American Journal of Operational Research 2013, 3(2): 57-64
DOI: 10.5923/j.ajor.20130302.05

# The Eigenstep Method: An Iterative Method for Unconstrained Quadratic Optimization

**John P. Battaglia**

Department of Mathematics and Statistics,Windsor Ontario, Canada

**Abstract** This paper presents a method for the unconstrained minimization of convex quadratic programming problems. The method is a line search method, an iterative nonmonotone gradient method that is a modification of the classical steepest descent method. The two methods are the same in the choice of the negative gradient as the search direction, but differ in the choice of step size. The steepest descent method uses the optimal step size introduced by Cauchy in the nineteenth century and the proposed method uses the reciprocal of the eigenvalues of the Hessian matrix as step sizes. Thus, the proposed method is referred to as the eigenstep method. We introduce and study three more recent developments, also modifications of the steepest descent method that alter the optimal Cauchy choice of steplength with nonmonotone steplength choices. Numerical examples with encouraging results are given to illustrate our new algorithm and a comparison is made to two standard optimization methods as well as to the three more recent developments in line search methods presented in this paper.

**Keywords** Nonmonotone, Gradient, Line Search

## 1. Introduction

This paper presents a method for the unconstrained minimization of the convex quadratic programming problem (QP) given by

$$\min f(x) = \frac{1}{2} x^T Q x - b^T x,$$

where $x \in \mathbb{R}^n$, $b \in \mathbb{R}^n$ and Q is a $\mathbb{R}^{n \times n}$ symmetric, positive definite matrix. This paper will be organized as follows. In ***section 2*** we define, give the development and a detailed description of our new eigenstep method. We also prove that the eigenstep method will solve QP in at most *n* iterations, *n* being of course the number of variables of QP.

In ***section 3*** we discuss two standard, well known optimization methods, the steepest descent method introduced by Cauchy in the nineteenth century as well as the conjugate gradient method.

***Section 4*** will be devoted to three more recent developments in iterative line search methods. The first being a method developed by Barzilai and Borwein[2] in 1988. Later in 2002, Raydan and Svaiter[1] extended this line of research by studying the positive effects of using over and under relaxed steplengths for the Cauchy method and introduced the relaxed steepest descent or Cauchy method. Motivated by the results of their relaxed Cauchy method, Raydan and Svaiter present a new method that is a modification of the Barzilai-Borwein method, namely the Cauchy - Barzilai-Borwein method. We also have provided references that give commentary on the above mentioned developments.

For example, in 2002 Dai and Liao[3] prove R-linear rate of convergence of the Barzilai -Borwein method and Raydan [5] establishes global convergence of the Barzilai-Borwein method in the convex quadratic case.

Finally, in ***section 5*** we give two examples to illustrate our new eigenstep method and then we provide numerical experiments comparing all methods.

## 2. The Eigenstep Method

We present a method for the unconstrained minimization of the convex programming problem (QP) given by

$$\min f(x) = \frac{1}{2} x^T Q x - b^T x$$

where $x \in \mathbb{R}^n$, $b \in \mathbb{R}^n$ and Q is a $n \times n$ real, symmetric positive definite (SPD) matrix. Since Q is SPD it has *n* real positive eigenvalues $\lambda_k$, k = 0,1, …,n-1. The function f is referred to as the objective function. The vector of first derivatives of f(x) is $\nabla f(x) = Qx - b.$ The matrix of second derivatives of f(x) is the Hessian matrix Q. This problem is equivalent to solving the linear system $Qx = b$ and since Q is SPD and f is strictly convex our QP has a unique global minimizer $x^* = Q^{-1}b.$

* Corresponding author:
johnbatt@uwindsor.ca (John P. Battaglia)
Published online at http://journal.sapub.org/ajor

Many algorithms, for the solution of the QP use the *standard iteration*

$$x_{k+1} = x_k + \alpha_k s_k,$$

where $\alpha_k$ is a positive real scalar referred to as the step size and $s_k \in \mathbb{R}^n$ is referred to as the search direction. For convenience we use the notation $g_k$ to refer to the gradient of f(x) at $x_k$, i.e., $g_k = \nabla f(x_k)$. We say that $\alpha_k$ is the optimal step size, which we will denote by $\alpha_k^*$ if

$$f(x_k + \alpha_k s_k) = \min_{\alpha}\{f(x_k + \alpha s_k)\}.$$

For our QP the optimal step size is given by

$$\alpha_k^* = -\frac{g_k^T s_k}{s_k^T Q s_k}.$$

The following two lemmas motivate the eigenstep method.

