Menemui Matematik (Discovering Mathematics)
Vol. 40, No. 1: 31-45 (2018)

Efficient ODE-based Methods for Unconstrained Optimization

Yap Chui Ying[1] and Leong Wah June[2]
[1,2]*Department of Mathematics, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor*
[1]cy-yap89@hotmail.com, [2]leongwj@upm.edu.my

ABSTRACT

This paper presents some efficient methods for unconstrained optimization based upon approximating the gradient flow of the objective function. Most ODE-based methods would generate Levenberg-Marquardt-like steps that require the solution of linear systems. On the other hand our proposed methods used some quasi-Newton matrices to approximate the solution of these linear systems, thus avoiding the solution of linear systems repeatedly. Two implementations of the modified ODE-based methods - line search and trust region implementation are proposed. Under some suitable assumptions, the convergence of the proposed methods is then established. Numerical results indicate that the modified methods are more effective and comparable than the standard line search and trust region method using the well-known BFGS formula.

Keywords: Gradient flow, Line search method, Quasi-Newton formula, Trust region method, Unconstrained optimization

INTRODUCTION

In this paper, we consider the following unconstrained optimization problems:

$$\min_{x \in R^n} f(x), \tag{1}$$

where the objective function $f(x)$ is assumed to be twice continuously differentiable for all x in R^n.

There have been enormous development of powerful algorithms for solving (1) numerically and these methods are generally iterative. In an iterative algorithm, an initial point x_0 is given and a new iterative point x_k is to be computed. For each *k*-iteration, the next iterative point x_{k+1} depends on the information at the current iterative point x_k and the objective function f. The computation can also be done by using the information stored from earlier iterates. Hopefully the sequence $\{x_k\}$ generated will converge to the solution of (1) satisfying the first and second-order necessary conditions for a local multivariate unconstrained minimum:

$$\nabla f(x^*) = 0, \tag{2}$$

and $\nabla^2 f(x^*)$ be positive definite (or at least positive semi-definite), then an optimum solution x^* of f can be obtained.

Presently, these iterative algorithms can be categorized into two broad classes which are line search methods and trust region methods. The line search methods are presented in the form

$$x_{k+1} = x_k + \alpha_k d_k, \forall k \geq 0. \tag{3}$$

Particularly, quasi-Newton methods (see, for example, Khiyabani and Leong (2014), Wu and Sun (2006)) for minimizing $f(x)$ are commonly used especially when the analytical expression of the second derivative of $f(x)$, called the Hessian is hard to obtain or is expensive to compute or store. Quasi-Newton methods compute a search direction first:

$$d_k = -B_k^{-1} \nabla f(x_k), \tag{4}$$

where B_k is an $n \times n$ symmetric matrix that approximates to the Hessian via updating formula, such as Broyden, Fletcher, Goldfrab and Shanno (BFGS) formula (see, for example, Broyden (1970)):

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \tag{5}$$

where $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$ and $s_k = x_{k+1} - x_k$. Using the Sherman-Morrison Householder formula, the update formula for the approximation H_k of the inverse Hessian can be obtained as follow:

$$H_{k+1} = H_k + \left(1 + \frac{y_k^T H_k y_k}{s_k^T y_k}\right) \frac{s_k s_k^T}{s_k^T y_k} - \frac{s_k y_k^T H_k + H_k y_k s_k^T}{s_k^T y_k}. \tag{6}$$

Then a steplength α_k is obtained by searching along that particular search direction using an appropriate line search in order to enforce global convergence. Hence, a new iterate point x_{k+1} is computed and the process is continues until the sequence $\{x_k\}$ reaches the termination criterion.

On the other hand, trust region methods have attracted much attention from more and more researchers since their emergence (see, for example, Conn et al. (2000), Wang et al. (2008)). This is basically due to they can guarantee strong global convergence for solving (1). The basic idea of trust region methods is they define a region around the current iterate in which the relatively simple model is “trusted” to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this trust region. Traditionally, a quadratic model is usually being chosen as the approximation to the $f(x)$. Thus, trust region methods compute a trial step by solving the following subproblem:

$$\min q_k(d_k) = f_k + \nabla f(x_k)^T d_k + \frac{1}{2} d_k{}^T B_k d_k, \quad \text{s.t.} \quad \|d_k\| \le \Delta_k, \tag{7}$$

where $d_k = x_{k+1} - x_k$, B_k is an $n \times n$ symmetric matrix which approximates to the Hessian of the objective function and $\Delta_k > 0$ is the radius of trust region. We will describe trust region methods in more detail in the later section. For an in-depth overview of trust region methods, refer to Conn et al. (2000). Some authors also proposed a hybrid algorithm which combines line search methods with trust region methods (see, for example, Gertz (2004), Nocedal and Yuan (1998)).

