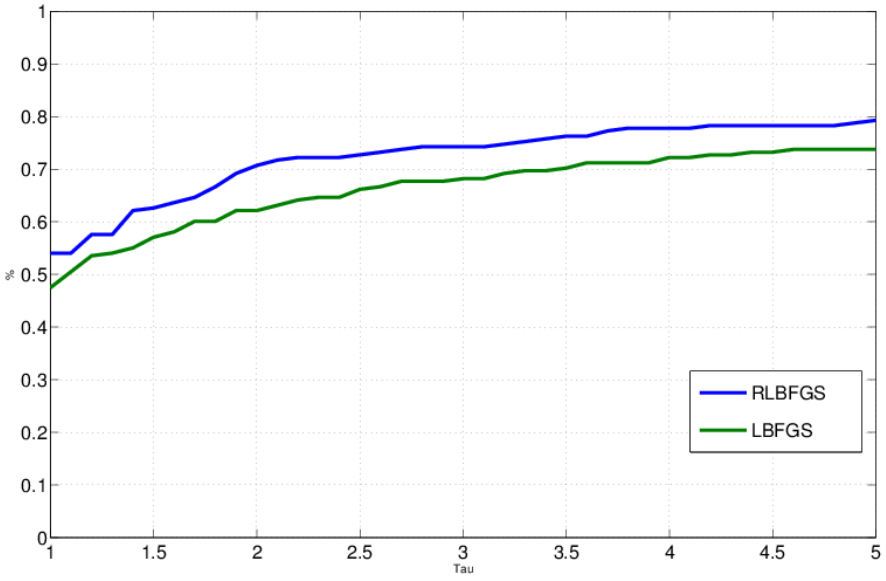

Figure 5.6: Comparison of RLBFGS and LBFGS in terms of solved time for $\mathcal{P}_\mathcal{S}$.

Table 5.6: The number of successes and the rate of success of RLBFGS and LBFGS for 147 large scale problems.

Algorithm	The number of successes	The rate of success (%)
RLBFGS	125	85.0
LBFGS	101	68.7

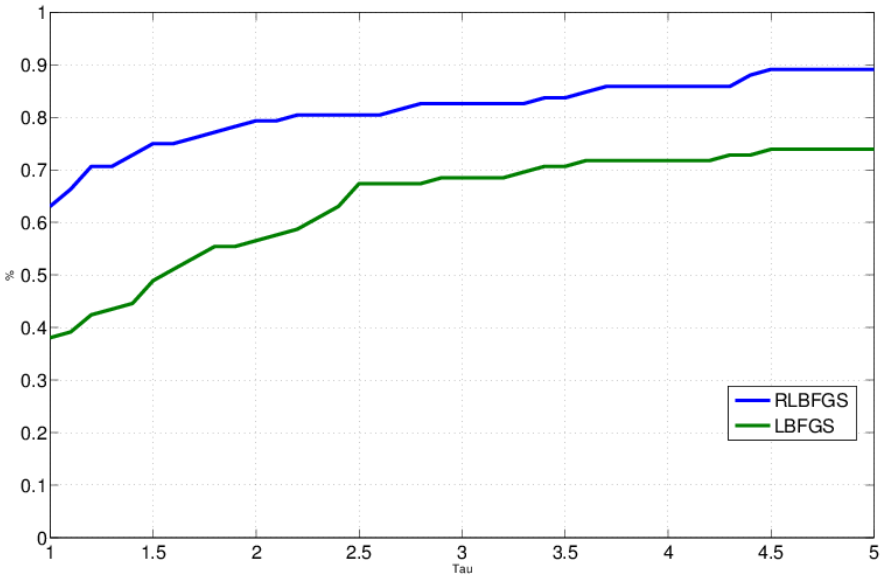

Figure 5.7: Comparison of RLBFGS and LBFGS in terms of n_f for $\mathcal{P}_{\mathcal{S}}^{\text{large}}$.

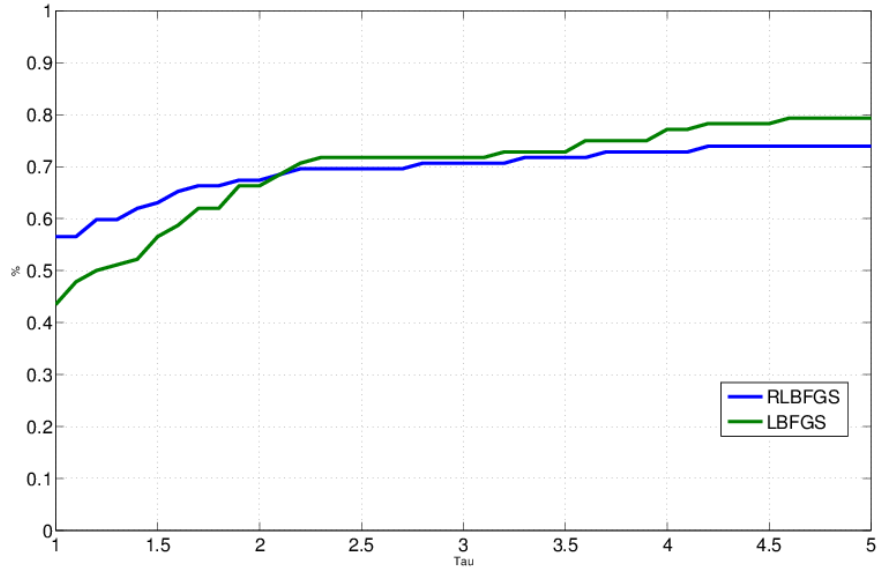

Figure 5.8: Comparison of RLBFGS and LBFGS in terms of solved time for $\mathcal{P}_{\mathcal{S}}^{\text{large}}$.

6 Conclusion

In this paper, we proposed the regularized L-BFGS method for unconstrained minimization problems and we showed the global convergence of the regularized L-BFGS method under

some appropriate assumptions. From the numerical results, we see that the regularized L-BFGS method can solve more test problems that the original L-BFGS. This result indicates that the proposed method is robust.