**Lemma 2.1** If $g_k$ is an eigenvector of Q with eigenvalue $\lambda$, and if $s_k = -g_k$ then

$$\alpha_k^* = \frac{1}{\lambda}.$$

**Proof.** We know $Qg_k = \lambda g_k$. Substituting this in $\alpha_k^* = \frac{g_k^T g_k}{g_k^T Q g_k}$ we obtain the desired result.

**Lemma 2.2** Suppose that $(x^* - x_k)$ is an eigenvector of Q with eigenvalue $\lambda$. If $s_k = -g_k$ and $\alpha_k = \frac{1}{\lambda_k}$, then $x_{k+1} = x^*$, the global minimizer of f(x).

**Proof.** Since $\nabla f(x^*) = 0$ and $\nabla f(x) = Qx - b$, we have

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k s_k \\ &= x_k - \frac{1}{\lambda} g_k \\ &= x_k + \frac{1}{\lambda}(\nabla f(x^*) - g_k) \\ &= x_k + \frac{1}{\lambda}(Qx^* - b - (Qx_k - b)) \\ &= x_k + \frac{1}{\lambda} Q(x^* - x_k) \\ &= x_k + \frac{1}{\lambda}\lambda(x^* - x_k) \\ &= x^*. \end{aligned}$$

*Lemmas 2.1 and 2.2 suggest that for our QP if we force the step sizes at iteration k of the steepest descent method to be* $\frac{1}{\lambda_k}$*, optimality could be attained in fewer steps.*

**The Eigenstep Algorithm** (the eigenstep method for *QP*)

Calculate the distinct eigenvalues of *Q* denoted by $\lambda_k$, k = 0,1,…, *p*-1, $p \le n$,

*n* being the number of variables and *p* the number of distinct eigenvalues of *Q*. Arrange the eigenvalues in decreasing order, $\lambda_0 < \lambda_1 < \ldots < \lambda_{p-1}$.

Given $x_0 \in \mathbb{R}^n$

1. For $k = 0,1,\ldots,$ do
2. Set $g_k = \nabla f(x_k)$
3. Set $x_{k+1} = x_k - \frac{1}{\lambda_k} g_k$
4. If the stop condition is satisfied end do,

otherwise set $k = k+1$ and go to 1.

Operated iteratively, the eigenstep algorithm initiated at point $x_0 \in \mathbb{R}^n$ will generate the sequence $\{x_k\}$ defined by the main iterative step

$$x_{k+1} = x_k - \frac{1}{\lambda_k} g_k,\ k = 0,1,\ldots,p\text{-}1,\ p \le n,$$

where each $\lambda_k$ represents an eigenvalue of Q.

**Remark:** In order to apply the eigenstep method one must choose a step size order. In the proof of termination to follow it is evident that the order is of no consequence, thus to apply our new method we utilize an increasing step size.

**Transformation to Standard Form.**

Next we describe two linear substitutions that transform our objective function to standard form.

Since Q is SPD there exists an orthogonal matrix P such that

$$D = P^T Q P = \mathrm{diag}\{\lambda_0,\ldots,\lambda_{n-1}\}.$$

If we let x = Py we can rewrite

$$f(x) = f(Py) = \frac{1}{2} y^T P^T Q P y - b^T P y.$$

Letting $g(y) = f(Py)$ and $d = P^T b$, we have transformed the objective function to

$$g(y) = \frac{1}{2} y^T D y - d^T y.$$

We now set $y = z + D^{-1}d$ since $\nabla g(D^{-1}d) = 0$ to obtain

$$\begin{aligned} g(z + D^{-1}d) &= \frac{1}{2}(z + D^{-1}d)^T D(z + D^{-1}d) - d^T(z + D^{-1}d) \\ &= \frac{1}{2} z^T D z - \frac{1}{2} d^T D^{-1} d. \end{aligned}$$

Letting $h(z) = g(z + D^{-1}d) = g(z + D^{-1}P^T b)$ and ignoring the constant $\frac{1}{2} d^T D^{-1} d$

(it does not affect the value of the minimizer $z^*$) we have transformed the objective function to standard form

$$h(z) = \frac{1}{2} z^T D z .$$

The next lemma shows a one to one correspondence between the minimizers of our QP and our QP in standard form.