GRADIENT FLOW SYSTEM AND QUASI-NEWTON UPDATING

Traditionally, optimal control, gradient flow systems and partial differential equations are ideas from different fields of mathematics that have been rather disparate. However, the interactions between these ideas have surprisingly sparked the curiosity from many researchers in these recent years. Courant (1962) was the first to propose the method of gradients in year 1941 for solving variational partial differential equations. Courant considered the use of the following gradient flow system:

$$\dot{x}(t) = -\nabla f(x(t)), \tag{8}$$

with the initial-value condition

$$x(0) = x_0, \tag{9}$$

to obtain the equilibrium point x^* such that

$$\nabla f(x^*) = 0, \tag{10}$$

The solution is called an integral curve and is simply the curve that at each instant proceeds in the direction of the steepest descent of f.

ODE-based methods for solving (8)-(9) proceed by discretizing the time step in (8) to obtain some difference equation. Since we are mainly interested in the long term behaviour of the gradient flow system (8), and do not concern about the accurate solutions of its immediate courses, we consider implicit (backward) Euler method for (8). Another reason that motivates our choice is that the implicit Euler method is unconditionally stable regardless of the (discretized) time step. Hence, it avoids the unnecessary amount of computational effort to control the time step using the local error principal. Applying the implicit Euler method to (8) gives

$$x_{k+1} = x_k - \overline{\alpha}_k \nabla f(x_{k+1}); k = 0,1,2,... \tag{11}$$

One can view (11) as a line search method where $\overline{\alpha}_k$ is the steplength and $-\nabla f(x_{k+1})$ is the search direction. As oppose, the steepest descent method is obtained if we apply the forward Euler method to (8). However, stability of the forward Euler method requires a special attention on the choices of time step. Thus, we avoid this method.
An obvious difficulty on the implementation of (11) is that it requires the computation of the gradient at the unknown future point, x_{k+1}. Hence, some adequate approximation is needed for $\nabla f(x_{k+1})$. Usually, the approximation of f can be done through second order Taylor expansion, i.e. quadratic approximation. For this purpose one must define a region in which the approximation used is valid. Since the integral curve of the negative gradient field is not available to us, we propose to measure the validity of the quadratic approximation by a simple strategy that is often used within a trust region framework, namely a ratio test. To express our algorithm, suppose that given x_k and a ball region centered at x_k with radius Δ_k. Let $d_k = \nabla f(x_{k+1})$, then the quadratic approximation for f gives

$$d_k = -\overline{\alpha}_k \nabla f(x_{k+1}) \approx -[\nabla f(x_k) + \overline{\alpha}_k \nabla^2 f(x_k) d_k], \tag{12}$$

where $\nabla^2 f(x_k)$ is the Hessian matrix of f at x_k. Hence, d_k can be obtained by rearranging (12):

$$[I + \overline{\alpha}_k \nabla^2 f(x_k)] d_k = \overline{\alpha}_k \nabla f(x_k), \tag{13}$$

and therefore, the iteration equation (11) becomes

$$x_{k+1} = x_k - \overline{\alpha}_k [I + \overline{\alpha}_k \nabla^2 f(x_k)]^{-1} \nabla f(x_k), \tag{14}$$

Since the implicit Euler method is unconditional stable, we can choose any fixed arbitrary small $\overline{\alpha}_k$. However, it is desirable to use a variable step length to ensure convergence. Hence solving (13) exactly to obtain d_k is not feasible particularly we may need to solve the linear system multiple time for different steplengths. Hence, we shall propose some approximation for the inversion through some quasi-Newton updating scheme. Suppose that a quasi-Newton updating matrix, B_{k+1} is used to approximate $\nabla^2 f(x_k)$, then B_{k+1} should satisfy the quasi-Newton (QN) equation:

$$B_{k+1} s_k = y_k, \tag{15}$$

where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$. Then if an updating matrix, $\overline{B}_{k+1}$ would approximate $I + \overline{\alpha}_k \nabla^2 f(x_k)$, it should obey

$$\overline{B}_{k+1} s_k \approx s_k + \overline{\alpha}_k B_{k+1} s_k = s_k + \overline{\alpha}_k y_k = \overline{y}_k. \tag{16}$$

Here, $\overline{B}_{k+1}$ may be chosen to be any symmetric approximation that satisfying (16). In this paper we shall employ the famous BFGS updating formula (5) to generate the matrix sequence $\{\overline{B}_k\}$, i.e.,

$$\overline{B}_0 = 1, \overline{B}_{k+1} = \overline{B}_k - \frac{\overline{B}_k s_k s_k^T \overline{B}_k}{s_k^T \overline{B}_k s_k} + \frac{\overline{y}_k \overline{y}_k^T}{\overline{y}_k^T s_k}, \tag{17}$$

By using the Sherman-Morrison Householder formula, we can obtain the inverse of (17) explicitly, which is more convenient for the algorithmic purpose:

$$\overline{H}_{k+1} = \overline{H}_k + \left(1 + \frac{\overline{y}_k^T \overline{H}_k \overline{y}_k}{s_k^T \overline{y}_k}\right) \frac{s_k s_k^T}{s_k^T \overline{y}_k} - \frac{s_k \overline{y}_k^T \overline{H}_k + \overline{H}_k \overline{y}_k s_k^T}{s_k^T \overline{y}_k}, \tag{18}$$

where $\overline{H}_k = \overline{B}_k^{-1}$.

