Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 324181, 14 pages
http://dx.doi.org/10.1155/2014/324181

Research Article

Optimal Algorithms and the BFGS Updating Techniques for Solving Unconstrained Nonlinear Minimization Problems

Chein-Shan Liu

Department of Civil Engineering, National Taiwan University, Taipei 106-17, Taiwan

Correspondence should be addressed to Chein-Shan Liu; liucs@ntu.edu.tw

Received 5 November 2013; Revised 21 January 2014; Accepted 29 January 2014; Published 12 March 2014

Academic Editor: Jung-Fa Tsai

To solve an unconstrained nonlinear minimization problem, we propose an optimal algorithm (OA) as well as a globally optimal algorithm (GOA), by deflecting the gradient direction to the best descent direction at each iteration step, and with an optimal parameter being derived explicitly. An invariant manifold defined for the model problem in terms of a locally quadratic function is used to derive a purely iterative algorithm and the convergence is proven. Then, the rank-two updating techniques of BFGS are employed, which result in several novel algorithms as being faster than the steepest descent method (SDM) and the variable metric method (DFP). Six numerical examples are examined and compared with exact solutions, revealing that the new algorithms of OA, GOA, and the updated ones have superior computational efficiency and accuracy.

1. Introduction

The steepest descent method (SDM), which can be traced back to Cauchy (1847), is the simplest gradient method for solving unconstrained minimization problems. However, the SDM performs well during earlier stages and as approaching to a stationary point it converges very slowly. In this paper, we consider the following nonlinear minimization problem without considering constraint:

$$\min f(\mathbf{x}), \tag{1}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a $\mathbb{C}^2$ differentiable function.

In the iterative solution of (1), if $\mathbf{x}_k$ is the current iterative point, then we denote $f(\mathbf{x}_k)$ by f_k, $\nabla f(\mathbf{x}_k)$ by $\mathbf{g}_k$, and $\nabla^2 f(\mathbf{x}_k)$ by $\mathbf{A}_k$, which is known to be a symmetric Hessian matrix. The second-order Taylor expansion of function $f(x)$ at the point $\mathbf{x}_k$ is

$$f(\mathbf{x}) = f_k + \mathbf{g}_k^{\mathrm{T}}\Delta\mathbf{x} + \frac{1}{2}(\Delta\mathbf{x})^{\mathrm{T}}\mathbf{A}_k\Delta\mathbf{x}, \tag{2}$$

where $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_k$. The superscript T signifies the transpose and meanwhile $\mathbf{g}_k^{\mathrm{T}}\Delta\mathbf{x}$ signifies the inner product of $\mathbf{g}_k$ and $\Delta\mathbf{x}$.

Let $\mathbf{x} = \mathbf{x}_k - \lambda_0\mathbf{g}_k$, and inserting it into (2) we can obtain

$$f(\mathbf{x}_k - \lambda_0\mathbf{g}_k) = f_k - \lambda_0\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_k + \frac{\lambda_0^2}{2}\mathbf{g}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{g}_k. \tag{3}$$

By requiring the minimization with respect to λ_0, we can derive

$$\lambda_0 = \frac{\|\mathbf{g}_k\|^2}{\mathbf{g}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{g}_k}. \tag{4}$$

Then, we have a famous steepest descent method (SDM) for solving (1).

(i) Give an initial $\mathbf{x}_0$, and then compute $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$.

(ii) For $k = 0, 1, 2, \ldots$, we repeat the following iteration:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \frac{\|\mathbf{g}_k\|^2}{\mathbf{g}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{g}_k}\mathbf{g}_k. \tag{5}$$

If $\|\mathbf{g}_{k+1}\| < \varepsilon$, then stop; otherwise, go to step (ii).

For the minimization problem (1) we need to solve $\nabla f = \mathbf{0}$, and hence the residual means the value of $\|\nabla f\| = \|\mathbf{g}\|$. The above convergence criterion $\|\mathbf{g}_{k+1}\| < \varepsilon$ means that when

the residual norm is smaller than a given error tolerance ε, the iterations are terminated.

In the derivation of SDM for solving (1) it is easy to see that we have transformed the global minimization problem into a model problem in terms of a quadratic minimization problem of

$$\phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{x} + c_0 \tag{6}$$

and determined the coefficient λ_0 by (4), where $c_0 = f_k - \mathbf{g}_k^{\mathrm{T}}\mathbf{x}_k + \mathbf{x}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{x}_k/2 + k_0$ with k_0 being a constant to raise the level value of ϕ, and $\mathbf{b} = \mathbf{A}_k\mathbf{x}_k - \mathbf{g}_k$ is a constant vector within each iterative step. Here for the purpose of simple notation we omit the subscript k in (6), and we are going to modify the SDM by starting from the above locally quadratic function.

Several modifications of the SDM have been addressed. These modifications have led to a new interest in the SDM that the gradient vector itself is not a bad choice but rather that the original steplength λ_0 leads to a slow convergence behavior. Barzilai and Borwein [1] have presented a new choice of steplength through a two-point stepsize. Although their method did not guarantee the descent of the minimum function values, Barzilai and Borwein [1] were able to produce a substantial improvement of the convergence speed for a certain test of a quadratic function. The results of Barzilai and Borwein [1] have spurred many researches on the SDM, for example, Raydan [2, 3], Friedlander et al. [4], Raydan and Svaiter [5], Dai et al. [6], Dai and Liao [7], Dai and Yuan [8], Fletcher [9], and Yuan [10]. In this paper, we will approach this problem from a quite different viewpoint of invariant manifold and propose a new strategy to modify the steplength and the descent direction. Besides the SDM, there were many modifications of the conjugate gradient method for the unconstrained minimization problems, like Birgin and Martinez [11], Andrei [12–14], Zhang [15], Babaie-Kafaki et al. [16], and Shi and Guo [17].

Also, there is another class method with the descent direction $\mathbf{d}$ in $f(\mathbf{x}_k - \lambda\mathbf{d})$ being taken to be $\mathbf{D}\nabla f(\mathbf{x}_k)$, where $\mathbf{D}$ is a positive definite matrix that approximates the inverse of the Hessian matrix $\mathbf{A}$, which is usually named the quasi-Newton method. The earlier method of minimization of a nonlinear function by using this type approach is performed by Davidon [18], which was then simplified and reformulated by Fletcher and Powell [19] and was referred to as the variable metric method (DFP).

The remaining portions of this paper are arranged as follows. In Section 2 we describe an invariant manifold to derive the governing ordinary differential equations (ODEs). The main results are derived in Section 3, which includes the proof of convergence theorem, optimal parameter, optimal algorithm, a critical parameter, and a globally optimal algorithm. Then, in Section 4 we employ the rank-two Broyden-Fletcher-Goldfarb-Shanno (BFGS) updating techniques to update the Hessian matrix or its inversion, resulting in several novel optimal algorithms. The numerical examples are tested in Section 5 to assess the performance of the newly proposed algorithms. Finally, the conclusions are drawn in Section 6.

2. An Invariant Manifold

From this section on, we focus on the local minimum problem defined in terms of ϕ in (6), rather than that of f. When the novel algorithms are developed, we will return to the minimization problem (1). The present approach is different from the conventional line search method by minimizing the steplength λ_k in

$$f(\mathbf{x}_k - \lambda_k\mathbf{d}_k) = \min_{\lambda>0} f(\mathbf{x}_k - \lambda\mathbf{d}_k), \tag{7}$$

where $\mathbf{d}_k$ is a given search direction.

At the first, we consider an iterative scheme of $\mathbf{x}$ derived from the ordinary differential equations (ODEs) defined on an invariant manifold which is formed from $\phi(\mathbf{x})$:

$$h(\mathbf{x}, t) := Q(t)\,\phi(\mathbf{x}) = C. \tag{8}$$

Here, we let $\mathbf{x}$ be a function of a fictitious time variable t. We do not need to specify the function $Q(t)$ a priori, of which $C/Q(t)$ is merely a measure of the decreasing of ϕ in time. Hence, we expect that in our algorithm if $Q(t) > 0$ is an increasing function of t, the iterative point $\mathbf{x}_k$ can tend to the minimal point. We let $Q(0) = 1$, and C is determined from the initial condition $\mathbf{x}(0) = \mathbf{x}_0$ by

$$C = \phi(\mathbf{x}_0) > 0. \tag{9}$$

We can suitably choose the constant k_0 and hence c_0 in (6), such that $\phi(\mathbf{x}) > 0$. Indeed, the different level of ϕ does not alter its minimal point.

When $C > 0$ and $Q > 0$, the manifold defined by (8) is continuous, and thus the following differential operation being carried out on the manifold makes sense. For the requirement of consistency condition we have

$$\dot{Q}(t)\,\phi(\mathbf{x}) + Q(t)\,(\mathbf{A}\mathbf{x} - \mathbf{b}) \cdot \dot{\mathbf{x}} = 0, \tag{10}$$

which is obtained by taking the time differential of (8) with respect to t, using (6), and considering $\mathbf{x} = \mathbf{x}(t)$. We suppose that $\mathbf{x}$ is governed by the following ODEs:

$$\dot{\mathbf{x}} = -\kappa\mathbf{u}, \tag{11}$$

where κ is to be determined. Inserting (11) into (10) we can solve

$$\kappa = \frac{q(t)\,\phi}{\mathbf{g} \cdot \mathbf{u}}, \tag{12}$$

where

$$\mathbf{g} := \mathbf{A}\mathbf{x} - \mathbf{b}, \tag{13}$$

$$q(t) := \frac{\dot{Q}(t)}{Q(t)}. \tag{14}$$

We further suppose that

$$\mathbf{u} = \mathbf{u}_1 + \alpha\mathbf{u}_2 =: \mathbf{g} + \alpha\mathbf{B}\mathbf{g}, \tag{15}$$

where α is a parameter to be determined below through the solution of an optimal equation derived, and $\mathbf{B}$ is a descent matrix to be specified. Here, we assert that the driving vector $\mathbf{u}$ is an optimal linear combination of the gradient vector and a supplemental vector being the gradient vector times $\mathbf{B}$.