For future work, we may consider to solve the following boxed constrained optimization problems.

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad l \leq x \leq u, \end{aligned}$$

where l and u are vectors such that $l_i \in [-\infty, \infty)$, $u_i \in (-\infty, \infty]$ and $l_i < u_i$ for all i. It is known that the boxed constrained optimization problems play an important role in the development of algorithms for the general constrained problems.

Acknowledgements

First of all, I would like to express sincere appreciation to Associate Professor Nobuo Yamashita. Although I sometimes troubled him due to my greenness, he always kindly looked after me and gave me plenty of precise advice. It is an honor to have studied under him. I would like to thank all members of Yamashita Laboratory, my friends and my family for their encouraging words.

References

[1] J. Barazilai, J. M. Borwein, *Two-Point Step Size Gradient Methods*, IMA Journal of Numerical Analysis 1988.

[2] J. V. Burke and A. Wiegmann, *Notes on limited memory BFGS updating in a trust-region framework*, SIAM Journal on Optimization, July 2, 1996.

[3] J. V. Burke, A. Wiegmann and L. Xu, *Limited memory BFGS updating in a trust-region framework*, 2008.

[4] R. H. Byrd, J. Nocedal and R. B. Schnabel, *Representations of quasi-newton matrices and their use in limited memory methods*, Technical Report of Northwestern university NAM–03, January 21, 1996.

[5] J. E. Dennis Jr., and J. J. Mor, *Quasi-Newton methods, motivation and theory*, SIAM review 19.1 (1977), pp. 46–89

[6] E. D. Dolan and J. J. More', *Benchmarking optimization software with performance profiles*, Mathematical Programming 91 (2002), pp. 201–213.

[7] N. I. M. Gould, D. Orban and P. L. Toint, *CUTEr and SifDec, a constrained and unconstrained testing environment, revisited,* ACM Transactions on Mathematical Software, 29 (2003), pp. 373–394.

[8] L. Grippo, F. Lampariello and S. Lucidi, *A nonmonotone line search technique for newton's method*, SIAM Jarnal Numerical Analisis, Vol. 23, No. 4, 1986.

[9] D. C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization*, Mathematical programming 45. 1-3 (1989), pp. 503–528

[10] J. J. Mor and C. S. Danny, *Computing a trust region step*, SIAM Journal on Scientific and Statistical Computing 4.3 (1983), pp. 553–572

[11] J. Nocedal, *Updating quasi-Newton matrices with limited storage*, Mathematics of computation 35.151 (1980), pp. 773–782

[12] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer series in operations research, New York, 1999.

[13] J. Nocedal *Software for Large-scale Unconstrained Optimization: L-BFGS destribution*, Available at:http://www.ece.northwestern.edu/ nocedal/lbfgs.html, Jan. 2014.

[14] J. D. Pearson, *Variable metric methods of minimisation*, The Computer Journal 12.2 (1969), pp. 171–178.

[15] M. J. D. Powell, *Some global convergence properties of a variable metric algorithm for minimization without exact line search*, in: R. W. Cottle and C. E. Lemke eds., *Nonlinear Programming*, SIAM-AMS Proceedings IX (SIAM Publications, 1976).

[16] M.Raydan, *The Barzilai and Borwein gradient method for large scale unconstrained minimization problem*, SIAM Jarnal Optimization Vol. 7, No. 1, pp. 26–33, February 1997.

[17] D. F. Shanno, P. A. Kang-Hoh, *Matrix conditioning and nonlinear optimization*, Mathematical Programming 14 (1978), pp. 149–160.

[18] W. Sun, *Nonmonotone trust region method for solving optimization problems*, Applied Mathematics and Computation, 156 (2004), pp. 159–174.

[19] K. Ueda, N. Yamashita, *Convergence Properties of the Regularized Newton Method for the Unconstrained Nonconvex Optimization*, Applied Mathematics and Optimization, 62(2010), pp. 27–46.

[20] K. Ueda, N. Yamashita, *A regularized Newton method without line search for unconstrained optimization*, Technical Report Kyoto University, 2009.

[21] K. Ueda, N. Yamashita, *Studies on Regularized Newton-type methods for unconstrained minimization problems and their global complexity bounds*, Doctoral thesis Kyoto University, March 2012.

A Details of the numerical experiments

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
3PK	30	8.00E+00	3.66E-01	0.00174	214	-	-	-	-
AIRCRFTB	8	2.36E-09	4.02E-04	0.00019	28	6.96E-13	1.64E-06	0.00016	48
AKIVA	2	6.17E+00	1.23E-04	0.0002	18	6.17E+00	2.54E-06	0.0002	22
ALLINIT	4	5.74E+00	5.05E-04	0.00016	16	5.74E+00	2.96E-07	0.00009	14
ALLINITU	4	5.74E+00	5.05E-04	0.0001	16	5.74E+00	2.96E-07	0.00005	14
ARGLINA	200	2.00E+02	2.58E-04	0.00362	7	2.00E+02	1.38E-13	0.00324	4
ARGLINB	200	9.96E+01	5.90E-04	0.01055	24	-	-	-	-
ARGLINC	200	1.01E+02	2.07E-04	0.01119	26	-	-	-	-
ARWHEAD	5000	3.81E-09	2.72E-03	0.01697	15	1.22E-11	1.36E-04	0.0199	14
BARD	3	8.21E-03	3.99E-04	0.00011	22	8.21E-03	7.98E-06	0.00008	23
BDEXP	5000	8.56E-01	1.66E-03	0.0141	12	8.52E-03	2.28E-05	0.01515	16
BDQRTIC	5000	2.00E+04	6.70E-03	0.27811	188	-	-	-	-
BEALE	2	1.58E-06	1.97E-03	0.00009	15	1.54E-11	7.90E-06	0.00005	15
BIGGS3	6	1.98E-03	5.28E-03	0.00029	34	1.33E-07	4.99E-05	0.00073	97
BIGGS5	6	1.98E-03	5.28E-03	0.00033	34	1.33E-07	4.99E-05	0.00054	97
BIGGS6	6	6.12E-03	3.89E-03	0.00028	52	5.66E-03	2.50E-05	0.00033	46
BIGGSB1	5000	1.33E-02	1.48E-03	0.17365	174	1.01E-03	1.87E-05	2.05008	2830
BOX	10000	-1.86E+03	1.76E-02	0.08024	27	-	-	-	-
BOX2	3	8.15E-07	4.93E-04	0.0001	12	-1.86E+03	5.67E-06	0.02899	12
BOX3	3	8.15E-07	4.93E-04	0.00009	12	7.16E-07	3.60E-05	0.0001	12
BOXPOWER	20000	4.71E-02	8.07E-02	0.09806	26	7.16E-07	3.60E-05	0.00006	12
BQP1VAR	1	-2.50E-01	9.66E-05	0.00008	7	5.97E-06	2.82E-04	0.13415	51
BQPGABIM	50	-1.84E-04	2.22E-04	0.00071	22	-2.50E-01	0.00E+00	0.00005	3
BQPGASIM	50	-1.84E-04	2.22E-04	0.00046	22	-1.84E-04	2.85E-06	0.00061	46
BQPGAUSS	2003	-1.32E+58	4.11E+30	4.80107	8595	-1.84E-04	2.85E-06	0.00061	46
BRATU1D	5003	-	-	-	-	-	-	-	-
BRKMCC	2	1.69E-01	1.59E-04	0.00007	11	-	-	-	-
BROWNAL	200	3.16E-07	1.11E-03	0.00236	9	1.69E-01	1.04E-08	0.00003	8
BROWNBS	2	1.29E-10	1.24E+01	0.00018	137	1.47E-09	2.77E-06	0.00339	13

