Modifying Of Barzilai and Borwein Method for Solving Large-Scale Unconstrained Optimization Problems.

Khalil K. Abbo*

ABSTRACT:

In this paper we present a technique for computing the minimum value of an objective function in the frame of gradient descent methods based on combination of Barzilai and Borwein approximation of Hessian matrix of objective function and Lipchitez constant in the gradient flow algorithm which is derived from a system of ordinary differential equations associated to unconstrained optimization problem. This algorithm suitable for large- scale unconstrained optimization problems, computational results for this algorithm is given and compared with BB method showing a considerable improvement.

* College of Computers Sciences and Mathematics /Department of Mathematics/ University of Mosul

Received: 29/ 8 /2006 ____________________Accepted: 30 / 10 / 2006
1- Introduction

A well-known algorithm for the unconstrained optimization of function \( f(x) \) in \( n \) variables

\[
f: \mathbb{R}^n \rightarrow \mathbb{R} ; \quad x \in \mathbb{R}^n \quad \text{.........(1)}
\]

having Lipchetz continuous first partial derivatives whose gradient

\( \nabla f(x) = g(x) \) is available, is the steepest descent method first proposed by Cauchy in 1874. The iterations are made according to the following equation

\[
x_{k+1} = x_k + \alpha_k d_k \quad \text{.........(2)}
\]
Where \( d_k = -g_k \) and \( \alpha_k \) is a step size. It’s well known that the negative gradient direction has the following optimal property see (Dai et. al. 1998)

\[
-g_k = \text{Min}_{d \in \mathbb{R}^n} \text{Lim}_{\alpha \to 0} \left[ f(x_k) - f \left( x_k + \frac{\alpha d}{\|d\|^2} \right) \right] \times \frac{1}{\alpha} \quad \ldots \ldots \text{(3)}
\]

Where \( \| \) is Euclidean norm. In the classical steepest descent method, the step size is obtained by carrying out an exact line search namely

\[
\alpha_k = \arg\min_{\alpha} f(x_k + \alpha d_k) \quad \ldots \ldots \text{(4)}
\]

However, despite the simplicity of the method and the optimal properties (3) and (4), the steepest descent method convergence slowly and is badly affected by ill-conditioning (Fletcher 1987), therefore not recommended for practical use. Different modifications are made to this method corresponding to different ways of choosing step size or modifying search directions.

The paper is organized as follows. In section 2 we review Barzilai and Borwein method. In section 3 steepest descent method with adaptive step size (SDAS). In section 4 problem reformulation and gradient flow algorithm are introduced. In section 5 our algorithm are derived and finally in section 6 numerical results are presented in order to compare the performance of the new algorithm.
2- Barzilai and Borwein (BB) method

Barzilai and Borwein in 1988 proposed two point step size gradient (BB) method (Barzilai and Borwein, 1988) by regarding $H_k = \lambda_k I$ as an approximation to the Hessian of $f$ at $x_k$ and imposing some–quasi–Newton property on $H_k$.

Denote $v_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$ by minimizing $\|v_{k-1} - H_k y_{k-1}\|$ they obtained

$$\lambda_k^{BB} = \frac{v_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}} \quad \cdots \cdots \cdots \cdots (5)$$

With this, the method of Barzilai and Borwein is given by the following iteration scheme

$$x_{k+1} = x_k - \frac{1}{\lambda_k^{BB}} g_k \quad \cdots \cdots \cdots \cdots (6)$$

The quantity given in (5) is frequently referred as a Rayleigh quotient. Indeed, if $f$ is twice continuously differentiable, we have (Andrei 2005)

$$y_k = \left[ \int_0^1 \nabla^2 f(x_k + tv_k) dt \right] v_k \quad \cdots \cdots \cdots \cdots (7)$$

Therefore

$$\lambda_k^{BB} = v_k^T \left[ \int_0^1 \nabla^2 f(x_k + tv_k) dt \right] v_k / v_k^T v_k \quad \cdots \cdots \cdots \cdots (8)$$
Which lies between the largest and the smallest eigenvalue of the Hessian Average.

\[
\int_{0}^{1} \nabla^2 f(x_k + tv_k) dt
\]

The scalar \( \lambda_k \) has been already used as scaling factor in the context of limited memory quasi Newton algorithms see for example (Liu and Nocedal 1989) or conjugate gradient algorithms (Shanno and Phua 1980) and (Andrei 2005).