3. Numerical Methods

3.1. Convergence Theorem. Before the derivation of optimal algorithms we can prove the following convergence result.

Theorem 1. *For an iterative scheme to solve* (1) *by*

$$\mathbf{x}(t+\Delta t)=\mathbf{x}(t)-(1-\gamma)\frac{\mathbf{g}\cdot\mathbf{u}}{\mathbf{u}^{\mathrm{T}}\mathbf{A}\mathbf{u}}\mathbf{u}, \tag{16}$$

which is generated from the ODEs in (11)*, the iterative point* $\mathbf{x}$ *on the manifold* (8) *has the following convergence rate:*

$$\text{Convergence Rate} := \frac{Q(t+\Delta t)}{Q(t)}=\frac{1}{s}>1, \tag{17}$$

where

$$0<s=1-\frac{1-\gamma^2}{2a_0}<1, \tag{18}$$

$$a_0:=\frac{\phi\mathbf{u}^{\mathrm{T}}\mathbf{A}\mathbf{u}}{(\mathbf{g}\cdot\mathbf{u})^2}\geq\frac{1}{2}, \tag{19}$$

and $0\leq\gamma<1$ *is a relaxation parameter.*

Proof. The proof of this theorem is quite lengthy and we divide it into three parts.

(A) Inserting (12) into (11) we can obtain an evolution equation for $\mathbf{x}$:

$$\dot{\mathbf{x}}=-q(t)\frac{\phi}{\mathbf{g}\cdot\mathbf{u}}\mathbf{u}. \tag{20}$$

In the algorithm if $Q(t)$ can be guaranteed to be an increasing function of t, we might have an absolutely convergent property in finding the minimum of ϕ through the following equation:

$$\phi(t)=\frac{C}{Q(t)}, \tag{21}$$

which is obtained from (8). Here we simplify the notation of $\phi(\mathbf{x}(t))$ to $\phi(t)$.

(B) By applying the Euler method to (20) we can obtain the following algorithm:

$$\mathbf{x}(t+\Delta t)=\mathbf{x}(t)-\beta\frac{\phi}{\mathbf{g}\cdot\mathbf{u}}\mathbf{u}, \tag{22}$$

where

$$\beta=q(t)\Delta t. \tag{23}$$

In order to keep $\mathbf{x}$ on the manifold defined by (21) we can insert the above $\mathbf{x}(t+\Delta t)$ into

$$\frac{1}{2}\mathbf{x}^{\mathrm{T}}(t+\Delta t)\mathbf{A}\mathbf{x}(t+\Delta t)-\mathbf{b}^{\mathrm{T}}\mathbf{x}(t+\Delta t)=\frac{C}{Q(t+\Delta t)}-c_0, \tag{24}$$

and obtain

$$\begin{aligned}\frac{C}{Q(t+\Delta t)}-c_0=&\frac{1}{2}\mathbf{x}^{\mathrm{T}}(t)\mathbf{A}\mathbf{x}(t)-\mathbf{b}^{\mathrm{T}}\mathbf{x}(t)\\&+\beta\phi\frac{[\mathbf{b}-\mathbf{A}\mathbf{x}(t)]^{\mathrm{T}}\mathbf{u}}{\mathbf{g}\cdot\mathbf{u}}\\&+\beta^2\phi^2\frac{\mathbf{u}^{\mathrm{T}}\mathbf{A}\mathbf{u}}{2(\mathbf{g}\cdot\mathbf{u})^2}.\end{aligned} \tag{25}$$

Thus, by (13), (21), and (6) and through some manipulations we can derive the following scalar equation:

$$\frac{1}{2}a_0\beta^2-\beta+1=\frac{Q(t)}{Q(t+\Delta t)}, \tag{26}$$

where

$$a_0:=\frac{\phi\mathbf{u}^{\mathrm{T}}\mathbf{A}\mathbf{u}}{(\mathbf{g}\cdot\mathbf{u})^2}\geq\frac{1}{2}, \tag{27}$$

of which the inequality can be achieved by taking a suitable value of k_0 and hence c_0 in (6).

(C) Let

$$s:=\frac{Q(t)}{Q(t+\Delta t)}, \tag{28}$$

and by (26) we can derive

$$\frac{1}{2}a_0\beta^2-\beta+1-s=0. \tag{29}$$

From (29), we can take the solution of β to be

$$\beta=\frac{1-\sqrt{1-2(1-s)a_0}}{a_0},\quad \text{if } 1-2(1-s)a_0\geq 0. \tag{30}$$

Let

$$1-2(1-s)a_0=\gamma^2\geq 0, \tag{31}$$

$$s=1-\frac{1-\gamma^2}{2a_0}, \tag{32}$$

and the sufficient condition $1-2(1-s)a_0\geq 0$ in (30) is satisfied automatically, and thus by (30) and (31) we can obtain a preferred solution of β by

$$\beta=\frac{1-\gamma}{a_0}. \tag{33}$$

Here $0\leq\gamma<1$ is a relaxation parameter. The inequality $0<s<1$ follows from (32), $1-\gamma^2>0$, and $2a_0\geq 1$. Inserting the above β into (22) and using (19) we can derive algorithm (16). For this algorithm we can define the local convergence rate by

$$\text{Convergence Rate} := \frac{\phi(t)}{\phi(t+\Delta t)}, \tag{34}$$

which, using (28) and (21) and $0<s<1$ just proved, renders (17). This ends the proof of Theorem 1. □

3.2. Optimization of α. In Algorithm (22) we do not yet specify how to choose the parameter α. We can determine a suitable value of α such that s defined in (32) is minimized with respect to α, because a smaller s will lead to a faster convergence as shown by (17).

Thus by inserting (19) for a_0 into (32) we can write s to be

$$s = 1 - \frac{\left(1-\gamma^2\right)(\mathbf{g}\cdot\mathbf{u})^2}{2\phi\mathbf{u}\cdot(\mathbf{A}\mathbf{u})}, \tag{35}$$

where $\mathbf{u}$ as defined by (15) includes a parameter α. Let $\partial s/\partial\alpha = 0$, and through some algebraic operations we can solve α and denote it by

$$\alpha_o = \frac{\mathbf{g}\cdot\mathbf{u}_1\mathbf{u}_1\cdot(\mathbf{A}\mathbf{u}_2) - \mathbf{g}\cdot\mathbf{u}_2\mathbf{u}_1\cdot(\mathbf{A}\mathbf{u}_1)}{\mathbf{g}\cdot\mathbf{u}_2\mathbf{u}_1\cdot(\mathbf{A}\mathbf{u}_2) - \mathbf{g}\cdot\mathbf{u}_1\mathbf{u}_2\cdot(\mathbf{A}\mathbf{u}_2)}, \tag{36}$$

where the subscript o signifies that α_o is the optimal value of α.

Remark 2. For the usual three-dimensional vectors $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$, the following formula is famous:

$$\mathbf{a}\times(\mathbf{b}\times\mathbf{c}) = (\mathbf{a}\cdot\mathbf{c})\,\mathbf{b} - (\mathbf{a}\cdot\mathbf{b})\,\mathbf{c}. \tag{37}$$

Liu [20] has developed a Jordan algebra by extending the above formula to vectors in n-dimension:

$$[\mathbf{a},\mathbf{b},\mathbf{c}] = (\mathbf{a}\cdot\mathbf{b})\,\mathbf{c} - (\mathbf{c}\cdot\mathbf{b})\,\mathbf{a}, \quad \mathbf{a},\mathbf{b},\mathbf{c}\in\mathbb{R}^n. \tag{38}$$

In terms of the Jordan algebra we can write

$$\alpha_o = \frac{[\mathbf{u}_1,\mathbf{g},\mathbf{u}_2]\cdot(\mathbf{A}\mathbf{u}_1)}{[\mathbf{u}_2,\mathbf{g},\mathbf{u}_1]\cdot(\mathbf{A}\mathbf{u}_2)}, \tag{39}$$

where the symmetry of $\mathbf{A}$ was used. It can be seen that the above equation is a more symmetric form than that in (36).

3.3. An Optimal Algorithm. Now we can let $\mathbf{x}_k$ denote the numerical value of $\mathbf{x}$ at the kth step and go back $\mathbf{g}$ to $\mathbf{g}_k$, $\mathbf{u}$ to $\mathbf{u}_k$, $\mathbf{A}$ to $\mathbf{A}_k$, and $\mathbf{B}$ to $\mathbf{B}_k$. Thus, by using (16) we can derive an iterative algorithm:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta\frac{\mathbf{g}_k\cdot\mathbf{u}_k}{\mathbf{u}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{u}_k}\mathbf{u}_k, \tag{40}$$

where

$$\eta = 1-\gamma. \tag{41}$$

Therefore, we have the following optimal algorithm (OA).

(i) Select $0\le\gamma<1$, and give an initial $\mathbf{x}_0$.

(ii) For $k = 0, 1, 2, \ldots$, we repeat the following iterations:

$$\begin{aligned}
\alpha_k &= \frac{[\mathbf{g}_k,\mathbf{g}_k,\mathbf{B}_k\mathbf{g}_k]\cdot(\mathbf{A}_k\mathbf{g}_k)}{[\mathbf{B}_k\mathbf{g}_k,\mathbf{g}_k,\mathbf{g}_k]\cdot(\mathbf{A}_k\mathbf{B}_k\mathbf{g}_k)},\\
\mathbf{u}_k &= \mathbf{g}_k + \alpha_k\mathbf{B}_k\mathbf{g}_k,\\
\mathbf{x}_{k+1} &= \mathbf{x}_k - (1-\gamma)\frac{\mathbf{g}_k\cdot\mathbf{u}_k}{\mathbf{u}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{u}_k}\mathbf{u}_k.
\end{aligned} \tag{42}$$

If $\|\mathbf{g}_{k+1}\| < \varepsilon$, then stop; otherwise, go to step (ii).

Again we emphasize that we need to solve $\nabla f = \mathbf{0}$ for the minimization problem (1), and hence the residual means the value of $\|\nabla f\| = \|\mathbf{g}\|$. The above convergence criterion $\|\mathbf{g}_{k+1}\| < \varepsilon$ means that when the residual norm is smaller than a given error tolerance ε, the iterations are terminated.

3.4. A Critical Value for α. In Sections 3.2 and 3.3 we have used $\partial s/\partial\alpha = 0$ (or equivalently, $\partial a_0/\partial\alpha = 0$) to find the optimal value of α in the descent vector $\mathbf{u} = \mathbf{g} + \alpha\mathbf{B}\mathbf{g}$. Usually, this value of α obtained from $\partial s/\partial\alpha = 0$ is not the global minimum of a_0 (or s). Here, we try another approach and attempt to derive a better value of α than α_o, such that the value of α obtained in this manner is the global minimum of a_0 (or s).