LINE SEARCH IMPLEMENTATION

Traditionally, a variable steplength strategy makes decision on the size of the step using information on some (local) error estimation. If the current step is not acceptable, one can reduce the steplength by a portion of its original size. Otherwise, the steplength is enlarged. Therefore, using the reduction in objective function as an indicator, it is equivalent to perform a line search procedure to obtain the steplength.

There are a large number of line search procedures (see, for example, Goldstein (1965), Wolfe (1969), Wolfe (1971)), among which the Armijo backtracking line search Armijo (1966) is in common use where its general algorithm is as follows:

Algorithm 3.1 (Armijo backtracking line search):

Step 0. The constants $\eta \in (0,1)$ and τ_1, τ_2 with $0 < \tau_1 < \tau_2 < 1$, are given.
Step 1. Set $\overline{\alpha}_k = 1$.
Step 2. Test the relation

$$f(x_k + \overline{\alpha}_k d_k) \le f(x_k) + \eta \overline{\alpha}_k \nabla f(x_k)^T d_k. \tag{19}$$

Step 3. If (19) is not satisfied, choose a new $\overline{\alpha}_k \in [\tau_1 \overline{\alpha}_k, \tau_2 \overline{\alpha}_k]$ and go to Step 2.
If (19) is satisfied, set $\overline{\alpha}_{k+1} = \overline{\alpha}_k$ and $x_{k+1} = x_k + \overline{\alpha}_{k+1} d_k$.

Several procedures have been used to choose a new trial value of $\overline{\alpha}_k$ in Step 3. The classical Armijo line search is to simply multiply the old value of $\overline{\alpha}_k$ by $\frac{1}{2}$ or some other constant in (0,1) until (19) is satisfied. Using the backtracking line search, an algorithm of line search ODE-based method can be outlined as follows:

ODE-LS method:

Step 0. Choose an initial point $x_0 \in R^n$ and $\overline{B}_0 = I$. Let $k := 0$.

Step 1. Compute $\nabla f(x_k)$. If the stopping criterion $\|\nabla f(x_k)\| \le \varepsilon$ is reached, then stop. Else go to Step 2.

Step 2. Compute $d_k = -\overline{B}_k^{-1}\nabla f(x_k)$ and calculate $\overline{\alpha}_k > 0$ using Algorithm 3.1. Set $x_{k+1} = x_k - \overline{\alpha}_k \overline{B}_k^{-1}\nabla f(x_k)$ and update $\overline{B}_{k+1}$ by (17).

Step 3. Set $k := k+1$ and return to Step 1.

Remark 1. In practical computation, $\overline{H}_k$ is used instead of $-\overline{B}_k^{-1}$. Also since $\overline{\alpha}_k$ appears explicitly in $\overline{y}_k$, this does not incur additional computational cost when compute $\overline{\alpha}_k$ using (18) within Algorithm 3.1.

To study the convergence properties of the algorithm, we consider the following standard assumptions.

Assumption 3.1:

A1. The level set $D = \{x \in R^n \mid f(x) \le f(x_0)\}$ is bounded for a chosen x_0.

A2. The gradient is Lipschitz continuous on an open convex set S contained in D, namely there exists a constant $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|, \forall x, y \in S. \tag{20}$$

A3. There exists positive constants m and M such that

$$m\|z\|^2 \le z^T \nabla^2 f(x) z \le M\|z\|^2, \tag{21}$$

for all $z \in R^n$ and $x \in D$, where, and for the rest of this paper, $\|\cdot\|$ denotes the l_2 vector norm.

Lemma 3.1. *Under Assumption 3.1, there exist positive constants* $\overline{L}$, $\overline{m}$ *and* $\overline{M}$ *such that*

$$\|\overline{y}_k\| \le \overline{L}\|s_k\|, \tag{22}$$

$$\overline{m}\|s_k\|^2 \le s_k^T \overline{y}_k \le \overline{M}\|s_k\|^2, \ \forall k \ge 0. \tag{23}$$

Proof. By the Lipschitz continuity (20) and the fact that $\overline{\alpha}_k \in (0,1]$, we have

$$\|\overline{y}_k\| = \|s_k + \overline{\alpha}_k y_k\| \le \|s_k\| + \overline{\alpha}_k\|y_k\| \le (1+L)\|s_k\| = \overline{L}\|s_k\|, \tag{24}$$

and

$$s_k^T s_k \le s_k^T \bar{y}_k \le s_k^T s_k + s_k^T y_k \le (1+M)\|s_k\|^2, \tag{25}$$

where $\bar{m} = 1$ and $\bar{M} = 1 + M$.