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
BROWNDEN	4	8.58E+04	1.77E-04	0.00015	32	1.72E-10	9.13E+00	0.00004	25
BROYDN7D	5000	1.98E+03	1.51E-02	3.67027	1594	-	-	-	-
BRYBND	5000	4.61E-08	2.63E-04	0.0902	52	1.99E+03	7.71E-05	3.19344	1607
CAMEL6	2	-1.03E+00	1.38E-04	0.00016	31	6.57E-13	9.74E-07	0.04952	32
CHAINWOO	4000	-	-	-	-	-1.03E+00	1.88E-08	0.0001	22
CHARDIS0	2000	8.35E-15	3.29E-06	0.41465	10	-	-	-	-
CHEBYQAD	100	-	-	-	-	1.28E-20	8.74E-09	0.21511	5
CHENHARK	5000	-9.69E+07	3.95E+02	0.1674	157	-	-	-	-
CHNROSNB	50	1.84E-06	2.50E-03	0.00288	209	-6.73E+09	1.48E+02	0.85605	1161
CHNRSNBM	50	1.66E-07	2.86E-03	0.00653	340	3.85E-11	2.24E-05	0.00312	306
CLIFF	2	-	-	-	-	2.50E-11	3.02E-05	0.00435	382
CLPLATEA	5041	-1.37E-01	4.04E-02	0.88618	570	2.00E-01	8.66E-06	0.00006	41
CLPLATEB	5041	-1.74E-01	1.71E-02	0.45593	292	-4.76E+04	6.12E+01	3.97555	3030
CLPLATEC	5041	-	-	-	-	-4.39E+04	6.57E+01	2.37146	1804
COSINE	10000	-1.00E+04	8.93E-02	0.1027	36	-	-	-	-
CRAGGLVY	5000	1.69E+03	1.06E-02	0.09785	60	-1.00E+04	1.01E-03	0.03025	17
CUBE	2	3.19E-11	3.13E-04	0.00009	39	1.69E+03	8.75E-05	0.11141	88
CURLY10	10000	-	-	-	-	1.70E-19	3.22E-09	0.00006	50
CURLY20	10000	-	-	-	-	-	-	-	-
CURLY30	10000	-	-	-	-	-	-	-	-
CVXBQP1	10000	1.51E-07	6.45E-05	8.58676	4124	-	-	-	-
DECONVB	63	1.82E-05	1.67E-03	0.00108	33	-	-	-	-
DECONVU	63	1.82E-05	1.67E-03	0.00105	33	6.67E-08	1.71E-05	0.00592	212
DEGDIAG	100001	3.27E-08	8.09E-07	0.11842	8	6.67E-08	1.71E-05	0.00526	212
DEGTRID	100001	-1.00E+05	1.12E-01	0.13004	7	1.73E-18	5.88E-12	0.03107	6
DEGTRID2	100001	-1.00E+05	1.12E-01	0.11739	7	-1.00E+05	1.93E-03	0.22282	17
DENSCHNA	2	9.35E-10	8.36E-05	0.00007	12	-1.00E+05	1.93E-03	0.22466	17
DENSCHNB	2	8.97E-08	5.08E-04	0.00006	10	8.22E-14	2.79E-07	0.00003	11
DENSCHNC	2	9.05E-09	6.63E-04	0.00007	14	5.12E-14	6.39E-07	0.00002	9
DENSCHND	3	3.34E-06	5.49E-04	0.00013	41	8.49E-13	2.61E-06	0.00005	17
DENSCHNE	3	-	-	-	-	1.18E-08	5.15E-06	0.0001	56