In practice, we can take

$$a_0 := \frac{\phi\mathbf{u}^{\mathrm{T}}\mathbf{A}\mathbf{u}}{(\mathbf{g}\cdot\mathbf{u})^2} = a_s. \tag{43}$$

When a_s is near to 0.5, the convergence speed is very fast. Inserting (15) for $\mathbf{u}$ into the above equation and through some elementary operations we can derive a quadratic equation to solve α:

$$e_1\alpha^2 + e_2\alpha + e_3 = 0, \tag{44}$$

where

$$e_1 := \phi\mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - a_s(\mathbf{g}\cdot\mathbf{u}_2)^2, \tag{45}$$

$$e_2 := 2\phi\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - 2a_s\mathbf{g}\cdot\mathbf{u}_1\mathbf{g}\cdot\mathbf{u}_2, \tag{46}$$

$$e_3 := \phi\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_1 - a_s(\mathbf{g}\cdot\mathbf{u}_1)^2. \tag{47}$$

If the following condition is satisfied:

$$D := e_2^2 - 4e_1e_3 \ge 0, \tag{48}$$

then α in (44) has a *real solution*:

$$\alpha = \frac{\sqrt{D} - e_2}{2e_1}. \tag{49}$$

Inserting (45)–(47) into the *critical equation*:

$$D = e_2^2 - 4e_1e_3 = 0, \tag{50}$$

we can derive an algebraic equation to determine that a_s is the lowest bound of (48). *In this lowest bound a_s is a critical value denoted by a_c*, and for all $a_s \ge a_c$ it can satisfy (48) automatically. From (50) through some elementary operations, the critical value a_c can be solved as

$$a_c = \frac{\phi\left[\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_1\mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - \left(\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2\right)^2\right]}{\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_1(\mathbf{g}\cdot\mathbf{u}_2)^2 + \mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2(\mathbf{g}\cdot\mathbf{u}_1)^2 - 2\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2\mathbf{g}\cdot\mathbf{u}_1\mathbf{g}\cdot\mathbf{u}_2}. \tag{51}$$

Then by inserting it for a_s into (45) and (46) we can obtain a *critical value* α_c for α from (49):

$$\alpha_c = \frac{a_c \mathbf{g}\cdot\mathbf{u}_1 \mathbf{g}\cdot\mathbf{u}_2 - \phi \mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2}{\phi \mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - a_c(\mathbf{g}\cdot\mathbf{u}_2)^2}, \tag{52}$$

where $D = 0$ was used in view of (50).

By inserting (51) for a_c into (52) and cancelling out the common term ϕ we can derive a final equation for α_c, of which we use the same symbols for saving notations:

$$a_c = \frac{\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_1 \mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - \left(\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2\right)^2}{\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_1(\mathbf{g}\cdot\mathbf{u}_2)^2 + \mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2(\mathbf{g}\cdot\mathbf{u}_1)^2 - 2\mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 \mathbf{g}\cdot\mathbf{u}_1 \mathbf{g}\cdot\mathbf{u}_2}, \tag{53}$$

$$\alpha_c = \frac{a_c \mathbf{g}\cdot\mathbf{u}_1 \mathbf{g}\cdot\mathbf{u}_2 - \mathbf{u}_1^{\mathrm{T}}\mathbf{A}\mathbf{u}_2}{\mathbf{u}_2^{\mathrm{T}}\mathbf{A}\mathbf{u}_2 - a_c(\mathbf{g}\cdot\mathbf{u}_2)^2}. \tag{54}$$

Here we must emphasize that, in the current descent vector $\mathbf{u} = \mathbf{g} + \alpha\mathbf{B}\mathbf{g}$, the above value α_c is the *best one*, and the vector

$$\mathbf{u} = \mathbf{g} + \alpha_c \mathbf{B}\mathbf{g} \quad \text{(best descent vector)} \tag{55}$$

is the *best descent vector. Due to its criticality, if one attempts to find a better value of the parameter* α *than* α_c*, there would be no real solution of* α. Furthermore, the best descent vector is also better than the optimal vector $\mathbf{u} = \mathbf{g} + \alpha_o \mathbf{B}\mathbf{g}$ derived in Section 3.2.

3.5. A Globally Optimal Algorithm. Then, we can derive the following *globally optimal algorithm* (GOA) to solve the minimization problem in (1).

(i) Select $0 \le \gamma < 1$ and give an initial guess of $\mathbf{x}_0$.

(ii) For $k = 0, 1, 2, \ldots$, we repeat the following iterations:

$$\mathbf{u}_1^k = \mathbf{g}_k,$$

$$\mathbf{u}_2^k = \mathbf{B}_k \mathbf{g}_k,$$

$$\begin{aligned} a_c^k &= \left(\mathbf{u}_1^k\cdot\mathbf{A}_k\mathbf{u}_1^k \mathbf{u}_2^k\cdot\mathbf{A}_k\mathbf{u}_2^k - \left(\mathbf{u}_1^k\cdot\mathbf{A}_k\mathbf{u}_2^k\right)^2\right) \\ &\quad\times\left(\mathbf{u}_1^k\cdot\mathbf{A}_k\mathbf{u}_1^k\left(\mathbf{g}_k\cdot\mathbf{u}_2^k\right)^2 + \mathbf{u}_2^k\cdot\mathbf{A}_k\mathbf{u}_2^k\left(\mathbf{g}_k\cdot\mathbf{u}_1^k\right)^2\right. \\ &\quad\left.-2\mathbf{u}_1^k\cdot\mathbf{A}_k\mathbf{u}_2^k \mathbf{g}_k\cdot\mathbf{u}_1^k \mathbf{g}_k\cdot\mathbf{u}_2^k\right)^{-1}, \end{aligned} \tag{56}$$

$$\alpha_k = \frac{a_c^k \mathbf{g}_k\cdot\mathbf{u}_1^k \mathbf{g}_k\cdot\mathbf{u}_2^k - \mathbf{u}_1^k\cdot\mathbf{A}_k\mathbf{u}_2^k}{\mathbf{u}_2^k\cdot\mathbf{A}_k\mathbf{u}_2^k - a_c^k(\mathbf{g}_k\cdot\mathbf{u}_2^k)^2},$$

$$\left(\text{critical optimal parameter}\right),$$

$$\mathbf{u}_k = \mathbf{g}_k + \alpha_k \mathbf{B}_k \mathbf{g}_k, \quad \left(\text{best descent vector}\right),$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (1-\gamma)\frac{\mathbf{g}_k\cdot\mathbf{u}_k}{\mathbf{u}_k\mathbf{A}_k\mathbf{u}_k}\mathbf{u}_k. \tag{57}$$

If $\mathbf{x}_{k+1}$ converges according to a given stopping criterion $\|\mathbf{g}_{k+1}\| < \varepsilon$, then stop; otherwise, go to step (ii).

Remark 3. We have derived a *novel globally optimal algorithm* for solving the minimization problem in (1). In terms of the descent vector $\mathbf{u} = \mathbf{g} + \alpha_c \mathbf{B}\mathbf{g}$, the GOA is the best one, which leads to the *global minimum* of a_0 (or s) and hence the *largest convergence rate*. While the parameter γ is chosen by the user with problem dependence, the parameter α_k is exactly given by (56). Up to here we have successfully derived a novel best descent vector algorithm, with the help from (19), (53), and (54).

Remark 4. At the very beginning we have set an invariant manifold $h(\mathbf{x}, t) = Q(t)\phi(\mathbf{x})$ in (8) as our starting point to derive the iterative optimal algorithms, which includes a locally objective function ϕ in the governing equations. However, in the final stage the terms which include ϕ can be cancelled out, and thus we have obtained the optimal algorithm in (42) and the globally optimal algorithm in (57), which are both independent of ϕ.

4. The Broyden-Fletcher-Goldfarb-Shanno Updating Techniques

In the above we have derived two optimal algorithms by leaving the descent matrix $\mathbf{B}$ to be specified by the user. First we fix $\mathbf{B}$ to be the exact Hessian matrix $\mathbf{A}$, and then we can obtain two optimal algorithms OA and GOA.

We can also apply the technique of BFGS by updating the Hessian matrix $\mathbf{A}$ and its inverse matrix $\mathbf{B}$. To derive this updating technique let us mention the Newton iterative scheme to solve (1):

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \lambda_k \nabla^2 f(\mathbf{x}_k)^{-1}\mathbf{g}_k, \tag{58}$$

where λ_k is the optimal steplength along the Newton direction $\nabla^2 f(\mathbf{x}_k)^{-1}\mathbf{g}_k$ at the kth step. In order to construct a matrix $\mathbf{B}_k$ to approximate $\nabla^2 f(\mathbf{x}_k)^{-1}$ we can analyze the relation between $\nabla^2 f(\mathbf{x}_k)^{-1}$ and the first order derivative $\mathbf{g}_k$. We take the Taylor expansion of $f(\mathbf{x})$ at a point $\mathbf{x}_{k+1}$, obtaining

$$\begin{aligned} f(\mathbf{x}) \approx{}& f(\mathbf{x}_{k+1}) + \nabla f(\mathbf{x}_{k+1})^{\mathrm{T}}(\mathbf{x} - \mathbf{x}_{k+1}) \\ &+ \frac{1}{2}(\mathbf{x} - \mathbf{x}_{k+1})^{\mathrm{T}}\nabla^2 f(\mathbf{x}_{k+1})(\mathbf{x} - \mathbf{x}_{k+1}). \end{aligned} \tag{59}$$

Then we have

$$\nabla f(\mathbf{x}_k) \approx \nabla f(\mathbf{x}_{k+1}) + \nabla^2 f(\mathbf{x}_{k+1})(\mathbf{x}_k - \mathbf{x}_{k+1}). \tag{60}$$

Let

$$\mathbf{p}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \tag{61}$$

$$\mathbf{q}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k), \tag{62}$$

and from (60) we have

$$\mathbf{q}_k \approx \nabla^2 f(\mathbf{x}_{k+1})\mathbf{p}_k. \tag{63}$$

Assume that the Hessian matrix $\nabla^2 f(\mathbf{x}_{k+1})$ is invertible and denote the inverse matrix by $\mathbf{B}_{k+1}$. Then from the above equation we have the so-called quasi-Newton condition:

$$\mathbf{p}_k = \mathbf{B}_{k+1}\mathbf{q}_k. \tag{64}$$

When we take the descent matrix $\mathbf{B}$ to be the inverse of the Hessian matrix $\mathbf{A}$, we might accelerate the convergence speed, which is however a more difficult task when the dimension of the Hessian matrix is quite large. So far, as that done in the quasi-Newton method, we can employ the following updating technique due to BFGS:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \left(1 + \frac{\mathbf{q}_k^{\mathrm{T}}\mathbf{B}_k\mathbf{q}_k}{\mathbf{q}_k \cdot \mathbf{p}_k}\right)\frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{q}_k \cdot \mathbf{p}_k} - \frac{\mathbf{p}_k\mathbf{q}_k^{\mathrm{T}}\mathbf{B}_k + \mathbf{B}_k\mathbf{q}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{q}_k \cdot \mathbf{p}_k}. \tag{65}$$

The advantage by using the above updating technique is that we can obtain an approximation of the inverse of the Hessian matrix, and we do not need to really calculate $\mathbf{A}^{-1}$.