The BB method received a great deal of attention for its simplicity and numerical efficiency for well-conditioned problems, and analyzed by Raydan (Raydan, 1993), have a number of interesting feature that make them attractive for the numerical solution of (1). The most important features of this method is that only gradient directions are used, that the memory requirements are minimal and that they do not involve a decrease in the objective function, which allows fast local convergence. They have been applied successfully to find local minimizers of large scale real problems (Luengo and Raydan, 2003).

Raydan proved that for strictly convex function with any variable the (BB) method is globally convergence, despite of these advances of (BB) method on quadratic functions, still many open questions about this method on non-quadratic functions although
Fletcher (Fletcher, 2001) show that the method may very slow on some problems.

3- **Steepest Descent with Adaptive Stepsize (SDAS)**

In the scheme (2) the search direction $d_k$ satisfied descent condition i.e

$$d_k^T g_k < 0 \quad \text{........}(9)$$

Which guarantees that $d_k$ is a descent direction of $f(x)$ at $x_k$. In order to guarantee the global convergence it’s usually required to satisfy the condition

$$g_k^T d_k \leq -c\|g_k\| \quad \text{........}(10)$$

Where $c > 0$ is a constant.

Many procedures for step size computation $\alpha_k$ have been proposed for example minimizer rule, Armijo rule, limited minimization rule, and strong wolf rule which states that at the k-th iteration, $\alpha_k$ satisfies simultaneously

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \sigma_1 \alpha_k g_k^T d_k \quad \text{........}(11)$$

And

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma_2 g_k^T d_k \quad \text{...............}(12)$$

Where $\sigma \in (0, \frac{1}{2})$ and $\rho \in (\sigma, 1)$, and there are many other line search procedures for example see (Vrahatis et al 2000).

In the recent paper of zhen, (Zhen and Jie 2005)
An estimation of stepsize is introduced by means of lipschitz constant. As follows

For $k=0$ select $L_0 > 0$ and for $k \geq 1$

$$L_k = \text{Max} \left[ L_{k-1}, \frac{\|g_{k-1}\|}{\|g_k\|} \right] \quad \ldots \ldots (13)$$

And they proved that their algorithm is globally convergent and rate of convergence is linear (Zhen and Jie 2005), her we review briefly their algorithm.

**Algorithm (A):**

**Step (1):** $k = 0$ choose $x_0 \in \mathbb{R}^n, \delta \in (0,2) and L_0 > 0, \varepsilon = 10^{-6}$

**Step (2):** if $\|g_k\| < \varepsilon$ then stop. else goto step (3)

**Step (3):** Estimate $L_k$ from (13)

**Step (4):** $x_{k+1} = x_k + \alpha_k d_k$ where $\alpha_k = \frac{\delta}{L_k}$

**Step (5):** $k = k + 1$ goto step (2)

In the above algorithm line search procedure is avoided at each iteration, which may reduce the cost of computation. However, we must estimate $L_k$ at each iteration. Main draw back of algorithm (A) is, if $L_k$ is very large then $\alpha_k$ will be very small and will slow the convergence rate of the method, on the other hand if $L_k$ is very small
then $\alpha_k$ will be large and hence the method may fail to guarantee the global convergence.