Lemma 3.1 implies that with the Lipschitz condition as well as the boundedness on the curvature hold for $\bar{y}_k$ and thus, one can proceed with a similar proof by Byrd and Nocedal (1989) to show that the modified BFGS method under backtracking line search with $\bar{y}_k$ is globally and superlinearly convergent on objective functions that satisfied Assumption 1.

TRUST REGION APPROACH

Alternative to the adjustment on steplength to achieve sufficient descent in objective function, the trust region method adjust a trust region bound Δ_k on the norm of the trial step d_k. It can be easily seen that (14) is equivalent to Levenberg-Marquardt method for unconstrained optimization, which obtain a trial step by solving the following linear equation at k -th iteration,

$$[\lambda_k I + \nabla^2 f(x_k)]d_k = -\nabla f(x_k), \tag{26}$$

when $\lambda_k = \dfrac{1}{\alpha_k}$.

In order to use the iterative equation in the trust region method for solving unconstrained minimization problem (1), we construct the following subproblems:

$$SP1: \begin{cases} \min q_k(d_k) = f(x_k) + \nabla f(x_k)^T d_k + \frac{1}{2} d_k^T B_k d_k, \\ \text{s.t.} \quad \|d_k\| \le \Delta_k, \end{cases} \tag{27}$$

where B_k is some symmetry approximation of the Hessian matrix at x_k, and

$$SP2: \begin{cases} \min q_k(d_k) = f(x_k) + \nabla f(x_k)^T d_k + \frac{1}{2} d_k^T \bar{B}_k d_k \\ \qquad = f(x_k) + \frac{1}{2} \nabla f(x_k)^T d_k \\ \text{s.t.} \quad \|d_k\| \le \Delta_k. \end{cases} \tag{28}$$

Remark 2. In this first subproblem, a standard quadratic model is considered where the computed trial step is benchmarked with the quasi-Newton step (or the Newton step, if

$B_k = \nabla^2 f(x_k)$). On the other hand, the second subproblem uses the quadratic model that based upon second order Taylor expansion on (14).

Similar to the line search method, we can approximate the trial step using some quasi-Newton approximating matrix. Let $\hat{B}_{k+1} \approx [\lambda_k I + \nabla^2 f(x_k)]$, then,

$$\hat{B}_{k+1} s_k \approx [\lambda_k I + \nabla^2 f(x_k)] s_k \approx \lambda_k s_k + y_k = \hat{y}_k. \quad (29)$$

Hence, a BFGS updating matrix that based upon (29) can be obtained by

$$\hat{B}_0 = 1, \hat{B}_{k+1} = \hat{B}_k - \frac{\hat{B}_k s_k s_k^T \hat{B}_k}{s_k^T \hat{B}_k s_k} + \frac{\hat{y}_k \hat{y}_k^T}{\hat{y}_k^T s_k}. \quad (30)$$

Again if $\hat{H}_k = \hat{B}_k^{-1}$, we can compute

$$\hat{H}_{k+1} = \hat{H}_k + \left(1 + \frac{\hat{y}_k^T \hat{H}_k \hat{y}_k}{s_k^T \hat{y}_k}\right) \frac{s_k s_k^T}{s_k^T \hat{y}_k} - \frac{s_k \hat{y}_k^T \hat{H}_k + \hat{H}_k \hat{y}_k s_k^T}{s_k^T \hat{y}_k}. \quad (31)$$

The steplength is fixed rather than being computed through some line searches procedure because we shall adjust the radius of the region to obtain desired reduction in the objective function within a trust region framework. For this purpose, we can emply the following steplength, which is due to Ou et al. (2009):

$$\overline{\alpha}_0 = 1, \overline{\alpha}_k = -\frac{\nabla f(x_k)^T d_k}{L_k (d_k^T d_k)}, \forall k \geq 1, \quad (32)$$

where $L_k = \max\left\{1, \frac{\|y_{k-1}\|}{\|s_{k-1}\|}\right\}$.

Before we proceed to state our proposed algorithm, we present the characterization of the solution of subproblem (27) by giving the following well known lemmas (see, for example, Yuan (2000)):

Lemma 4.1. *A vector* $d_k^* \in R^n$ *is a solution of the subproblem (27), if and only if* $\|d_k\| \leq \Delta_k$ *and there exists* $\alpha^* \geq 0$ *such that*

$$(B_k + \alpha^* I) d_k^* = -\nabla f(x_k), \quad (33)$$

$$\alpha^* (\Delta_k - \|d_k^*\|) = 0, \quad (34)$$

$$B_k + \alpha^* I \geq 0. \quad (35)$$

Lemma 4.2. *If a vector* $d_k^* \in R^n$ *is a solution of the subproblem (27), then*

$$q_k(0) - q_k(d_k^{\,*}) \ge \frac{1}{2}\|\nabla f(x_k)\| \min\left(\Delta_k, \frac{\|\nabla f(x_k)\|}{\|B_k\|}\right). \tag{36}$$

In each iteration, a trial step d_k is computed approximately using (30). Hence, the conditions in Lemma 4.1 may not hold and thus, the inequality in Lemma 4.2 must also be satisfied on every iteration.