By the same token we can also apply the technique of BFGS to update the Hessian matrix without needing the computation of the real Hessian matrix:

$$\mathbf{A}_{k+1} = \mathbf{A}_k + \frac{\mathbf{q}_k\mathbf{q}_k^{\mathrm{T}}}{\mathbf{q}_k \cdot \mathbf{p}_k} - \frac{\mathbf{A}_k\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}\mathbf{A}_k}{\mathbf{p}_k \cdot (\mathbf{A}_k\mathbf{p}_k)}, \tag{66}$$

where at the first step we can take $\mathbf{A}_0 = \mathbf{I}_n$. The advantage of this approach is that we do not need to calculate the Hessian matrix exactly, as developed independently by Broyden [21], Fletcher [22], Goldfarb [23], and Shanno [24]. It is easy to check that the updating technique in (66) satisfies the quasi-Newton condition:

$$\mathbf{A}_{k+1}\mathbf{p}_k = \mathbf{q}_k, \tag{67}$$

which is an inversion relationship of (64).

The present notations of $\mathbf{A}$ and $\mathbf{B}$ are the same as those $\mathbf{B}$ and $\mathbf{H}$ used by Dai [25, 26].

The above technique together with the OA by using $\mathbf{B}_k = \mathbf{A}_k$ generated from (66) will be named the OABFGS, and two numerical examples will be given to test its performance.

When the matrix $\mathbf{A}$ is given exactly by the Hessian matrix in the OA and GOA, we use the above technique to compute $\mathbf{B}_k$ in the OA and GOA, and the resultant iterative algorithms are named OABFGS1 and GOABFGS1, respectively. Finally, if in the OA and GOA both $\mathbf{A}$ and $\mathbf{B}$ are updated by (66) and (65), the resultant iterative algorithms will be named OABFGS2 and GOABFGS2, respectively.

In order to show the high performance of the optimal algorithms OABFGS1, GOABFGS1, and OABFGS2, we will also apply the DFP method [18, 19] to solve nonlinear minimization problems, of which the updating technique is

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{q}_k \cdot \mathbf{p}_k} - \frac{\mathbf{B}_k\mathbf{q}_k\mathbf{q}_k^{\mathrm{T}}\mathbf{B}_k}{\mathbf{q}_k \cdot (\mathbf{B}_k\mathbf{q}_k)}. \tag{68}$$

For each iterative step we search the minimization of $f(\mathbf{x}_k - \lambda\mathbf{B}_k\nabla f(\mathbf{x}_k))$ to find λ by solving the optimality equation $\nabla f(\mathbf{x}_k - \lambda\mathbf{B}_k\nabla f(\mathbf{x}_k)) \cdot (\mathbf{B}_k\nabla f(\mathbf{x}_k)) = 0$ by using the half-interval method. Gill et al. [27], among several others, have shown that the modification in (65) performs more efficiently than that in (68) for most problems.

5. Numerical Examples

In order to evaluate the performance of the newly developed methods let us investigate the following examples. Some results are compared with those obtained from the steepest descent method (SDM) and the DFP method [18, 19]. In the above algorithms there exists a relaxation parameter γ, which is problem dependent. A good parameter value of γ can be selected easily by comparing the convergence speeds for different values of γ.

Example 1. As the first testing example of OA and GOA we use the following function given by Rosenbrock [28]:

$$\min\left\{f = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2\right\}. \tag{69}$$

In mathematical optimization, the Rosenbrock function is a nonconvex function used as a performance test benchmark problem for optimization algorithms. It is also known as Rosenbrock's valley or Rosenbrock's banana function. The minimum is zero occurring at $(x_1, x_2) = (1, 1)$. This function is difficult to be minimized because it has a steep sided valley following the parabolic curve $x_1^2 = x_2$. Kuo et al. [29] have used the particle swarm method to solve this problem; however, the numerical procedures are rather complex. Liu and Atluri [30] have applied a fictitious time integration method to solve the above problem under the constraints of $x_1 \geq 0$ and $x_2 \geq 0$, whose accuracy can reach to the fifth order.

We apply the OA to this problem by starting from the initial point at $(3, 2)$ and under a convergence criterion $\varepsilon = 10^{-10}$. The SDM is run 3749 steps as shown in Figures 1(a) and 1(b) by solid lines for showing the residual and the objective function f. The SDM can reach a very accurate minimum value of f with 1.22×10^{-20}. The OA with $\gamma = 0$ converges very fast with only 6 steps, with the residual and the objective function f being shown in Figures 1(a) and 1(b) by dashed lines. The OA is faster than the SDM with about 625 times, and furthermore the minimum value of f can be reduced to 1.925×10^{-25}. In Figure 1(c) we compare the solution paths generated by the SDM and OA. When the SDM is moving very slowly along the valley, the OA is moving outside the region of valley and is convergent very fast to the solution. In Figure 1(c) the red dashed line is used to represent the valley of the Rosenbrock function, which is not used to indicate the result of a numerical method. As shown in Figure 1(b), GOA is slightly better than OA, although the paths of OA and GOA seem to be identical in Figure 1(c).

The SDM usually works very well during early stages; however, as a stationary point is approached, it behaves poorly, taking small nearly orthogonal steps. On the other hand, with the help of the optimal direction $\mathbf{g} + \alpha_o\mathbf{A}\mathbf{g}$ the OA can fast reach to the final end point with high accuracy. For this example the GOA spends the same number of steps as the OA; however, the GOA gives very accurate minimum value of $f = 1.26 \times 10^{-29}$.

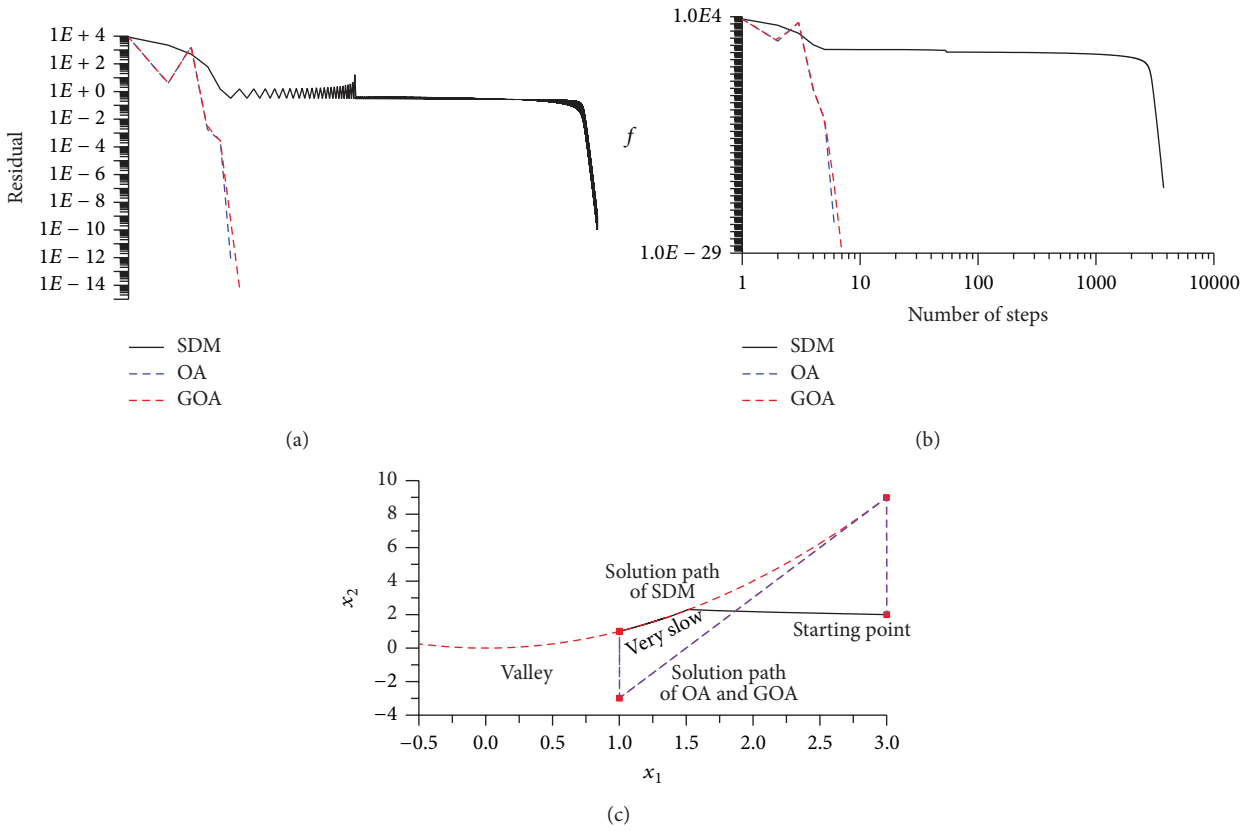

FIGURE 1: For the Rosenbrock problem, (a) the residuals, (b) the objective function f, and (c) the solution paths of SDM, OA, and GOA.