4-Problem Reformulation and Gradient Flow Algorithm

For the unconstrained optimization problem given in equation (1), as we know a necessary condition for point $x^*$ be an optimal solution is

$$g(x^*) = 0 \quad \ldots (14)$$

This is a system in $n$ Non-linear equations which must be solved to get the optimal solution $x^*$. In order to fulfill this optimality condition the following continuous gradient flow reformulation of the problems is suggested (khalaf and Al-Wagih, 2001). Solve the following system of ordinary differential equation

$$\frac{dx(t)}{dt} = -g(x(t)) \ldots (15)$$

With initial condition

$$x(0) = x_0 \quad \ldots (16)$$

The solution of the system (15) with initial condition (16) is convergence to optimal solution which is minimum of the function given in (1) according to the following theorems, see (Andrei, 2003)
Theorem (1):
Consider that \( x^* \) is a point satisfying (14) suppose that 
\[ G = \nabla^2 f(x^*) \] is positive definite, if \( x_0 \) is close enough to \( x^* \), then \( x(t) \) solution of (15) tends to \( x^* \) as \( t \to \infty \).

Theorem (2):
Let \( x(t) \) be the solution of (15), for fixed \( t_0 \geq 0 \) if \( g(x(t)) \neq 0 \) for all \( t > t_0 \), then \( f(x(t)) \) is strictly decreasing with respect to \( t \) for all \( t > t_0 \).

As we have seen solving the unconstrained optimization problem (1) has been reduced to that of integration of the ordinary differential equation (15) with initial condition (16). Andrei proposed an algorithm for solving the system (15) as follows

Let \( 0 = t_0 < t_1 < \ldots < t_k < \ldots \) be sequence of time points and consider 
\[ h_k = t_{k+1} - t_k \] the sequence of time distance between two successive time points, consider the following (Andrei, 2003), discretization of (15)
\[
\frac{x_{k+1} - x_k}{h_k} = -(1 - \theta)g_k + \theta g_{k+1} \quad \cdots \cdots (17)
\]
where \( \theta \in [0,1] \) is a parameter, then
\[
x_{k+1} = x_k - h_k [(1 - \theta)g_k + \theta g_{k+1}] \quad \cdots \cdots (18)
\]
It’s clear that when $\theta = 0$ the above discretization is the explicit forward Euler’s scheme. On the other hand, when $\theta = 1$ we have the implicit backward Euler’s scheme. But

$$g_{k+1} = g_k + G_k y_k + o(h^2) \ldots (19)$$

Omitting the last term equation (19) and substituting in equation (18) we obtain

$$x_{k+1} = x_k - h_k [I + h_\theta G_k]^{-1} g_k \ldots (20)$$

In fact the algorithm given in (20) introduced by Zghier but he derived it by using generalized trapezoidal rule with $\theta = 0.5$ see (Zghier 1981) or (Brown and Biggs 1989).

Many authors (for example Botsaris(1978), Brown(1986), and others solved systems (15) and (20) with initial condition (16) by some well known integration methods. Andrei (2003) showed if $x_0$ as initial guss close enough to $x^*$ and if $G_k$ is positive then the algorithm given in equation (20) is convergence for $\theta \in [0, 1]$ and rate of convergence is linear when $\theta = 0$, super linear for $0 < \theta < 1$ and order of convergence is two if $\theta = 1$

5- Proposed Algorithm  (Gradient Flow Steepest Descent GFSD say)

The method based on the equation (20) has quite good performance if $G_k$ is positive definite and have desirable features, but not recommended for practical use, the major drawback of the algorithm is computing $(I + h_\theta G_k)^{-1}$ at each iteration and
also there is no specified value of h. However one can deduce a simple implementation of the algorithm given (20) with preserving useful theoretical features as follows:

Let $G_k = \lambda_k^{bb} I$, where $I_{nn}$ Identity matrix as an approximation of $G_k$ and put $h_k = L_k$ then substitute in (20)

\[
x_{k+1} - x_k = -L_k \left[ I + L_k \theta \lambda_k^{bb} I \right]^{-1} g_k
\]

\[
\frac{1}{L_k} (x_{k+1} - x_k) = - \left[ I + \theta \frac{y_k^T y_k}{v_k^T v_k} \right]^{-1} g_k
\]

\[
\frac{1}{L_k} v_k = - \left[ 1 + \theta \frac{y_k^T y_k}{v_k^T v_k} \right]^{-1} g_k
\]

\[
d_k = - \left[ \frac{v_k^T v_k + \theta y_k^T y_k}{v_k^T v_k} \right] g_k
\]

\[
d_k = - \left[ \frac{v_k^T v_k}{v_k^T v_k + \theta y_k^T y_k} \right] g_k \quad \ldots \ldots (21)
\]

Therefore one can compute the new point from the following algorithm

\[
x_{k+1} = x_k - \alpha_k g_k \quad \text{where} \quad \alpha_k = \frac{v_k^T v_k}{v_k^T v_k + \theta y_k^T y_k}
\]

(22)

which is generalization of the BB method. It's clear that if $\theta = 0$ the algorithm restarts with steepest descent direction. The convergence properties of the method given in (22) can be studied providing that $d_k = -g_k$ is descent direction and using the following proposition.
Proposition (1)
Assume that the step size $\alpha_k$ satisfies wolf conditions (11) and (12) and that $d_k$ is descent direction then $y_k^T v_k > 0$.

for proof see (Barzilai and Borwein 1988).

Theorem (3):
Suppose that $f$ is bounded below in $R^n$ and that $f$ is continuously differentiable in neighborhood of the level set $L = \{ x : f(x) \leq f(x_0) \}$ -
Assume also that the gradient $g_k$ is lipchitz continuous i.e there exists a constant $c > 0$ s.t
$$\| g(x) - g(y) \| \leq c \| x - y \| \forall x, y \in$$
Consider any iteration of the form
$$x_{k+1} = x_k + \alpha_k d_k \text{ where } \alpha_k = \frac{v_k^T v_k}{v_k^T v_k + \theta_k^T v_k} \text{ and } d_k = -g_k$$
and $\alpha_k$ satisfies wolf conditions then $\lim_{k \to \infty} \| g_k \| = 0$.

proof:
from equation (12) we have
$$(g_{k+1} - g_k)^T d_k \geq (\sigma_2 - 1) g_k^T d_k \quad \text{........(23)}$$
on the other hand ,the lipchitz condition
$$(g_{k+1} - g_k)^T d_k \leq \alpha_k c \| d_k \|^2 \quad \text{........(24)}$$
from (23) and (24) we get
$$\alpha_k \geq \left( \frac{\sigma_2 - 1}{c} \right) \frac{g_k^T d_k}{\| d_k \|^2} \quad \text{.................(25)}$$
using equations (11) and (25) we have

\[
 f_{k+1} \leq f_k + \sigma_1 \left( \frac{\sigma_2 - 1}{c} \right) \left( \frac{g_k^T d_k}{\|d_k\|^2} \right)^2 
\]  

(26)

now using the relation

\[
 \|g_k\| d_k \cos \gamma_k = -g_k^T d_k \quad \text{where } \gamma_k \text{ is the angle between } g_k \text{ and } d_k
\]

then equation (26) can be written as

\[
 f_{k+1} \leq f_k + t \|g_k\| \cos^2 \gamma_k 
\]  

(27)

where \( t = \frac{\sigma_1 (\sigma_2 - 1)}{c} \)

summing the expression in equation (27) and recalling \( f \) bounded below, we obtain

\[
 \sum \cos^2 \gamma_k \|g_k\|^2 < \infty 
\]  

(28)

assuming that \( \cos^2 \gamma_k > \delta > 0 \) for all \( k \), then we conclude that

(Nocedal 1992)

\[
 \lim_{k \to \infty} \|g_k\| = 0
\]

Out line of the algorithm (GFSD)