Lemma 4.3. *Let Assumptions 3.1 hold. If Algorithm ODE-TR-I generates d_k that satisfies conditions (33)-(36), then*

$$\liminf_{k \to \infty} \|\nabla f(x_k)\| = 0. \tag{37}$$

Proof. Since the inequality in Lemma 4.2 is satisfied on every iteration, then either $x_k + d_k$ is accepted as a new iteration point or rejected according to the ratio of the comparison between the *actual reduction* in f

$$ared_k = f(x_k) - f(x_k + d_k), \tag{38}$$

with the *predicted reduction* in the approximation model

$$pred_k = q_k(0) - q_k(d_k). \tag{39}$$

i.e.,

$$\rho_k = \frac{ared_k}{pred_k}. \tag{40}$$

If ρ_k is close to 1 which means the reduction in f is satisfactory, then we accept the trial step and expand the trust region for the next iteration; if ρ_k is close to zero or negative, we contract the trust region for the next iteration; otherwise, we do not alter the trust region.

Note that $pred_k$ will always be nonnegative and hence a negative ρ_k implies that the new objective function value $f(x_k + d_k)$ is greater than the current function value $f(x_k)$. In other words, the vector x_{k+1} is also defined by the equation

$$x_{k+1} = \begin{cases} x_k + d_k, & f(x_k + d_k) < f(x_k), \\ x_k, & f(x_k + d_k) \ge f(x_k), \end{cases} \tag{41}$$

which provides the condition

$$f(x_k + d_k) \le f(x_k). \tag{42}$$

Since f is a decreasing sequence and f is bounded below, then $\lim_{k\to\infty} ared_k = 0$ implies (37).

Now we describe our trust region ODE-based algorithm as follows.

ODE-TR-I algorithm (ODE-TR using subproblem SP1):

Step 0. *Initialization*. Choose an initial point $x_0 \in R^n$, and an initial trust region radius $\Delta_0 \in (0, \tilde{\Delta}]$. Let B_0 and H_0 be a symmetric positive definite matrix. Choose the constants $\eta_1, \eta_{2,} \tau_1, \tau_2$ and ε satisfying $0 \le \eta_1 < \eta_2 < 1$, $0 < \tau_1 < 1 < \tau_2$ and $0 \le \varepsilon \le 1$. Set $k := 0$.

Step 1. *Test for convergence criterion*. Compute $\nabla f(x_k)$. If $\|\nabla f(x_k)\| \le \varepsilon$, then stop. Else go to Step 2.

Step 2. *Model definition*. Choose $\|\cdot\|_2$ and use quadratic model SP1.

Step 3. *Step Calculation*. Compute d_k that "sufficiently reduces the model" q_k for which $\|d_k\| \le \Delta_k$ such that

$$q_k(0) - q_k(d_k^{\ *}) \ge \frac{1}{2}\|\nabla f(x_k)\| \min\left(\Delta_k, \frac{\|\nabla f(x_k)\|}{\|B_k\|}\right), \qquad (43)$$

by using the following linear equation $d_k = -\hat{H}_k \nabla f(x_k)$.

Step 4. *Computation for the ratio of reduction*. Compute $f(x_k + d_k)$ and then calculate ρ_k using (40).

Step 5. *Update for trust region radius*. Set

$$\Delta_{k+1} = \begin{cases} \tau_1 \Delta_k, & \text{if } \rho_k \le \eta_1, \\ \Delta_k, & \text{if } \eta_1 < \rho_k < \eta_2, \\ \min\{\tau_2 \Delta_k, \tilde{\Delta}\}, & \text{if } \rho_k \ge \eta_2, \|d_k\| = \Delta_k. \end{cases} \qquad (44)$$

Step 6. *Acceptance of the trial point*. If $\rho_k \ge \eta_1$, define $x_{k+1} = x_k + d_k$ and then update matrices B_{k+1} and H_{k+1} using (5) and (31) respectively.
Go to Step 1. Otherwise define $x_{k+1} = x_k$ and go to Step 2. Set $k := k+1$.

ODE-TR-II algorithm: Equivalent to ODE-TR-I algorithm except that B_{k+1} is updated by (30) where α_k is given by (32).

NUMERICAL RESULTS AND DISCUSSION

In this section, we present and discuss some numerical experiments that were implemented on some famous optimization test problems in order to test the efficiency of the proposed algorithms. We report the numerical results for all the five Algorithms BFGS-LS, ODE-LS, TR-BFGS, ODE-TR-I and ODETR-II and to check whether the modified methods provide improvements over the standard method.