Example 2. Next, we consider a generalization of the Rosenbrock function [31, 32]:

$$\min\left\{f=\sum_{k=1}^{n-1}\left[100\left(x_{k+1}-x_k^2\right)^2+\left(1-x_k\right)^2\right]\right\}. \quad (70)$$

This variant has been shown to have exactly one minimum for $n = 3$ at $(x_1, x_2, x_3) = (1, 1, 1)$ and exactly two minima for $4 \le n \le 7$. The global minimum happens at $(x_1, x_2, \ldots, x_n) = (1, 1, \ldots, 1)$ and a local minimum is near $(x_1, x_2, \ldots, x_n) = (-1, 1, \ldots, 1)$. This result is obtained by setting the gradient of the function equal to zero, noticing that the resulting equation is a rational function of x_i. For small n the polynomials can be determined exactly and Sturm's theorem can be used to determine the number of real roots, while the roots can be bounded in the region of $|x_i| < 2.4$ [32]. For larger n this method breaks down due to the size of the coefficients involved.

We apply the OA to this problem with $n = 30$, starting from $x_i = 0.1i$, and under a convergence criterion $\varepsilon = 10^{-5}$. Under the above stopping criterion, the SDM is run over 13830 steps as shown in Figure 2(a) by solid lines for showing the residual and f. The SDM can reach a very accurate minimum value of f with 1.357×10^{-10}. The OA with $\gamma = 0.2$ converges with 9956 steps, with the residual and f being shown in Figure 2 by dashed lines. The OA is faster than the SDM, and similarly the minimum of f can be reduced to 1.96×10^{-11}.

At the same time we show the values of the optimal α in Figure 3(a) for the OA. The value of α is much smaller than 1, which means that the term $\mathbf{g}$ plays a dominant role in the descent direction; however, the influence of $\alpha\mathbf{Ag}$, although a little, is a key point to speed up the convergence.

Then we apply the GOA to this problem under the same conditions as that in OA. The GOA with $\gamma = 0.1$ converges with 9846 steps, with the residual and f being shown in Figure 2 by dashed-dotted lines. The GOA is faster than the SDM, and the minimum value of f can be further reduced to 9.59×10^{-12}. We show the values of the optimal α for the GOA in Figure 3(b). This problem is quite difficult, and many other algorithms are failure to solve this problem.

Example 3. We consider a problem due to Powell [33]:

$$\min\left\{f=\left(x_1+10x_2\right)^2+5\left(x_3-x_4\right)^2 +\left(x_2-2x_3\right)^4+10\left(x_1-x_4\right)^4\right\}. \quad (71)$$

The minimum of f is zero occurring at $(x_1, x_2, x_3, x_4) = (0, 0, 0, 0)$. We apply the OA to this problem, starting from $(x_1, x_2, x_3, x_4) = (3, -1, 0, 1)$ and under a convergence criterion of $\varepsilon = 10^{-6}$. The SDM converges very slowly over

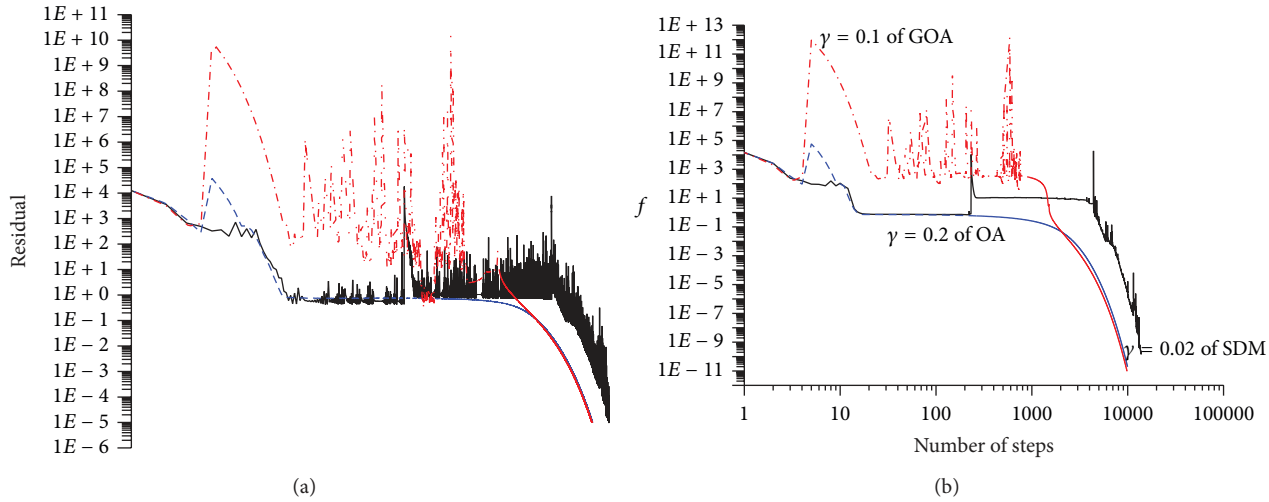

FIGURE 2: For the generalized Rosenbrock problem, (a) the residuals and (b) the objective functions obtained by SDM, OA, and GOA.

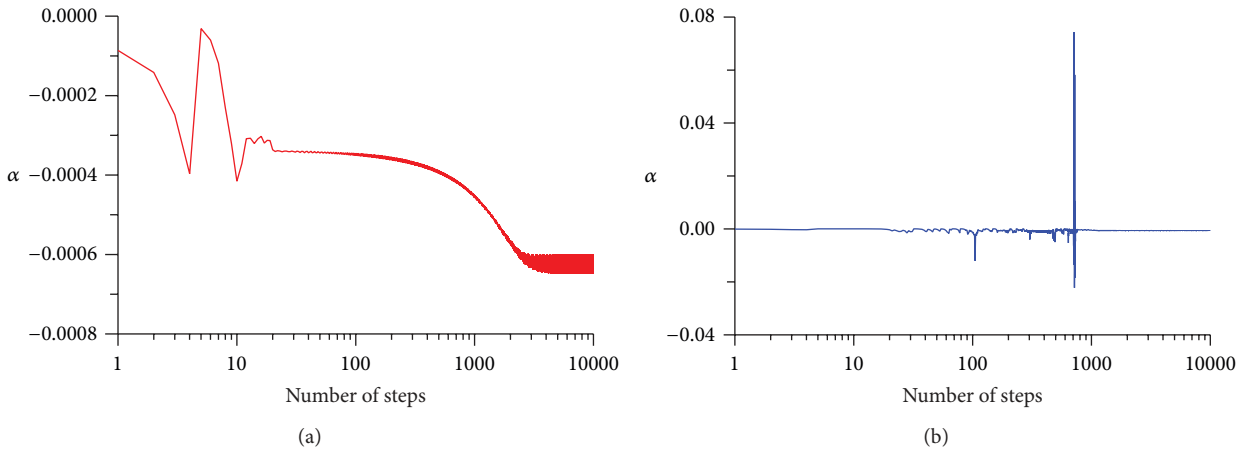

FIGURE 3: In the solution of the generalized Rosenbrock problem by the OA and GOA, showing the variation of optimal values of α for (a) OA and (b) GOA.

50,000 steps as shown in Figure 4 by solid lines for showing a_0, s, and the residual. The SDM can reach a very accurate value of $f = 9.938 \times 10^{-9}$. At the same time, the OA with $\gamma = 0.15$ converges with 349 steps, with a_0, s, and the residual shown in Figure 4 by dashed lines. The OA is faster than the SDM over 130 times, and furthermore we can get a more accurate $f = 8.33 \times 10^{-10}$.

As shown in Figure 4(c) the residual obtained by the OABFGS is rapidly growing to the order 10^{26} and then fast decaying to the order 10^{-6}, and through 1043 steps the OABFGS leads to a better solution with $f = 1.285 \times 10^{-9}$ than that obtained by the SDM. Here the combination of the optimal algorithm with the BFGS updating technique is very useful when the exact Hessian matrix is not available or when the computation of the Hessian matrix is cumbersome.

Then we apply the GOA in Section 3.5 to the Powell problem. Under the same convergence criterion of $\varepsilon = 10^{-6}$, the GOA with $\gamma = 0.001$ converges only with 96 steps as shown in Figure 4(c), where the GOA is faster than the SDM with 500 times and than OA with 3.5 times, and furthermore we can get a very accurate minimum value $f = 1.55 \times 10^{-9}$. In Figure 5 we compare the values of α obtained by the OA and the GOA, of which we can observe that both α are quite small and may be negative.

Then, we apply the OABFGS1 and the GOABGGS1 mentioned in Section 4 to solve the Powell problem. Under the same convergence criterion of $\varepsilon = 10^{-6}$, the OABFGS1 with $\gamma = 0.1$ converges only with 29 steps as shown in Figure 6(a), where we can get very accurate value of $f = 7.57 \times 10^{-12}$. Similarly, the GOABFGS1 with $\gamma = 0.1$ converges with 30 steps as shown in Figure 6(a), where we can get very accurate value of $f = 5.28 \times 10^{-10}$. In Figure 6(b) we compare the values of α obtained by the OABFGS1 and GOABFGS1, of which we can observe that both α are quite