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
DENSCHNF	2	8.73E-10	6.15E-04	0.00008	9	9.72E-12	6.21E-06	0.00006	42
DIXMAANA	3000	1.00E+00	1.38E-06	0.00773	12	6.81E-17	1.45E-07	0.00003	10
DIXMAANB	3000	1.00E+00	2.12E-05	0.0077	12	1.00E+00	6.86E-10	0.00685	14
DIXMAANC	3000	1.00E+00	3.16E-05	0.00838	13	1.00E+00	8.02E-08	0.0063	13
DIXMAAND	3000	1.00E+00	2.63E-05	0.00975	15	1.00E+00	5.95E-07	0.01004	14
DIXMAANE	3000	1.00E+00	1.63E-04	0.09637	133	1.00E+00	2.38E-06	0.0082	16
DIXMAANF	3000	1.00E+00	3.04E-04	0.06428	91	1.00E+00	2.08E-06	0.14614	260
DIXMAANG	3000	1.00E+00	2.15E-04	0.07365	104	1.00E+00	1.72E-06	0.13563	241
DIXMAANH	3000	1.00E+00	2.27E-04	0.07077	99	1.00E+00	1.81E-06	0.13136	236
DIXMAANI	3000	1.01E+00	5.95E-04	0.07641	107	1.00E+00	3.15E-06	0.12877	229
DIXMAANJ	3000	1.00E+00	2.01E-04	0.05937	67	1.00E+00	6.08E-06	1.14051	2067
DIXMAANK	3000	1.00E+00	9.88E-05	0.04432	61	1.00E+00	1.53E-06	0.20056	359
DIXMAANL	3000	1.00E+00	2.20E-04	0.05187	58	1.00E+00	1.78E-06	0.18627	341
DIXMAANM	15	1.00E+00	5.47E-04	0.00028	52	1.00E+00	1.99E-06	0.30661	564
DIXMAANN	15	1.00E+00	4.27E-04	0.00024	38	1.00E+00	3.43E-06	0.00026	67
DIXMAANO	15	1.00E+00	5.50E-04	0.00042	32	1.00E+00	6.22E-06	0.00031	62
DIXMAANP	15	1.00E+00	6.18E-04	0.00022	38	1.00E+00	3.37E-06	0.00026	59
DIXON3DQ	10000	1.44E-01	3.89E-02	0.06282	32	1.00E+00	5.10E-06	0.00023	57
DJTL	2	7.16E+10	1.04E+12	0.00013	51	3.10E-03	7.99E-05	2.51798	1738
DQDRTIC	5000	6.77E-12	3.27E-05	0.02234	19	-	-	-	-
DQRTIC	5000	3.11E+03	3.82E+00	0.02973	36	5.59E-14	4.83E-06	0.0158	20
DRCAV1LQ	4489	0.00E+00	0.00E+00	0.00205	1	3.01E+00	3.60E-02	0.02827	44
DRCAV2LQ	4489	0.00E+00	0.00E+00	0.00221	1	-	-	-	-
DRCAV3LQ	4489	0.00E+00	0.00E+00	0.00178	1	-	-	-	-
EDENSCH	2000	1.20E+04	8.07E-03	0.01247	22	-	-	-	-
EG1	3	-	-	-	-	1.20E+04	1.64E-04	0.01494	32
EG2	1000	-9.99E+02	1.30E-05	0.0011	5	-	-	-	-
EIGENALS	2550	1.97E+01	2.75E-01	6.08952	913	-9.99E+02	1.46E-06	0.00095	5
EIGENBLS	2550	1.34E-02	2.30E-03	5.06106	766	1.01E-04	7.28E-04	51.13009	7982
EIGENCLS	2652	1.23E-02	1.95E-02	24.09402	3368	1.28E-05	1.40E-05	43.73067	6813
ENGVAL1	5000	5.55E+03	1.31E-02	0.01746	15	4.32E-06	1.88E-04	55.20801	7975

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
ENGVAL2	3	1.15E-07	6.83E-04	0.00012	44	5.55E+03	1.82E-04	0.01658	18
ERRINROS	50	3.99E+01	1.93E-02	0.00184	120	5.18E-16	1.60E-06	0.00007	36
ERRINRSM	50	3.78E+01	2.88E-02	0.00277	210	3.99E+01	1.87E-04	0.00151	155
EXPFIT	2	2.41E-01	2.40E-06	0.0001	21	3.77E+01	5.39E-04	0.00387	298
EXPLIN	1200	-	-	-	-	2.41E-01	2.63E-06	0.00005	16
EXPLIN2	1200	-	-	-	-	-	-	-	-
EXPQUAD	1200	-	-	-	-	-	-	-	-
EXTROSNB	1000	2.01E-05	1.44E-03	0.08342	369	-	-	-	-
FLETBV3M	5000	-1.22E+05	7.43E-01	0.01186	8	8.10E-09	1.11E-05	0.69011	4781
FLETCBV2	5000	-5.00E-01	8.00E-08	0.00233	2	-2.49E+05	2.71E-02	0.06788	45
FLETCBV3	5000	-1.71E+06	7.50E-01	0.01074	8	-5.00E-01	1.77E-04	0.00292	3
FLETCHBV	5000	-2.51E+23	1.77E+10	0.16603	95	-4.12E+13	4.41E+02	0.02076	22
FLETCHCR	1000	8.20E-07	1.40E-02	1.37813	5491	-8.53E+24	2.70E+09	3.01659	1984
FMINSRF2	5625	1.01E+00	2.30E-03	0.32893	203	1.19E-10	1.05E-04	1.02225	5672
FMINSURF	5625	1.00E+00	9.53E-05	0.55925	338	1.00E+00	2.08E-05	0.36837	283
FREUROTH	5000	6.08E+05	5.59E-02	0.07026	48	1.00E+00	5.04E-07	0.83717	632
GENHUMPS	5000	1.96E+04	4.29E+01	0.45759	230	-	-	-	-
GENROSE	500	1.00E+00	1.48E-03	0.15877	1202	-	-	-	-
GRIDGENA	6218	2.35E+04	1.75E-02	0.09696	47	1.00E+00	7.01E-05	0.12015	1248
GROWTHLS	3	-	-	-	-	2.35E+04	2.38E-04	2.55836	1496
GULF	3	-	-	-	-	1.00E+00	7.11E-06	0.00065	213
HADAMALS	400	7.54E+03	1.48E-04	0.0036	16	4.00E-03	5.02E-04	0.00146	34
HAIRY	2	2.00E+01	7.40E-05	0.0003	121	7.54E+03	1.09E-06	0.00381	19
HARKERP2	1000	-	-	-	-	2.00E+01	8.05E-07	0.00013	92
HART6	6	-3.32E+00	1.18E-04	0.00015	16	-	-	-	-
HATFLDA	4	-	-	-	-	-3.32E+00	7.43E-07	0.00014	25
HATFLDB	4	-	-	-	-	-	-	-	-
HATFLDC	25	9.03E-07	1.76E-03	0.00026	15	-	-	-	-
HATFLDD	3	1.93E-04	1.61E-03	0.00009	14	1.39E-10	1.19E-05	0.00024	23
HATFLDE	3	8.79E-06	3.79E-04	0.00012	14	6.62E-08	9.43E-08	0.00012	24
HATFLDFL	3	6.66E-05	1.08E-04	0.00007	10	5.12E-07	1.65E-05	0.00017	41