We use Matlab R2012b programming language to code the procedures and implement them on a PC with 2.5GHz processor and 4.00GB RAM. The 65 problems tested are the unconstrained problems with standard starting points from Neculei (2008) and the convergence test is used with $\varepsilon = 10^{-3}$. For each test problem, we have considered 3 different dimensions which are $n = 10,100$ and 1000. All algorithms use exactly the same set of parameters which are $\eta_1 = 0.1, \eta_2 = 0.75, \tau_1 = 0.5, \tau_2 = 2$ and $\widetilde{\Delta} = 2$. Other than that, we choose the initial matrix B_0 and H_0 as the Identity matrices and we also restrict the number of iteration within 1000. For all the runs which the convergence criterion is not fulfilled within the maximum number of iterations, we consider it as failure. In order to compare and evaluate the performance of the proposed methods, we use the performance profiling proposed by Dolan and Moré (2002). Comparison is made in term of number of iterations, function calls and CPU time.

Table 1: Test Problem

Test Functions
Almost Perturbed Quadratic, ARWHEAD, BIGGSB1, COSINE, CUBE, Diagonal 1, Diagonal 2, Diagonal 3, Diagonal 4, Diagonal 5, Diagonal 6, Diagonal 7, Diagonal 8, Diagonal 9, DIXON3DQ, DQDRTIC, EDENSCH, EG2, ENGVALI, EXPLIN1, Extended BD1, Extended Beale, Extended Cliff, Extended DENSCHNB, Extended DENSCHNF, Extended EP1, Extended Freudenstein & Roth, Extended Hiebert, Extended Himmelblau, Extended Maratos, Extended Penalty, Extended Powell, Extended PSC1, Extended QP1, Extended QP2, Extended Rosenbrock, Extended TET, Extended Tridiagonal 1, Extended White & Holst, Extended Wood, FH1, FH2, FH3, FLETCHCR, Generalized PSC1, Generalized Quartic, Generalized Rosenbrock, Generalized Tridiagonal1, Generalized Tridiagonal 2, Generalized White & Holst, Hager, HIMMELBG, HIMMELH, LIARWHD, MCCORMCK, NONDIA, NONSCOMP, Perturbed Quadratic Diagonal, Perturbed Tridiagonal Quadratic, POWER, Quadratic QF1, Quadratic QF2, Raydan 1, Raydan 2, SINE.