**Step (1):** \( k = 0 \) choose \( x_0 \in \mathbb{R}^n, \varepsilon > 0, \theta \in [0,1]; d_0 = -g_0 \)

**Step (2):** if \( \|g_k\| < \varepsilon \) stop. else goto step (3)

**Step 3):**

Compute \( \alpha_k \) from equation (22) and test for Wolfe conditions if satisfied accept \( \alpha_k \) as stepszie else set \( \alpha_k = 1 \) and use backtracking to satisfy Wolf conditions
Step (4): \( x_{k+1} = x_k - \alpha_k g_k \)

Step (5): compute \( g_k, v_k, y_k \)

Step (6): \( k = k + 1 \) goto step (2)

6-Numerical Results

We present the numerical results for the Barzilai-Borwein method and proposed (GFSD) method for some well known test functions (Bongartz and el at,1995), these algorithms are coded in double precision FORTRAN 90 language. The criteria for stopping the iterations are

\[
\|g_{k+1}\| < 10^{-6} \quad \text{or} \quad \alpha_{k+1} |g_{k+1}^T g_{k+1}| < 10^{-20} |f_{k+1}|
\]

For both methods initial step size are computed by using backtracking procedure with \( \sigma = 0.0001 \) and \( \rho = 0.8 \). Also Wolfe conditions are used for accepting step size, the complete set of results are given (a) with \( 1000 \leq n \leq 5000 \) and table (b) with \( 6000 \leq n \leq 10000 \). In Tables (a) and (b) we present the comparison results of BB and GFSD methods for different dimensions with \( 1000 \leq n \leq 5000 \) consisting number of iteration (NOI), function gradient F&G ev evaluations (they are equal in these algorithms) and the execution time in nanosecond are compared it shown that the proposed algorithm is better than BB method in most cases. In Tables (a1) and (b1) we see that there is an improvement about % 5 in NOI , % 15.14 in F&G ev and %21 in execution time for
dimensions $1000 \leq n \leq 5000$. And $\%4$ in NOI, $\%11$ in F&G ev and $\%13$ in execution time for dimensions with $6000 \leq n \leq 10000$. These algorithms are not compared with algorithm (A) since it shown by (Andrei, 2005) BB method is better than algorithm (A).

Table (a) Comparison of BB method and GFSD with $1000 \leq n \leq 5000$

<table>
<thead>
<tr>
<th>Test Fun.</th>
<th>N</th>
<th>Barzilai-Bowen Algorithm</th>
<th>GFSD Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>NOI</td>
<td>F&amp;G</td>
</tr>
<tr>
<td>Freudenstein &amp; Roth</td>
<td>1000</td>
<td>218</td>
<td>1615</td>
</tr>
<tr>
<td>Extended Trigonometric</td>
<td>1000</td>
<td>44</td>
<td>148</td>
</tr>
<tr>
<td>Extended Rosenbrock</td>
<td>1000</td>
<td>248</td>
<td>1981</td>
</tr>
<tr>
<td>Beal Fun.</td>
<td>2000</td>
<td>153</td>
<td>614</td>
</tr>
<tr>
<td>Penalty Fun.</td>
<td>1000</td>
<td>54</td>
<td>232</td>
</tr>
<tr>
<td>Raydan 1</td>
<td>1000</td>
<td>1134</td>
<td>5631</td>
</tr>
<tr>
<td>Raydan 2</td>
<td>2000</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td>Ge. Tridigonal 1</td>
<td>1000</td>
<td>29</td>
<td>90</td>
</tr>
<tr>
<td>Extended Wood F</td>
<td>1000</td>
<td>1043</td>
<td>7083</td>
</tr>
<tr>
<td>DIXMAAN (cute)</td>
<td>2000</td>
<td>425</td>
<td>2335</td>
</tr>
<tr>
<td>Freudenstein &amp; Roth</td>
<td>3000</td>
<td>190</td>
<td>1410</td>
</tr>
<tr>
<td>Extended Trigonometric</td>
<td>4000</td>
<td>30</td>
<td>124</td>
</tr>
<tr>
<td>Extended Rosenbrock</td>
<td>5000</td>
<td>703</td>
<td>5544</td>
</tr>
<tr>
<td>Beal Fun.</td>
<td>3000</td>
<td>152</td>
<td>614</td>
</tr>
<tr>
<td>Penalty Fun.</td>
<td>3000</td>
<td>46</td>
<td>219</td>
</tr>
<tr>
<td>Raydan 1</td>
<td>2000</td>
<td>1832</td>
<td>9419</td>
</tr>
<tr>
<td>Raydan 2</td>
<td>5000</td>
<td>21</td>
<td>94</td>
</tr>
<tr>
<td>Ge. Tridigonal 1</td>
<td>3000</td>
<td>31</td>
<td>94</td>
</tr>
<tr>
<td>Extended Wood F</td>
<td>3000</td>
<td>1126</td>
<td>7545</td>
</tr>
<tr>
<td>Dixmaan (cute)</td>
<td>4000</td>
<td>641</td>
<td>3499</td>
</tr>
<tr>
<td>Total</td>
<td>8109</td>
<td>48285</td>
<td>68.3</td>
</tr>
</tbody>
</table>
Table a1) Percentage of improving the GFSD with $1000 \leq n \leq 5000$