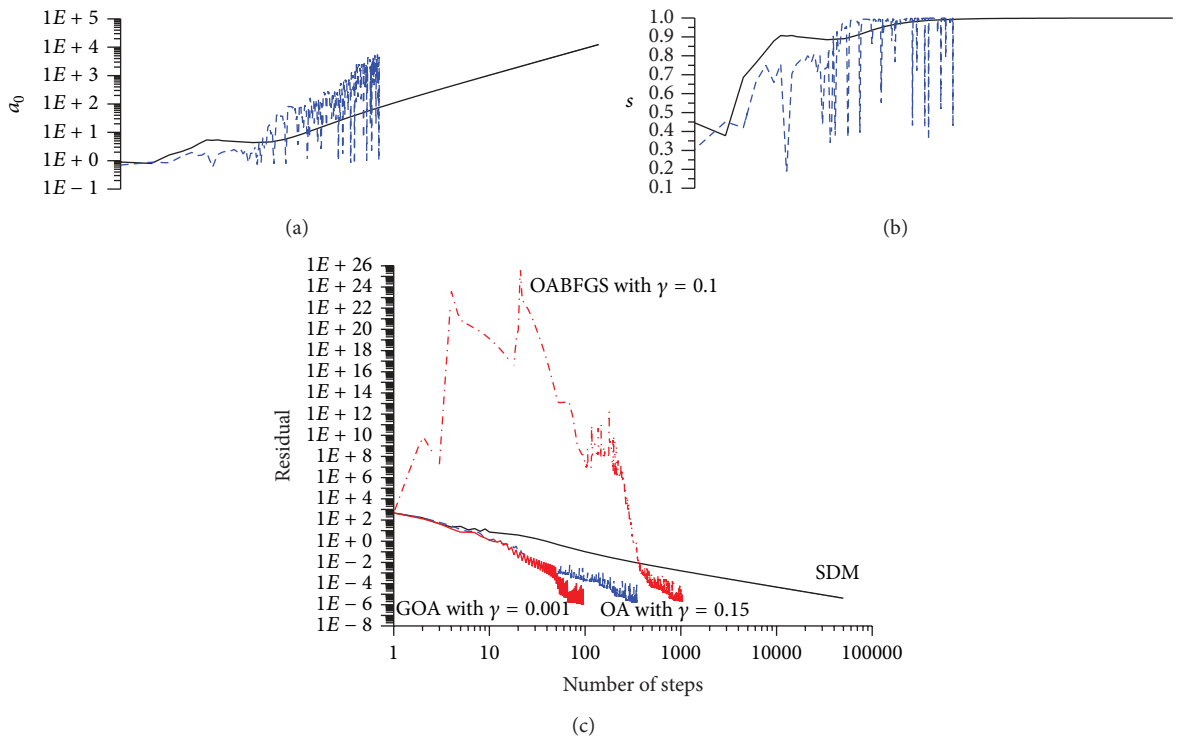

FIGURE 4: For the Powell problem, comparing (a) a_0 and (b) s of SDM and OA and (c) the residuals obtained by the SDM, OA, GOA, and OABFGS.

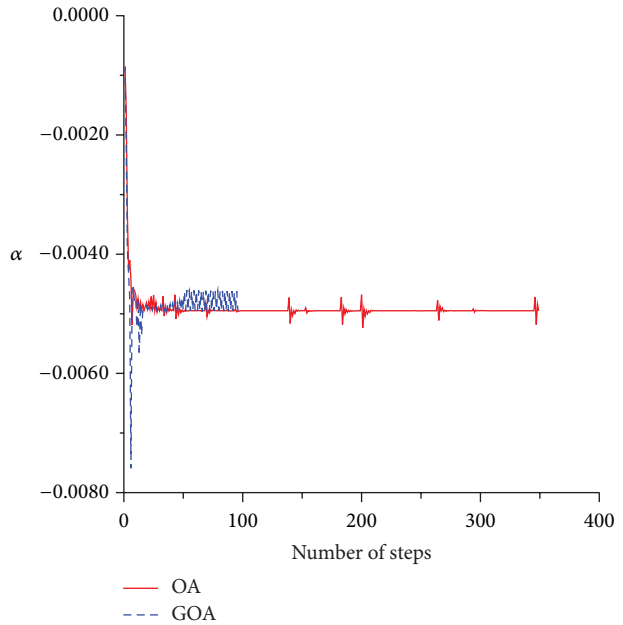

FIGURE 5: For the Powell problem, comparing α obtained by the OA and GOA.

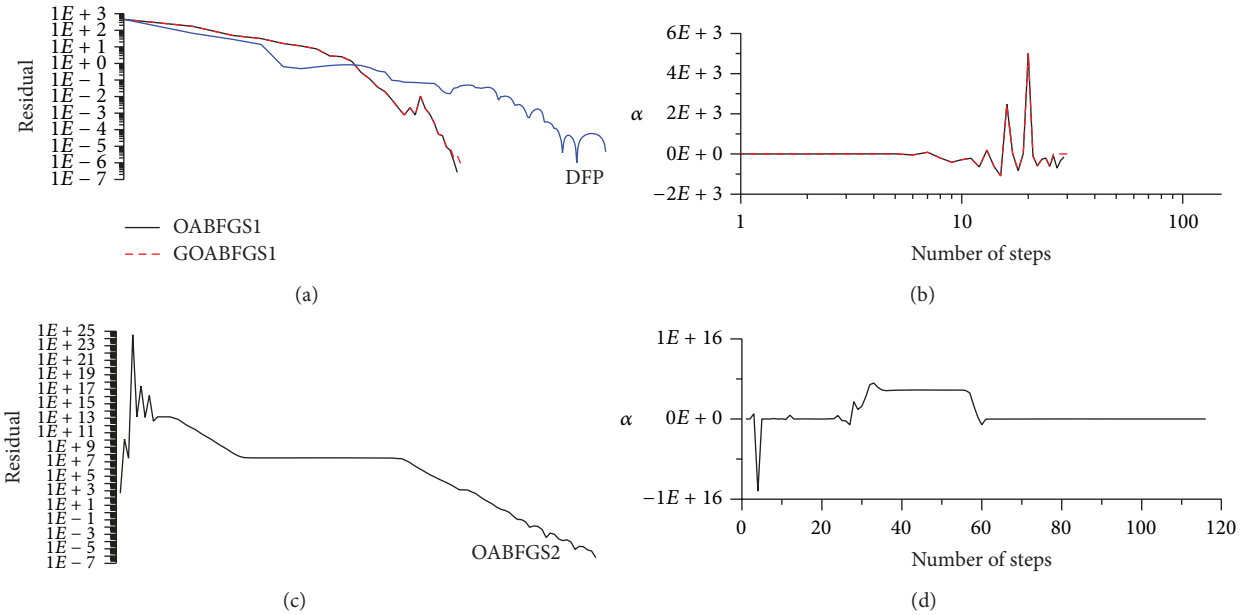

FIGURE 6: For the Powell problem, comparing (a) the residuals and (b) α obtained by the OABFGS1 and GOABFGS1 with DFP and showing (c) the residual and (d) α obtained by OABFGS2.

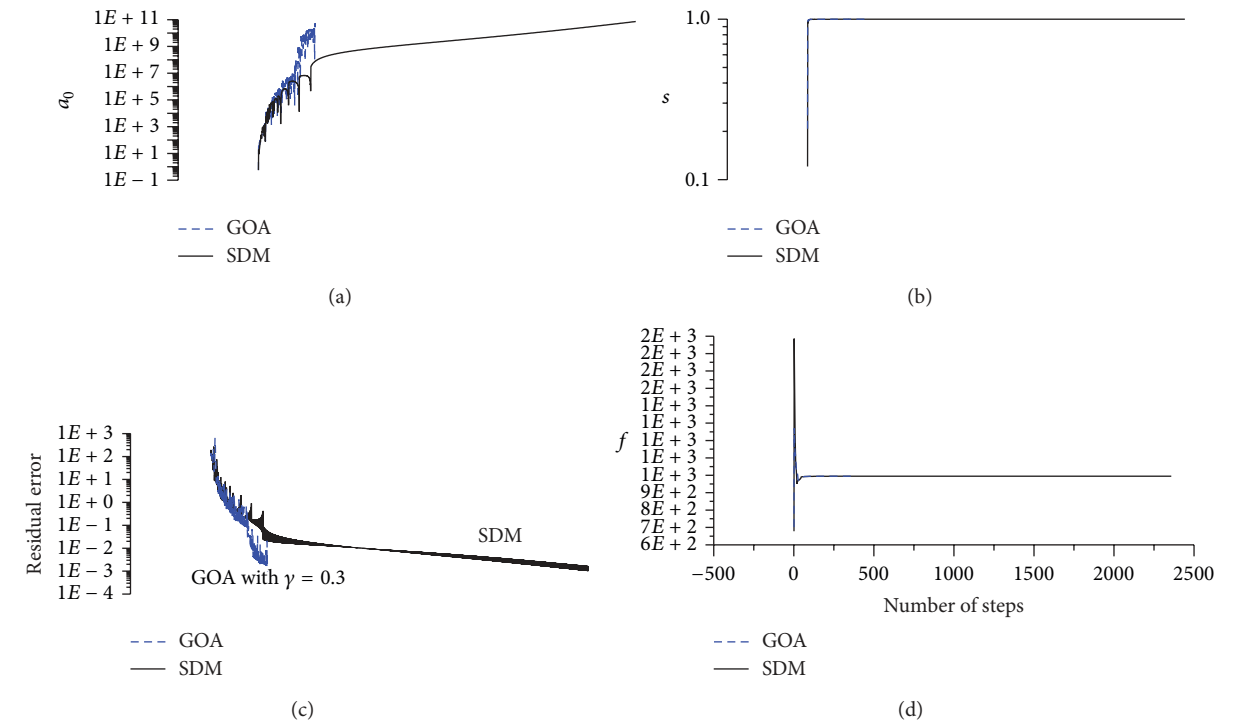

FIGURE 7: For the maximum area under a given curve comparing (a) a_0, (b) s, (c) residual error, and (d) f obtained by the SDM and GOA.

large, which means that the quasi-Newton direction is a dominant factor to accelerate the convergence speed.

Then, we apply the OABFGS2 to solve the Powell problem with $\gamma = 0$, which converges through 116 steps and with the value of f to be $f = 2.77 \times 10^{-11}$, which is very accurate. In Figures 6(c) and 6(d) we show the residual and the value of α obtained by the OABFGS2.

In order to reveal the high performance of the optimal algorithms OABFGS1, GOABFGS1, and OABFGS2, we apply the DFP method [18, 19] to solve the Powell problem, which converges through 131 steps as shown in Figure 6(a). For each iteration step we search the minimization of $f(\mathbf{x}_k - \lambda \mathbf{B}_k \nabla f(\mathbf{x}_k))$ to find λ by solving the optimality equation $\nabla f(\mathbf{x}_k - \lambda \mathbf{B}_k \nabla f(\mathbf{x}_k)) \cdot (\mathbf{B}_k \nabla f(\mathbf{x}_k)) = 0$ by using

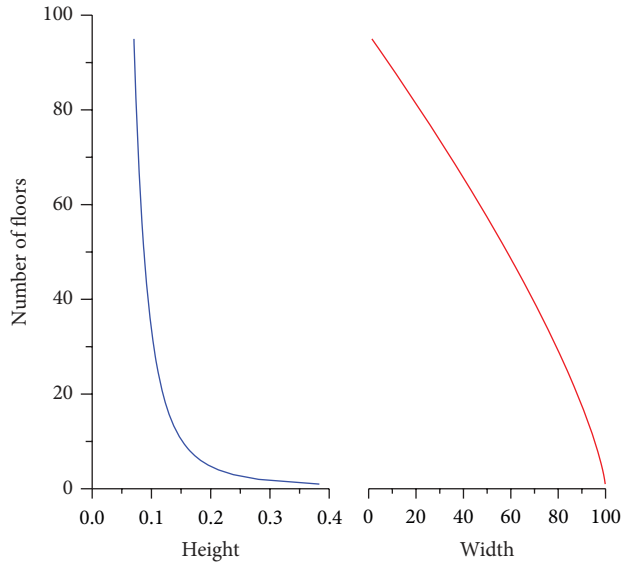

Figure 8: Showing the heights and widths of the floors with respect to the number of floors.