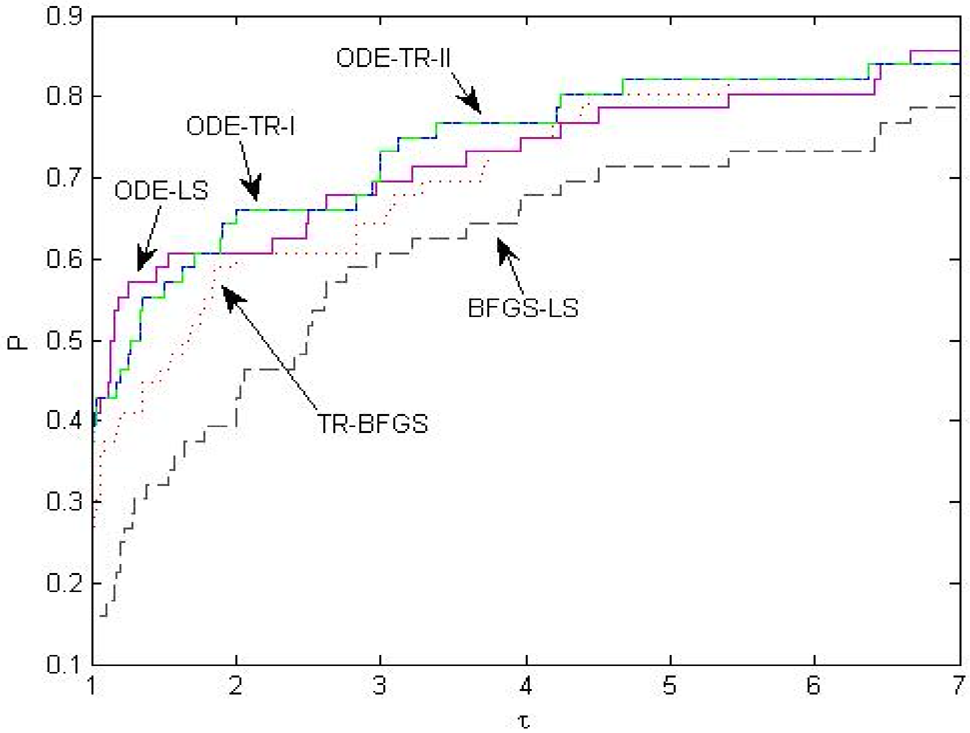

Figure 1: Performance profiles based on number of iterations

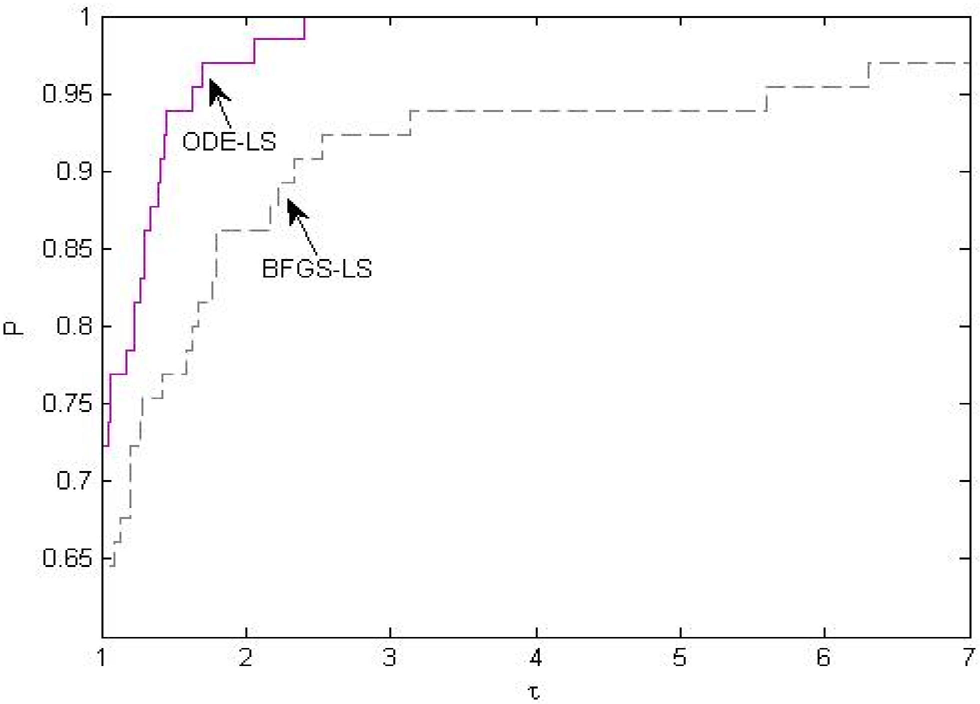

Figure 2: Performance profiles based on number of function evaluations (Trust region methods require one function calls per iteration.)

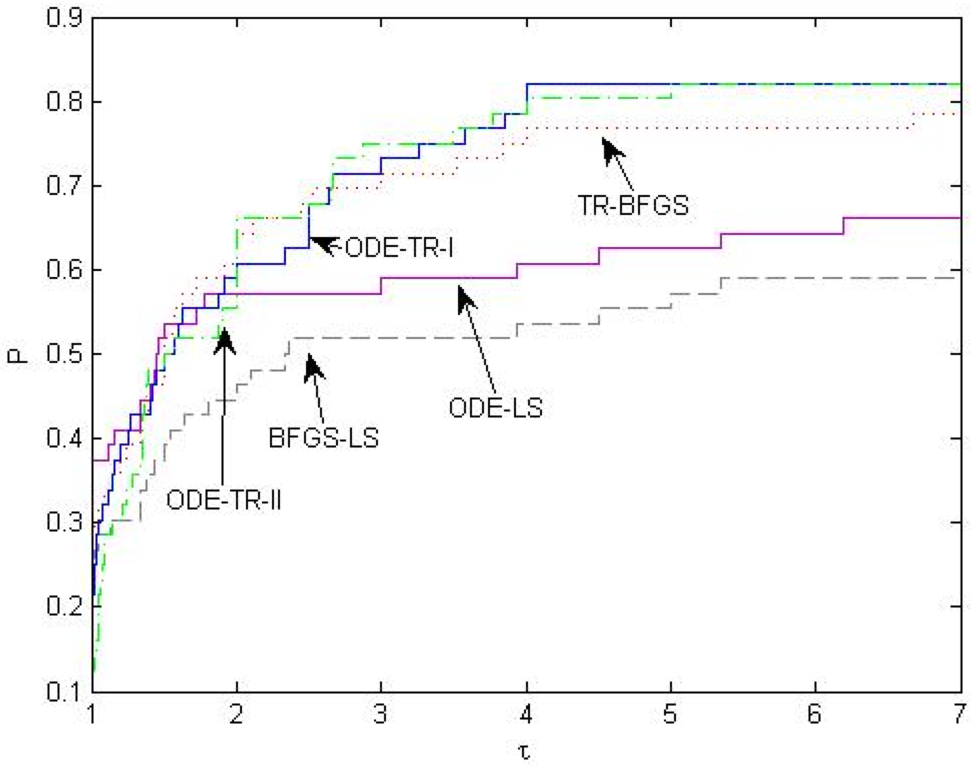

Figure 3: Performance profiles based on CPU time in second

From the numerical results, we observed that our proposed methods ODELS, ODE-TRI and ODE-TRII were able to solve as many test problems as the standard BFGS-LS and TR-BFGS methods. In term of CPU time, our proposed methods are also faster than the standard methods. For most test problems, the standard BFGS-LS and TR-BFGS methods needed more iterations and function calls to achieve convergence. In general, the proposed methods are comparable and outperform the standard BFGS and trust region methods.