<table>
<thead>
<tr>
<th>Tools</th>
<th>BB method</th>
<th>GFSD method</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOI</td>
<td>%100</td>
<td>%95.19</td>
</tr>
<tr>
<td>F&amp;G EV</td>
<td>%100</td>
<td>%84.36</td>
</tr>
<tr>
<td>Time</td>
<td>%100</td>
<td>%78.62</td>
</tr>
</tbody>
</table>

Table b) Comparison of BB method and GFSD with $6000 \leq n \leq 10000$

<table>
<thead>
<tr>
<th>Test Fun.</th>
<th>N</th>
<th>Barzilai-Bowen Algorithm</th>
<th>GFSD Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>NOI</td>
<td>FGEV</td>
</tr>
<tr>
<td>Freudenstein &amp; Roth</td>
<td>6000</td>
<td>270</td>
<td>1995</td>
</tr>
<tr>
<td>Extended Trigonometric</td>
<td>7000</td>
<td>32</td>
<td>133</td>
</tr>
<tr>
<td>Extended Rosenbrock</td>
<td>9000</td>
<td>408</td>
<td>3183</td>
</tr>
<tr>
<td>Beal Fun.</td>
<td>7000</td>
<td>46</td>
<td>608</td>
</tr>
<tr>
<td>Penalty Fun.</td>
<td>8000</td>
<td>55</td>
<td>241</td>
</tr>
<tr>
<td>Raydan 1</td>
<td>6000</td>
<td>4345</td>
<td>23262</td>
</tr>
<tr>
<td>Raydan 2</td>
<td>7000</td>
<td>20</td>
<td>38</td>
</tr>
<tr>
<td>Ge. Tridigonal 1</td>
<td>8000</td>
<td>37</td>
<td>102</td>
</tr>
<tr>
<td>Extended Wood F</td>
<td>9000</td>
<td>1094</td>
<td>7388</td>
</tr>
<tr>
<td>DIXMAAN (cute)</td>
<td>6000</td>
<td>697</td>
<td>3594</td>
</tr>
<tr>
<td>Freudenstein &amp; Roth</td>
<td>8000</td>
<td>295</td>
<td>2173</td>
</tr>
<tr>
<td>Extended Trigonometric</td>
<td>10000</td>
<td>32</td>
<td>137</td>
</tr>
<tr>
<td>Extended Rosenbrock</td>
<td>10000</td>
<td>527</td>
<td>4193</td>
</tr>
<tr>
<td>Beal Fun.</td>
<td>10000</td>
<td>152</td>
<td>612</td>
</tr>
<tr>
<td>Penalty Fun.</td>
<td>9000</td>
<td>58</td>
<td>255</td>
</tr>
<tr>
<td>Raydan 1</td>
<td>10000</td>
<td>6301</td>
<td>33159</td>
</tr>
<tr>
<td>Raydan 2</td>
<td>10000</td>
<td>22</td>
<td>44</td>
</tr>
<tr>
<td>Ge. Tridigonal 1</td>
<td>10000</td>
<td>39</td>
<td>115</td>
</tr>
<tr>
<td>Extended Wood F</td>
<td>10000</td>
<td>1052</td>
<td>6978</td>
</tr>
<tr>
<td>Dixmaan (cute)</td>
<td>7000</td>
<td>624</td>
<td>3609</td>
</tr>
<tr>
<td>Total</td>
<td>16206</td>
<td>91821</td>
<td>297.316</td>
</tr>
</tbody>
</table>
Table (b1) Percentage of improving the GFSD with $6000 \leq n \leq 10000$