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
HEART6LS	6	1.12E-01	1.61E-02	0.00621	2149	6.66E-05	4.28E-06	0.00003	10
HEART8LS	8	1.11E-06	1.14E-03	0.00158	278	-	-	-	-
HELIX	3	7.34E-10	7.16E-04	0.0001	32	8.13E-11	2.04E-05	0.00213	737
HIELOW	3	-	-	-	-	1.72E-13	6.50E-06	0.00006	31
HILBERTA	2	4.31E-07	4.56E-04	0.00007	12	-	-	-	-
HILBERTB	10	6.58E-10	6.45E-05	0.0001	8	2.76E-18	2.32E-09	0.00003	7
HIMMELBB	2	7.78E-07	8.22E-04	0.00007	19	1.97E-18	3.83E-09	0.00005	7
HIMMELBF	4	3.20E+02	7.73E-01	0.00011	29	3.75E-10	6.06E-06	0.00003	20
HIMMELBG	2	5.19E-09	1.88E-04	0.00007	12	3.20E+02	1.13E-02	0.00006	26
HIMMELBH	2	-1.00E+00	3.79E-04	0.00008	10	1.13E-15	1.12E-07	0.00003	13
HIMMELP1	2	-	-	-	-	-1.00E+00	9.16E-06	0.00004	6
HOLMES	180	1.17E+03	3.52E-02	5.58179	3034	-	-	-	-
HS1	2	1.58E-11	1.36E-04	0.00011	18	-	-	-	-
HS110	10	-4.58E+01	9.39E-04	0.00014	8	4.54E-14	3.54E-07	0.0001	51
HS2	2	1.58E-11	1.36E-04	0.00009	18	-4.58E+01	4.80E-06	0.00008	6
HS25	3	3.28E+01	1.99E-08	0.01293	438	4.54E-14	3.54E-07	0.00013	51
HS3	2	-1.09E+03	9.78E-01	0.00011	8	-	-	-	-
HS38	4	2.66E-11	6.76E-05	0.0004	162	-9.87E+04	1.56E+00	0.00006	10
HS3MOD	2	-3.01E+04	2.02E+01	0.00009	27	6.67E-15	2.85E-06	0.00022	120
HS4	2	-	-	-	-	-1.84E+10	2.62E+05	0.00009	27
HS45	5	-	-	-	-	-	-	-	-
HS5	2	-1.91E+00	1.60E-04	0.00009	9	-	-	-	-
HUMPS	2	9.30E-01	1.80E+01	0.00029	244	-1.91E+00	7.48E-07	0.00005	7
HYDC20LS	99	-	-	-	-	2.88E-12	7.50E-07	0.00018	201
INDEF	5000	-2.02E+07	4.22E+01	0.00929	8	-	-	-	-
INDEFM	100000	-9.89E+06	2.75E+03	3.86007	112	-	-	-	-
JENSMP	2	-	-	-	-	-9.65E+06	1.51E-01	16.29256	577
JIMACK	3549	8.68E-01	2.50E-03	9.54618	222	1.24E+02	1.86E-07	0.00012	51
JNLBRNG1	10000	-1.14E+00	1.80E-03	0.50106	139	8.67E-01	2.56E-05	43.82026	923
JNLBRNG2	10000	-1.04E+02	3.74E-02	0.68116	190	-1.15E+00	1.21E-05	1.08004	338
JNLBRNGA	10000	-2.15E+00	1.72E-03	0.35089	106	-1.06E+02	2.89E-04	1.45931	454

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
JNLBRNGB	10000	-8.02E+02	2.15E-01	0.74081	223	-2.18E+00	2.53E-05	1.05839	360
KOEBHELB	3	1.12E+02	3.99E-04	0.0009	37	-9.55E+02	2.07E-03	9.36018	2994
KOWOSB	4	3.90E-04	2.63E-04	0.0001	22	1.12E+02	7.17E-05	0.00085	36
LIARWHD	5000	1.38E-07	9.91E-04	0.03516	30	3.08E-04	3.15E-06	0.00017	46
LINVERSE	1999	6.81E+02	6.09E-03	0.03066	48	2.57E-12	4.33E-04	0.02644	26
LMINSURF	5625	1.01E+00	2.35E-03	0.33002	205	6.81E+02	3.23E-05	0.11835	213
LOGHAIRY	2	6.55E+00	1.63E-03	0.00007	2	1.00E+00	1.68E-05	0.35844	259
LOGROS	2	1.30E-12	1.03E-05	0.00052	396	6.55E+00	1.07E-03	0.00004	3
MANCINO	100	6.70E-11	1.01E-02	0.03761	15	2.22E-16	1.85E-06	0.00028	194
MARATOSB	2	9.92E-01	1.22E-01	0.00014	120	1.81E-14	9.89E-05	0.03917	14
MAXLIKA	8	-	-	-	-	-1.00E+00	4.96E-07	0.00127	1563
MCCORMCK	5000	-9.86E+05	2.19E+01	0.53445	338	-	-	-	-
MDHOLE	2	9.09E+02	6.02E+02	0.00008	25	-9.50E+09	7.31E+01	14.05179	8857
MEXHAT	2	-4.00E-02	2.75E-04	0.00011	66	-	-	-	-
MEYER3	3	8.79E+01	3.35E+00	0.00329	1189	-4.00E-02	4.93E-06	0.00007	53
MINSURF	64	1.00E+00	0.00E+00	0.00009	1	-	-	-	-
MINSURFO	5306	1.01E+00	6.38E-03	0.22412	105	-	-	-	-
MODBEALE	20000	3.04E+00	1.02E-01	5.26924	525	1.00E+00	3.17E-05	0.27562	144
MOREBV	5000	1.59E-07	7.89E-04	0.00147	2	-	-	-	-
MSQRTALS	1024	8.30E-03	2.62E-03	1.60735	1268	5.44E-09	6.54E-05	0.01083	11
MSQRTBLS	1024	2.14E-02	2.93E-03	2.30258	1799	2.81E-07	2.33E-05	3.37451	2215
NCB20	5010	-1.40E+02	3.23E-02	0.56567	128	2.99E-07	2.29E-05	2.98503	1874
NCB20B	5000	7.35E+03	3.69E-03	0.44351	99	-1.22E+03	2.22E-04	1.76569	352
NCVXBQP1	10000	-0.1016733+201	0.1271913+102	0.02933	15	7.35E+03	2.37E-05	7.85243	1726
NCVXBQP2	10000	-0.1371227+198	0.4503556+100	0.02969	15	-	-	-	-
NCVXBQP3	10000	-0.1594651+184	1.11E+93	0.03142	15	-	-	-	-
NLMSURF	5625	8.12E+01	2.16E-02	0.00237	2	-	-	-	-
NOBNDTOR	5476	-1.93E+02	8.43E-02	0.29342	165	1.00E+00	6.61E-05	0.44368	316
NONCVXU2	5000	1.19E+04	1.13E+00	0.47919	304	-2.58E+06	3.30E+01	0.35345	223
NONCVXUN	5000	1.19E+04	7.85E-01	0.53718	340	1.16E+04	2.87E-03	2.10216	1509
NONDIA	5000	2.58E-04	2.50E-04	0.00951	11	1.16E+04	3.81E-03	3.47439	2480