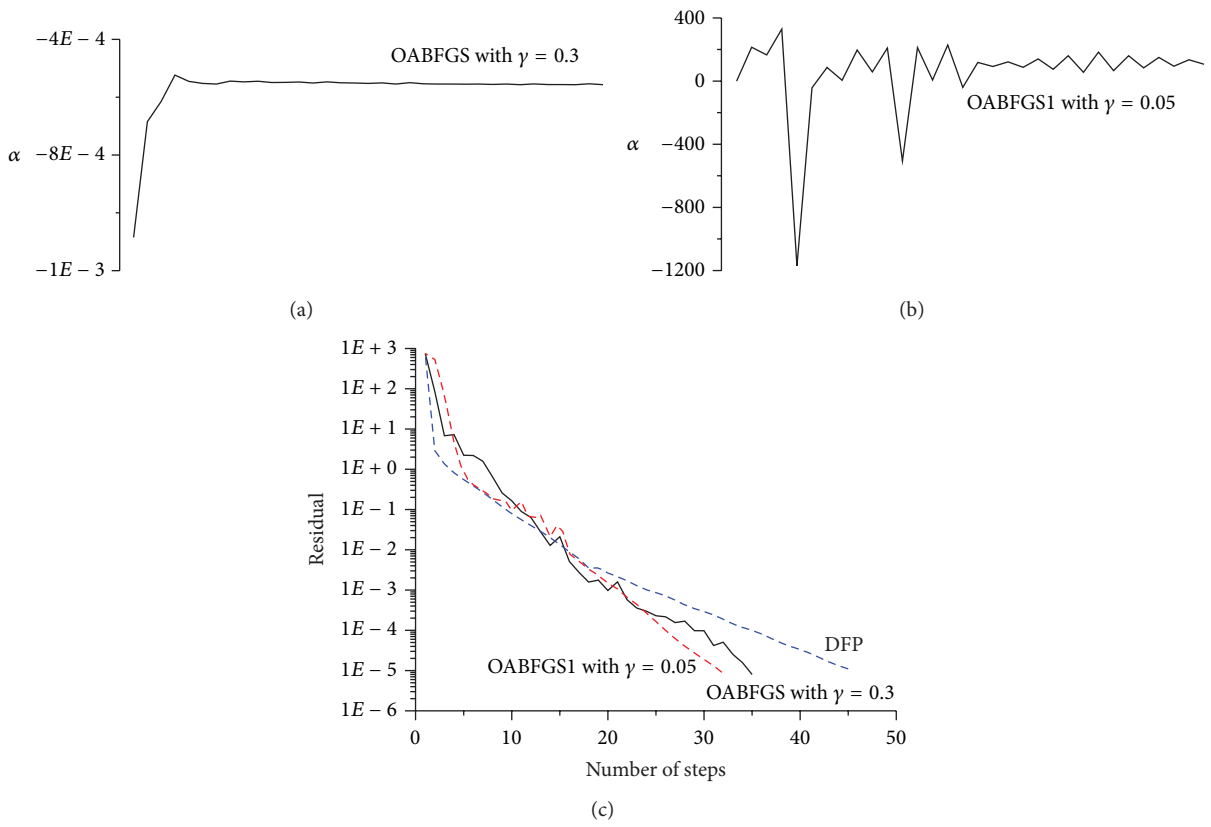

Figure 9: For Example 4, showing (a) α of OABFGS and (b) α of OABFGS1 and (c) comparing the residuals obtained by the DFP, OABFGS, and OABFGS1.

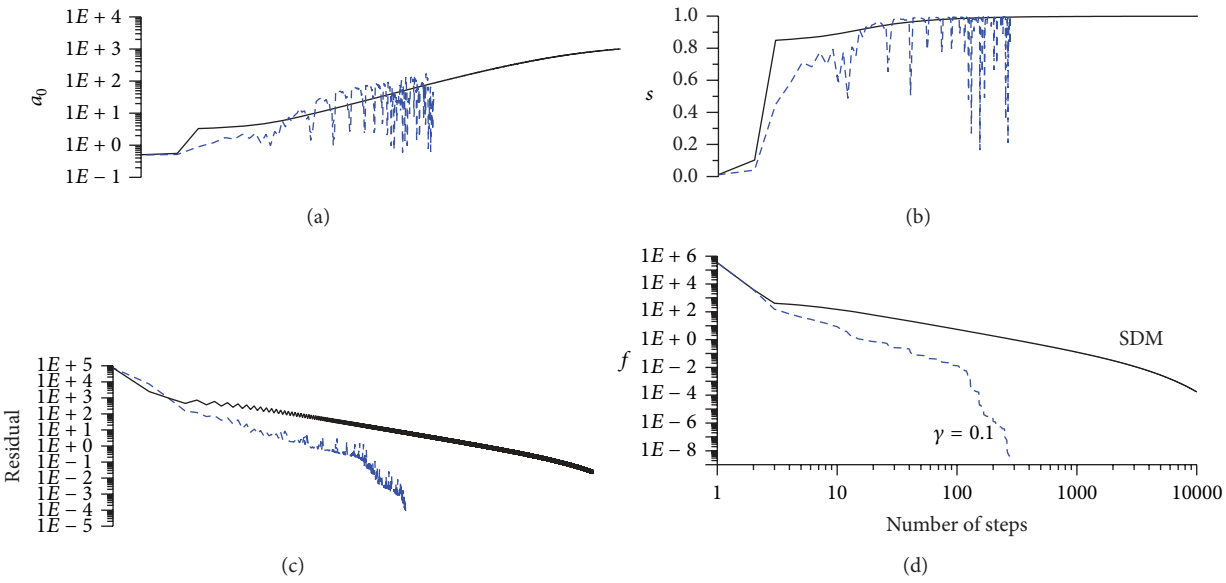

FIGURE 10: For the minimization of Schwefel function comparing (a) a_0, (b) s,(c) residual error, and (d) f obtained by the SDM and OA.

the half-interval method, with two end points fixed by $\lambda = 0$ and $\lambda = 1$, and the convergence criterion is given by $\varepsilon = 10^{-10}$. At the end of the iteration we obtain the minimum value of f to be $f = 9.79 \times 10^{-12}$, which is very accurate.

Example 4. In this example we design an office block inside a structure with a curved roof given by $x = 100 - y^2$. Suppose that the number of total cuboids is n and each cuboid can have different size, and we attempt to find the dimensions of all cuboids with maximum volume which would fit inside the given roof structure; that is,

$$\max \left\{ f = y_1 \left[100 - y_1^2\right] + y_2 \left[100 - \left(y_1 + y_2\right)^2\right] + \ldots + y_n \left[100 - \left(y_1 + \ldots + y_n\right)^2\right] \right\}, \tag{72}$$

where $y_i > 0$ is the height of the ith cuboid.

The maximum of f is tending to 2000/3 when n is increasing. When $n = 95$, we apply the GOA to this problem by starting from $y_i = 0.05$ and under a convergence criterion of $\varepsilon = 10^{-5}$. The SDM is convergent with 6868 steps as shown in Figure 7 by solid lines for showing a_0, s, residual, and f. At the same time, the GOA with $\gamma = 0.35$ converges with 96 steps, with a_0, s, residual, and f being shown in Figure 7 by dashed lines. Both f are tending to 661.9945. The GOA is faster than the SDM with 80 times. The heights and widths of the cuboids with respect to the number of floors are plotted in Figure 8. For this problem both OA and GOA lead to the same numerical results.

Moreover, we also apply the OABFGS and the OABFGS1 as well as the DFP to this problem under the above same initial condition and convergence criterion, where for the DFP we use the half-interval method to solve the local minimization to find the best λ with two end points fixed by $\lambda = 0$ and $\lambda = 1$, and the convergence criterion is given by $\varepsilon = 10^{-10}$. The values of γ used in the OABFGS and OABFGS1 are, respectively, 0.3 and 0.05. The residuals of these three methods are compared in Figure 9(c), from which we can observe that the OABFGS converges with 35 steps, the OABFGS1 converges with 32 steps, and the DFP converges with 46 steps. They are all better than the above OA and GOA algorithms. The value of α as shown in Figure 9(a) for the OABFGS is quite small, which indicates that the descent direction is dominated by the gradient direction. The value of α as shown in Figure 9(b) for the OABFGS1 is quite large, which indicates that the descent direction is dominated by the quasi-Newton direction.

Example 5. In this example we test the minimization of the Schwefel function with $n = 100$:

$$\min \left\{ f = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \right\}. \tag{73}$$

The minimum is zero at $x_j = 0$, $j = 1, \ldots, n$.

We apply the OA to this problem by starting from $x_i = 1$ and under a convergence criterion of $\varepsilon = 10^{-4}$. The SDM does not converge within 10000 steps as shown in Figure 10 by solid lines for showing a_0, s, residual, and f. At the same time, the OA with $\gamma = 0.1$ converges with 276 steps, with a_0, s, residual, and f being shown in Figure 10 by dashed lines. The OA is faster than the SDM over 100 times, and f is tending to 3.99×10^{-10} which is more accurate than 1.733×10^{-4} obtained by the SDM. When we apply the GOA with $\gamma = 0.05$ to this problem, it is convergence with 299 steps and is faster than the SDM over 100 times, and f is tending to 7.38×10^{-10} which is more accurate than SDM and OA.