<table>
<thead>
<tr>
<th>Tools</th>
<th>BB method</th>
<th>GFSD method</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOI</td>
<td>%100</td>
<td>%95.8</td>
</tr>
<tr>
<td>F&amp;G EV</td>
<td>%100</td>
<td>%88.9</td>
</tr>
<tr>
<td>Time</td>
<td>%100</td>
<td>%86.6</td>
</tr>
</tbody>
</table>

**Conclusion**

These types of algorithms are suitable for large-scale unconstrained optimization problems. Our numerical results indicates that there are an improvements of proposed algorithm especially on F&G EV I think which means that step size given by BB method is ether typically large or small and hence it require more functions and gradient evolutions to accept the step size to reduce in function value.

**References:**

1- Andrei. N “Gradient flow algorithm for unconstrained optimization”
   Research Institute for informatics center for advanced Modeling Optimization 8-10 Averescu Avenue, Bucharest, 2003
   E-mail: nandrei@u3.ici.ro.

2- Andrei. N “Scaled conjugate Gradient algorithms for unconstrained Optimization”

3- Barzilai. J and Borwein. M “Two Points stepsize Gradient Methods”
4- Brown, A and Batholomew-Biggs, C
  “Some Effective Methods For unconstrained optimization Based on the Solution of systems of differential equations”

5- Brown, A
  "Optimization Methods Involving The solution Of Ordinary Differential Equations"

7- Botsaris, C
  "A class Of Methods For Unconstrained Minimization Based On Stable Numerical Integration Techniques"
  Journal of Mathematical Analysis and Application 63. 1978

8- Bongartz, I., Conn, A, Gould, N and Toint, P
  "CUTE: Constrained and unconstrained testing environments"

9- Dai, Y., Yuan, J and Yuan, X
  “Modified two point stepsize Gradient Methods for unconstrained Optimization”
  E-mail: dyh@iase.cc.ac.cn

10- Fletcher, R
  “Practical Methods of Optimization”

11- Fletcher, R
  “On the Barzilai-Borwein Method”
“Parallel shooting Method for unconstrained Numerical Optimization”

“On the limited Memory BFGS Method for Large Scale Optimization”

14- Luengo. F and Raydan. M
"Gradient Method With Dynamical Retards For Large-Scale Optimization Problems" Electronic Transactions on Numerical Analysis vol(16), 2003 Kent state University

15- Raydan. M
“On the Barzilai-Borwein Choice of the step length for the gradient Method”

15- Nocedal. J "Theory Of Algorithms For Unconstrained Optimization"
Acta Numerica pp 199-242, 1992

16- Shanno. F and Phua.H
“Remark on algorithm 500: minimization of unconstrained multivariate Functions”

“A class of gradient unconstrained Minimization Algorithms with Adaptive stepszie”
18- Zhen. J and Jie. S

“Stepsie Estimation for unconstrained Optimization Methods”

19- Zghier. A

"The use of Differential Equations in Optimization"