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
NONDQUAR	5000	1.31E-02	2.77E-02	0.03675	38	4.59E-19	8.53E-07	0.01807	23
NONMSQRT	4900	-	-	-	-	1.57E-04	3.62E-04	0.16839	224
NONSCOMP	5000	5.16E-05	3.19E-02	0.03218	31	-	-	-	-
OBSTCLAE	10000	-9.80E-01	1.02E-04	0.00509	2	1.72E-08	2.17E-04	0.03275	35
OBSTCLAL	10000	-4.11E-01	2.55E-03	0.38141	97	-1.35E+04	1.08E+00	0.43394	140
OBSTCLBL	10000	-2.59E+00	1.70E-02	0.97007	290	-8.97E+03	1.03E+00	1.61029	428
OBSTCLBM	10000	-4.85E+00	3.21E-02	0.76942	232	-1.82E+04	1.36E+00	1.28086	358
OBSTCLBU	10000	-2.76E-01	1.62E-03	0.15624	46	-8.43E+03	6.76E-01	1.29224	358
ODC	5184	-1.10E+01	1.16E-01	0.31225	133	-3.39E+03	1.19E-01	1.68862	451
ODNAMUR	11130	8.80E+03	9.78E-02	16.42038	6382	-1.46E+05	7.91E+00	0.27307	118
OSBORNEA	5	7.72E-05	6.58E-04	0.00038	51	-	-	-	-
OSBORNEB	11	4.06E-02	4.10E-03	0.00217	91	5.46E-05	1.70E-05	0.0012	176
OSCIGRAD	100000	2.52E-10	5.61E-02	1.50511	60	4.01E-02	4.80E-05	0.00589	198
OSCIPATH	10	1.00E+00	1.43E-03	0.00007	9	6.88E-14	1.22E-03	1.46534	64
OSLBQP	8	-3.00E+00	1.01E-03	0.0001	7	1.00E+00	1.40E-05	0.00006	10
PALMER1	4	1.18E+04	1.58E-02	0.00041	54	-3.00E+00	5.55E-16	0.00007	3
PALMER1A	6	1.29E+00	2.93E-02	0.00162	200	-	-	-	-
PALMER1B	4	3.46E+04	5.31E+02	0.00035	44	8.99E-02	7.61E-04	0.00472	604
PALMER1C	8	4.99E+00	9.21E-02	0.02596	6252	3.43E+04	5.35E-05	0.00044	56
PALMER1D	7	1.22E+00	1.46E-01	0.01112	2764	-	-	-	-
PALMER1E	8	4.59E-01	4.67E-02	0.0031	453	-	-	-	-
PALMER2	4	4.58E+03	1.34E-02	0.00033	41	-	-	-	-
PALMER2A	6	2.69E+03	2.15E+03	0.00028	28	4.58E+03	1.83E-04	0.0002	42
PALMER2B	4	6.23E-01	1.92E-03	0.00038	67	1.71E-02	4.67E-05	0.00213	490
PALMER2C	8	1.66E-01	3.30E-02	0.01005	2351	6.23E-01	1.01E-04	0.00032	59
PALMER2E	8	2.96E-02	1.11E-02	0.00183	330	-	-	-	-
PALMER3	4	2.16E+03	6.00E-02	0.00027	54	-	-	-	-
PALMER3A	6	3.55E-02	6.84E-03	0.00045	64	2.16E+03	2.16E-03	0.00033	55
PALMER3B	4	5.05E+02	2.87E+02	0.00007	6	2.04E-02	1.43E-04	0.00246	623
PALMER3C	8	2.02E-01	2.53E-02	0.00485	1465	4.23E+00	9.71E-06	0.0004	63
PALMER3E	8	3.07E-02	4.92E-03	0.00166	314	-	-	-	-