Example 6. In this example we test the minimization of the Whitley function with $n = 8$:

$$\min \left\{ f = \sum_{i=1}^{n} \sum_{j=1}^{n} \left[\frac{y_{ji}^2}{4000} - \cos y_{ji} + 1 \right] \right\}, \tag{74}$$

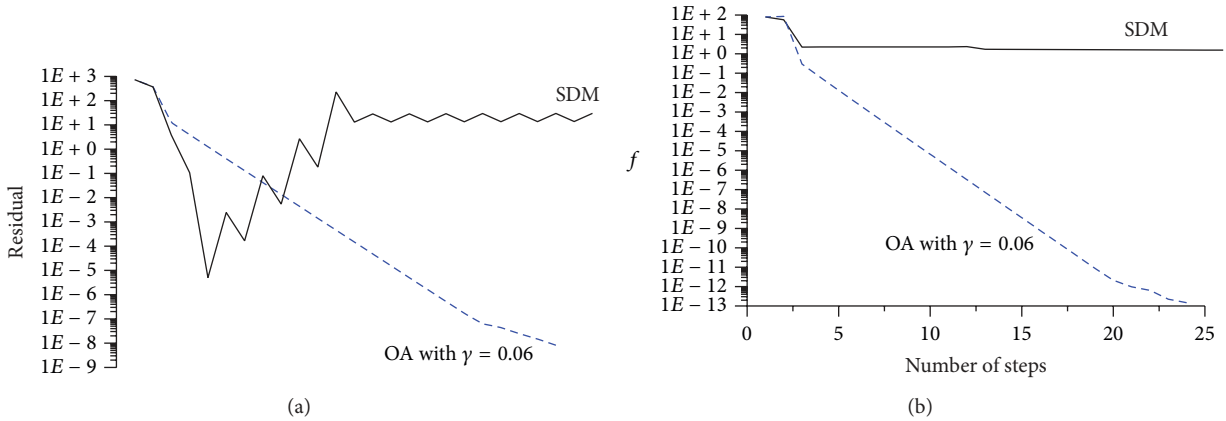

Figure 11: For the minimization of Whitley function comparing (a) residuals and (b) f obtained by the SDM and OA, where the SDM is failure.

where $y_{ji} = 100(x_i - x_j^2)^2 + (x_j - 1)^2$. The minimum is zero at $x_j = 1$, $j = 1, \ldots, n$. Here, we fix $n = 8$.

It is very difficult to minimize the Whitley function. The SDM diverges as shown in Figure 11 by solid lines. We apply the OA to this problem, starting from $x_i = 1.12$ and under a convergence criterion of $\varepsilon = 10^{-8}$. The OA with $\gamma = 0.06$ converges with 24 steps, with residual and f being shown in Figure 11 by dashed lines and f tending to 1.47×10^{-13}.

6. Conclusions

By formulating the minimization problem into a continuous manifold, we can derive a governing system of ODEs for deriving the iterative algorithms which being proven convergence to find the unknown minimum point $\mathbf{x}$ of a given nonlinear minimization function. The novel algorithm is named "an optimal algorithm (OA)," because in the local frame we have derived the optimal parameter of α in the descent direction, which is a linear combination of the gradient vector and a supplemental vector. We have demonstrated a critical descent vector to derive a *globally optimal algorithm* (GOA), which can substantially accelerate the convergence speed in the numerical solution of nonlinear minimization problem. It was proven that the critical value α_c in the critical descent vector leads to the *largest convergence rate* among all the descent vectors specified by $\mathbf{u} = \mathbf{g} + \alpha\mathbf{Bg}$. Due to its criticality, if one attempts to find a better descent vector than $\mathbf{u} = \mathbf{g} + \alpha_c\mathbf{Bg}$, there would be no real descent vector of $\mathbf{u}$. Through several numerical tests we found that the both the OA and the GOA outperformed very well. Then we have proposed novel algorithms based on OA and GOA by replacing the Hessian matrix $\mathbf{A}$ and the descent matrix $\mathbf{B}$ with the updating techniques of BFGS for one or for both of these two matrices. Two numerical examples were given to test the performances of these algorithms, which are faster than the original OA and GOA algorithms, due to the enhancement by using the quasi-Newton conditions on the updated matrices.

Conflict of Interests

This paper is a purely academic research, and the author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The author highly appreciates the constructive comments from anonymous referees, which improved the quality of this paper. Highly appreciated are also the Project NSC-102-2221-E-002-125-MY3, the 2011 Outstanding Research Award from the National Science Council of Taiwan, and the 2011 Taiwan Research Front Award from Thomson Reuters. It is also acknowledged that the author has been promoted as being a Lifetime Distinguished Professor of National Taiwan University since 2013.

References

[1] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.

[2] M. Raydan, "On the Barzilai and Borwein choice of steplength for the gradient method," *IMA Journal of Numerical Analysis*, vol. 13, no. 3, pp. 321–326, 1993.

[3] M. Raydan, "The Barzilaiai and Borwein gradient method for the large scale unconstrained minimization problem," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 26–33, 1997.

[4] A. Friedlander, J. M. Martinez, B. Molina, and M. Raydan, "Gradient method with retards and generalizations," *SIAM Journal on Numerical Analysis*, vol. 36, no. 1, pp. 275–289, 1999.

[5] M. Raydan and B. F. Svaiter, "Relaxed steepest descent and Cauchy-Barzilai-Borwein method," *Computational Optimization and Applications*, vol. 21, no. 2, pp. 155–167, 2002.

[6] Y. H. Dai, J. Y. Yuan, and Y. Yuan, "Modified two-point stepsize gradient methods for unconstrained optimization," *Computational Optimization and Applications*, vol. 22, no. 1, pp. 103–109, 2002.

[7] Y. H. Dai and L.-H. Liao, "*R*-linear convergence of the Barzilai and Borwein gradient method," *IMA Journal of Numerical Analysis*, vol. 22, no. 1, pp. 1–10, 2002.

[8] Y. H. Dai and Y. Yuan, "Alternate minimization gradient method," *IMA Journal of Numerical Analysis*, vol. 23, no. 3, pp. 377–393, 2003.

[9] R. Fletcher, "On the Barzilai-Borwein gradient method," in *Optimization and Control with Applications*, L. Qi, K. Teo, and X. Yang, Eds., vol. 96 of *Applied Optimization*, pp. 235–256, Springer, New York, NY, USA, 2005.

[10] Y.-X. Yuan, "A new stepsize for the steepest descent method," *Journal of Computational Mathematics*, vol. 24, no. 2, pp. 149–156, 2006.

[11] E. G. Birgin and J. M. Martinez, "A spectral conjugate gradient method for unconstrained optimization," *Applied Mathematics and Optimization*, vol. 43, no. 2, pp. 117–128, 2001.

[12] N. Andrei, "Scaled conjugate gradient algorithms for unconstrained optimization," *Computational Optimization and Applications*, vol. 38, no. 3, pp. 401–416, 2007.

[13] N. Andrei, "A Dai-Yuan conjugate gradient algorithm with sufficient descent and conjugacy conditions for unconstrained optimization," *Applied Mathematics Letters*, vol. 21, no. 2, pp. 165–171, 2008.

[14] N. Andrei, "New accelerated conjugate gradient algorithms as a modification of Dai-Yuan's computational scheme for unconstrained optimization," *Journal of Computational and Applied Mathematics*, vol. 234, no. 12, pp. 3397–3410, 2010.

[15] L. Zhang, "A new Liu-Storey type nonlinear conjugate gradient method for unconstrained optimization problems," *Journal of Computational and Applied Mathematics*, vol. 225, no. 1, pp. 146–157, 2009.

[16] S. Babaie-Kafaki, R. Ghanbari, and N. Mahdavi-Amiri, "Two new conjugate gradient methods based on modified secant equations," *Journal of Computational and Applied Mathematics*, vol. 234, no. 5, pp. 1374–1386, 2010.

[17] Z.-J. Shi and J. Guo, "A new family of conjugate gradient methods," *Journal of Computational and Applied Mathematics*, vol. 224, no. 1, pp. 444–457, 2009.

[18] W. C. Davidon, "Variable metric method for minimization," AEC Research Development Report ANL-5990, 1959.

[19] R. Fletcher and M. Powell, "A rapidly convergent descent method for minimization," *The Computer Journal*, vol. 6, pp. 163–168, 1963.

[20] C.-S. Liu, "A Jordan algebra and dynamic system with associator as vector field," *International Journal of Non-Linear Mechanics*, vol. 35, no. 3, pp. 421–429, 2000.

[21] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms: 2. The new algorithm," *Journal of the Institute of Mathematics and its Applications*, vol. 6, no. 3, pp. 222–231, 1970.

[22] R. Fletcher, "A new approach to variable metric algorithms," *The Computer Journal*, vol. 13, no. 3, pp. 317–322, 1970.

[23] D. Goldfarb, "A family of variable-metric methods derived by variational means," *Mathematics of Computation*, vol. 24, pp. 23–26, 1970.

[24] D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics of Computation*, vol. 24, pp. 647–656, 1970.

[25] Y.-H. Dai, "Convergence properties of the BFGS algoritm," *SIAM Journal on Optimization*, vol. 13, no. 3, pp. 693–701, 2003.

[26] Y.-H. Dai, "A perfect example for the BFGS method," *Mathematical Programming A*, vol. 138, no. 1-2, pp. 501–530, 2013.

[27] P. E. Gill, W. Murray, and P. A. Pitfield, "The implementation of two revised quasi-Newton algorithms for unconstrained optimization," Tech. Rep. NAC-11, National Physical Laboratory, Teddington, UK, 1972.

[28] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *The Computer Journal*, vol. 3, no. 3, pp. 175–184, 1960.

[29] H.-C. Kuo, J.-R. Chang, and C.-H. Liu, "Particle swarm optimization for global optimization problems," *Journal of Marine Science and Technology*, vol. 14, no. 3, pp. 170–181, 2006.

[30] C.-S. Liu and S. N. Atluri, "A fictitious time integration method (FTIM) for solving mixed complementarity problems with applications to non-linear optimization," *Computer Modeling in Engineering & Sciences*, vol. 34, no. 2, pp. 155–178, 2008.

[31] P. Bouvry, F. Arbab, and F. Seredynski, "Distributed evolutionary optimization, in Manifold: Rosenbrock's function case study," *Information Sciences*, vol. 122, no. 2–4, pp. 141–159, 2000.

[32] S. Kok and C. Sandrock, "Locating and characterizing the stationary points of the extended Rosenbrock function," *Evolutionary Computation*, vol. 17, no. 3, pp. 437–453, 2009.

[33] M. J. D. Powell, "An iterative method for finding stationary values of a function of several variables," *The Computer Journal*, vol. 5, no. 2, pp. 147–151, 1962.