CONCLUSION

In this paper, we have proposed some new quasi-Newton-like approximation for ODE-based methods to solve unconstrained optimization problem. By using these approximations, solving linear systems repeatedly on each iteration can be avoided. From a computational point of view, this approach is efficient as it often requires lower number of iteration to obtain an acceptable solution. Convergence of the modified methods is also established under some suitable assumptions. Numerical results show that the proposed methods are competitive and effective than the standard line search and trust region method using BFGS formula.

ACKNOWLEDGMENTS

The first author is supported by University Putra Malaysia Graduate Research Fellowship, 2014.

REFERENCES

Armijo, L. (1966), Minimization of functions having Lipschitz continuous partial derivatives, *Pac. J. Math*, **16**: 1 – 3.

Behrman, W. (1998), *An efficient gradient flow method for unconstrained optimization*, Ph.D. thesis, Stanford University.

Broyden, C. G. (1970), The convergence of a class of double rank minimization algorithms: 2. The new algorithm, *J. Inst. Maths Applics*, **6(3)**: 222 – 231.

Byrd, R. H. and Nocedal, J. (1989), A tool for the analysis of quasi-Newton methods with application to unconstrained optimization, *SIAM J. Numer. Anal*, **26(3)**: 727 – 739.

Conn, A. R., Gould, N. I. M. and Toint, P. L. (2000), *Trust Region Methods*, MOS-SIAM Series on Optimization, SIAM, Philadelphia.

Courant, R. (1962), *Calculus of variations: with supplementary notes and exercises, 1945–1946.*: Revised and amended by J. Moser. New York University, Courant Institute of Mathematical Sciences, New York.

Dolan, E. D. and Moré, J.J. (2002), Benchmarking optimization software with performance profiles, *Math. Program*, **91(2)**: 201 – 203.

Gertz, E. M. (2004), A quasi-Newton trust-region method, *Mathematical Programming Series A*, **100**: 447 – 470.

Goh, B. S. (1997), Algorithms for Unconstrained Optimization Problems via Control Theory, *Journal of Optimization Theory and Applications*, **92(3)**: 581 – 604.

Goldstein, A. A. (1965), On steepest descent, *SIAM J. Control*, **3**: 147 – 151.

Khiyabani, F. M. and Leong, W. J. (2014), Quasi-Newton methods based on ordinary differential equation approach for unconstrained nonlinear optimization, *Applied Mathematics and Computation*, **233**: 272 – 291.

Leong, W. J., Hassan, A. M. and Yusuf, M. W. (2011), A matrix-free quasi-Newton method for solving large-scale nonlinear systems, *Computers and Mathematics with Applications*, **62(5)**: 2354 – 2363.

Neculai, A. (2008), An Unconstrained Optimization Test Functions Collection, *Advanced Modeling and Optimization*, **10(1)**: 147 – 161.

Nocedal, J and Wright, S. J. (1999), *Numerical Optimization, (Second Edition)*, Springer, New York.

Nocedal, J and Yuan, Y. (1998), Combining trust-region and line-search techniques, *Optimization Technology Center mar OTC*, **98(04)**.

Ou, Y, Zhou, Q and Lin, H. (2009), An ODE-based trust region method for unconstrained optimization problems, *J. Comp. Appl. Math.*, **232**: 318 – 326.

Powell, M. J. D. (1975), *Convergence properties of a class of minimization algorithms, In Nonlinear Programming 2*, Mangasarian, O. L., Meyer, R. R. and Robinson, S. M. eds. Academic Press, New York.

Powell, M. J. D. (1984), On the Global Convergence of Trust Region Algorithms for Unconstrained Minimization, *Mathematical Programming*, **29**: 297 – 303.

Wang, F, Zhang, K, Wang, C and Wang, L. (2008), A variant of trust-region methods for unconstrained optimization, *Applied Mathematics and Computation*, **203**: 297 – 307.

Wang, S, Yang, X. Q. and Teo, K. L. (2003), A unified gradient flow approach to constrained nonlinear optimization problems, *Comput. Optim. Appl.*, **25**: 251 – 268.

Wolfe, P. (1969), Convergence conditions for ascent methods, *SIAM Rev*, **11**: 226 – 235.

Wolfe, P. (1971), Convergence conditions for ascent methods II: some corrections, *SIAM Rev*, **13**: 185 – 188.

Wu, T and Sun, L. (2006), A quasi-Newton based pattern search algorithm for unconstrained optimization, *Applied Mathematics and Computation*, **183**: 685 – 694.

Yuan, Y. (2000), A review of trust region algorithms for optimization, *In Proceedings of the Fourth International Congress on Industrial and Applied Mathematics*, Ball, J. M. and Hunt, J. C. R. eds., 271–282.