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
PALMER4	4	2.12E+03	1.06E-01	0.00028	31	-	-	-	-
PALMER4A	6	4.90E-02	4.43E-03	0.00083	184	2.12E+03	1.95E-03	0.00026	60
PALMER4B	4	6.84E+00	7.18E-03	0.00031	39	4.06E-02	1.52E-04	0.00101	249
PALMER4C	8	4.13E-01	4.44E-02	0.00663	1932	6.84E+00	1.13E-05	0.00031	60
PALMER4E	8	6.56E-02	4.93E-03	0.00211	390	-	-	-	-
PALMER5A	8	2.13E+00	9.76E-03	0.00021	19	-	-	-	-
PALMER5B	9	4.65E+00	8.18E-02	0.00123	305	2.13E+00	3.10E-04	0.00016	19
PALMER5C	6	2.13E+00	3.09E-02	0.00008	13	-	-	-	-
PALMER5D	4	8.73E+01	5.01E-02	0.00009	24	2.13E+00	2.49E-04	0.00005	14
PALMER5E	8	5.00E-01	3.67E-02	0.00015	22	8.73E+01	7.34E-04	0.00008	36
PALMER6A	6	1.83E+04	1.23E+06	0.00014	15	3.20E-02	2.42E-03	0.00402	1206
PALMER6C	8	9.73E-01	3.07E-02	0.00094	289	5.59E-02	2.75E-04	0.00247	787
PALMER6E	8	5.57E-02	9.85E-03	0.00119	280	-	-	-	-
PALMER7A	6	2.79E+01	4.83E-03	0.0004	29	-	-	-	-
PALMER7C	8	4.35E+00	9.31E-03	0.00234	880	1.05E+01	1.19E-02	0.01029	3229
PALMER7E	8	1.05E+01	1.42E-02	0.00209	538	-	-	-	-
PALMER8A	6	6.97E+00	1.06E-02	0.00016	30	1.02E+01	7.55E-04	0.00759	2034
PALMER8C	8	8.19E-01	2.89E-02	0.00333	1083	7.40E-02	9.17E-05	0.00069	260
PALMER8E	8	1.38E-01	1.44E-02	0.00083	185	-	-	-	-
PARKCH	15	-	-	-	-	-	-	-	-
PENALTY1	1000	9.73E-03	4.29E-05	0.00913	49	-	-	-	-
PENALTY2	200	4.71E+13	3.68E-03	0.01065	138	9.69E-03	2.74E-08	0.01098	79
PENALTY3	200	-	-	-	-	-	-	-	-
PENTDI	5000	-2.09E+02	3.30E-03	0.01081	11	-	-	-	-
PFIT1LS	3	-	-	-	-	-2.09E+02	1.72E-05	0.01054	13
PFIT2LS	3	5.46E+03	2.14E+05	0.00022	85	3.18E-14	9.06E-07	0.00055	276
PFIT3LS	3	-	-	-	-	1.10E-13	1.64E-05	0.00144	766
PFIT4LS	3	-	-	-	-	1.49E-13	3.19E-05	0.00123	719
POWELLBC	1000	1.11E+01	2.88E-03	0.89817	114	-	-	-	-
POWELLSG	5000	2.86E-05	8.36E-06	0.04354	43	3.09E+02	1.46E-01	2.32753	278
POWER	10000	1.25E-06	1.24E-04	0.68971	384	2.00E-11	4.58E-08	0.04458	54

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
PROBPENL	500	3.99E-07	2.13E-06	0.00116	9	2.32E-09	1.08E-06	0.82552	433
PSPDOC	4	2.00E+00	1.27E-04	0.00013	16	3.99E-07	3.07E-07	0.00041	4
QR3DLS	610	1.96E-01	1.61E-02	0.32834	873	2.00E+00	1.76E-06	0.00007	15
QRTQUAD	5000	-	-	-	-	-	-	-	-
QUARTC	5000	3.11E+03	3.82E+00	0.02981	36	-	-	-	-
QUDLIN	5000	-1.20E+62	2.74E+31	0.02699	31	3.01E+00	3.60E-02	0.02915	44
RAYBENDL	2050	9.80E+01	7.07E-01	0.00065	2	-	-	-	-
RAYBENDS	2050	9.80E+01	7.08E-01	0.0177	2	-	-	-	-
ROSENBR	2	3.00E-15	3.15E-07	0.00012	66	-	-	-	-
S308	2	7.73E-01	7.07E-04	0.00007	13	2.78E-17	1.44E-07	0.00007	48
S368	8	-	-	-	-	7.73E-01	7.79E-07	0.00003	14
SBRYBND	5000	-	-	-	-	-	-	-	-
SCHMVETT	5000	-1.50E+04	7.84E-03	0.10273	51	-	-	-	-
SCOND1LS	5002	-	-	-	-	-1.50E+04	2.77E-04	0.06677	35
SCOSINE	5000	2.58E+03	5.25E+09	0.7693	575	7.47E-03	2.60E-05	8.3935	6688
SCURLY10	10000	-	-	-	-	-	-	-	-
SCURLY20	10000	-	-	-	-	-	-	-	-
SCURLY30	10000	-	-	-	-	-	-	-	-
SENSORS	100	-2.11E+03	8.53E-02	0.16901	68	-	-	-	-
SIM2BQP	2	-1.39E-01	1.92E-05	0.00009	10	-2.11E+03	9.79E-06	0.05728	23
SIMBQP	2	-1.39E-01	1.92E-05	0.00009	10	-1.39E-01	1.37E-06	0.00005	5
SINEALI	1000	4.31E+04	1.46E+14	0.00694	17	-1.39E-01	1.37E-06	0.00006	5
SINEVAL	2	2.27E+21	3.01E+12	0.00007	23	-9.99E+04	7.09E-05	0.01013	49
SINQUAD	5000	-6.76E+06	3.31E-02	0.05881	39	9.40E-18	1.43E-07	0.00011	97
SISSER	2	1.07E-05	8.51E-04	0.00007	14	-	-	-	-
SNAIL	2	7.91E+05	2.27E+05	0.00006	17	1.16E-08	5.89E-06	0.00003	12
SPARSINE	5000	2.39E-01	4.92E-02	14.05977	7611	5.42E-12	3.65E-06	0.00017	143
SPARSQUR	10000	6.52E-06	1.01E-04	0.13986	42	-	-	-	-
SPECAN	9	5.43E+04	2.70E-02	0.21829	125	1.57E-08	9.62E-07	0.11678	39
SPMSRTLS	4999	2.28E-03	5.99E-03	0.21985	148	4.98E-11	1.42E-04	0.31638	168
SROSENBR	5000	2.32E-08	1.12E-04	0.01672	18	2.10E-07	5.20E-05	0.21979	167

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
SSBRYBND	5000	8.24E-10	1.55E-04	12.68708	7363	4.76E-15	4.37E-08	0.01593	20
SSC	5184	-	-	-	-	3.00E-14	1.01E-06	15.11032	9779
SSCOSINE	10	-1.76E+00	6.53E+12	0.00017	39	-	-	-	-
STRATEC	10	-	-	-	-	-	-	-	-
TESTQUAD	5000	1.62E-08	3.85E-04	3.90243	4168	-	-	-	-
TOINTGOR	50	1.37E+03	1.19E-02	0.00109	67	1.17E-12	2.93E-06	3.46883	4823
TOINTGSS	5000	1.00E+01	1.98E-03	0.0173	15	1.37E+03	1.10E-04	0.00129	107
TOINTPSP	50	2.26E+02	5.06E-03	0.00126	97	1.00E+01	1.06E-05	0.01589	18
TOINTQOR	50	1.18E+03	2.96E-03	0.00034	25	2.26E+02	7.91E-05	0.001	102
TORSION1	5476	-1.93E+02	8.43E-02	0.29341	165	1.18E+03	6.69E-05	0.0004	37
TORSION2	5476	-1.28E+02	1.49E-01	0.17176	98	-2.58E+06	3.30E+01	0.35931	223
TORSION3	5476	-6.32E+02	2.43E-01	0.29451	167	-2.22E+06	4.09E+01	0.1921	125
TORSION4	5476	-5.11E+02	2.98E-01	0.17186	98	-5.13E+06	3.28E+01	0.37729	237
TORSION5	5476	-2.32E+03	5.03E-01	0.23474	133	-5.95E+06	7.06E+01	0.19818	122
TORSION6	5476	-2.05E+03	5.95E-01	0.1724	98	-2.13E+07	6.68E+01	0.25263	158
TORSIONA	5476	-1.95E+02	9.91E-02	0.28839	152	-5.81E+07	1.73E+02	0.19827	129
TORSIONB	5476	-1.36E+02	3.40E-01	0.20855	111	-5.36E+06	7.31E+01	0.4359	246
TORSIONC	5476	-1.05E+03	2.28E-01	0.38466	202	-2.11E+06	2.29E+01	0.17605	105
TORSIOND	5476	-5.45E+02	6.81E-01	0.20878	111	-6.93E+06	5.13E+01	0.28507	163
TORSIONE	5476	-2.89E+03	6.30E-01	0.28208	148	-9.91E+06	7.32E+01	0.16801	100
TORSIONF	5476	-2.18E+03	1.36E+00	0.20963	111	-2.07E+07	8.28E+01	0.22357	131
TQUARTIC	5000	2.99E-08	1.95E-02	0.06512	56	-2.20E+07	1.26E+02	0.23338	135
TRIDIA	5000	3.52E-09	1.92E-04	1.45461	1452	1.54E-14	1.60E-06	0.01955	26
VARDIM	200	3.44E-13	2.35E-04	0.00195	58	2.47E-13	1.28E-06	1.54801	2079
VAREIGVL	50	8.30E-09	8.80E-05	0.00037	19	2.89E-18	6.80E-07	0.00128	42
VIBRBEAM	8	-	-	-	-	6.96E-13	8.39E-07	0.00037	25
WALL10	1461	-5.57E+05	7.07E+00	2.1085	3649	-	-	-	-
WALL100	149624	-5.00E+81	2.36E+42	38.48618	631	-	-	-	-
WALL20	5924	-	-	-	-	-	-	-	-
WALL50	37311	-5.89E+74	1.13E+39	20.28027	1334	-	-	-	-
WATSON	12	1.36E-05	5.32E-04	0.00156	150	-	-	-	-

Table A.1: The details of the comparison of RLBFGS and LBFGS

		RLBFGS				LBFGS			
Problem	n	f^*	g^*	Time (Sec.)	n_f	f^*	g^*	Time (Sec.)	n_f
WEEDS	3	-	-	-	-	1.54E-07	5.55E-06	0.007	748
WOODS	4000	8.08E-08	2.32E-04	0.14525	163	2.59E+00	4.86E-04	0.00036	110
YFIT	3	4.51E-02	3.97E+00	0.00046	85	1.38E-11	2.97E-06	0.07053	117
YFITU	3	4.51E-02	3.97E+00	0.00035	85	7.00E-13	8.01E-05	0.0004	98
ZANGWIL2	2	-1.82E+01	1.90E-04	0.00006	7	7.00E-13	8.01E-05	0.00035	98

Master's Thesis

A regularized limited memory BFGS method for unconstrained minimization problems

Guidance

Associate Professor Nobuo YAMASHITA

Shinji SUGIMOTO

Department of Applied Mathematics and Physics

Graduate School of Informatics

Kyoto University

February 2014

A regularized limited memory BFGS method for unconstrained minimization problems	Shinji SUGIMOTO	February 2014

A regularized limited memory BFGS method for unconstrained minimization problems

Shinji SUGIMOTO

Abstract

The limited memory BFGS (L-BFGS) method is one of the popular methods for solving large scale unconstrained minimization problems. The L-BFGS method works with small memory, and it is more efficient than the steepest descent method. However, since L-BFGS method needs line search with the Wolfe condition, it may require many function evaluations for ill-posed problems. To overcome the difficulty, the L-BFGS method combined with the trust region method has been proposed, and it has nice performances in terms of the number of function evaluations. However, the trust region method solves a nonconvex subproblem at each iteration, and hence it takes much time for the large scale problems.

In this paper, we propose a method which combines the L-BFGS method with the regularized Newton method instead of the trust region method. The computational cost for the single iteration of the proposed method is same as that of the original L-BFGS method. We show that the proposed method has global convergence under usual conditions. Moreover, we presents numerical results that show the rubustness of the proposed method.