**Lemma 2.3** $z^*$ is a unique global minimizer for $h(z)$, if and only if

$x^* = Py^* = P(z^* + D^{-1}d) = P(z^* + D^{-1}P^T b)$ is the unique global minimizer for f(x).

**Proof.** We know $D = P^T Q P = \mathrm{diag}\{\lambda_0, ..., \lambda_{n-1}\}$ is non-singular therefore $z^* = 0.$

Since, $x^* = Py^* = PD^{-1}d$, we have

$$\begin{aligned} \nabla f(x^*) &= Qx^* - b \\ &= Q(Py^*) - b \\ &= QPD^{-1}d - b \\ &= QPD^{-1}P^T b - b \\ &= QQ^{-1}b - b \\ &= 0. \end{aligned}$$

Conversely, assume $x^*$ is the unique global minimizer for f(x), thus $x^* = Q^{-1}b$

and $z^* = P^{-1}(x^* - Q^{-1}b) = 0.$

The next theorem sets the stage for our general proof which follows.

**Theorem 2.1** The eigenstep method will solve a QP in standard form in at most *n* iterations.

**Proof.** The QP to be solved is min $h(z) = \frac{1}{2} z^T D z$. Let $z_0$ be arbitrary but fixed. The eigenstep method gives the iterates

$$z_1 = z_0 - \frac{1}{\lambda_0} D z_0 = (I - \frac{1}{\lambda_0} D) z_0,$$

$$z_2 = z_1 - \frac{1}{\lambda_1} D z_1 = (I - \frac{1}{\lambda_1} D)(I - \frac{1}{\lambda_0} D) z_0,$$

$$z_3 = z_2 - \frac{1}{\lambda_2} D z_2 = (I - \frac{1}{\lambda_2} D) z_2 = (I - \frac{1}{\lambda_2} D)(I - \frac{1}{\lambda_1} D)(I - \frac{1}{\lambda_0} D) z_0$$

Continuing we see that

$$z_n = (I - \frac{1}{\lambda_0} D)(I - \frac{1}{\lambda_1} D) \ldots\ldots (I - \frac{1}{\lambda_{n-1}} D) z_0 = \prod_{i=0}^{n-1} (I - \frac{1}{\lambda_i} D) z_0 .$$

The i-th row of $(I - \frac{1}{\lambda_i} D)$ is a zero row since the diagonal element is $1 - (\frac{1}{\lambda_i})\lambda_i$.

Thus, the product is the zero matrix and it follows that $z_n = 0,$ the global minimizer.

**Proof of Termination**

**Theorem 2.2** The eigenstep method will solve our QP in at most *n* iterations.

**Proof.** We will show a one to one correspondence between the iterates of the eigenstep method applied to our QP and the iterates of the eigenstep method applied to our QP in standard form. The result will then follow from Theorem 1.

We proceed by induction. Let $x_0$ be arbitrary but fixed, and let $z_0 = P^T x_0 - D^{-1} P^T b.$

Assume that for

$$i = 1, ..., k, \; x_i = P(z_i + D^{-1}P^T b) = Pz_i + Q^{-1}b.$$

We have

$$\begin{aligned} x_{k+1} &= x_k - \frac{1}{\lambda_k}(Qx_k - b) \\ &= Pz_k + Q^{-1}b - \frac{1}{\lambda_k}(Q(Pz_k + Q^{-1}b) - b) \\ &= Pz_k + Q^{-1}b - \frac{1}{\lambda_k}(QPz_k) \\ &= Pz_k - \frac{1}{\lambda_k} PDP^T Pz_k + Q^{-1}b \\ &= Pz_k - \frac{1}{\lambda_k} PDz_k + Q^{-1}b \\ &= P(z_k - \frac{1}{\lambda_k} Dz_k) + Q^{-1}b \\ &= Pz_{k+1} + Q^{-1}b. \end{aligned}$$

A practical motivation for the introduction of Eigenstep method is that it not only solves the QP in at most *n* iterations, but also saves the computation in each iteration. Since the eigenvalues of Q are fixed for the problem and do not vary in each iteration, we can pre-calculate them before performing the iterative programming. The total computation complexity in iterations is the cheapest compared with all other methods presented in this paper.

# 3. Standard Methods

**The Steepest Descent Method.**

The steepest descent method uses $s_k = -g_k$ as the search direction since $-g_k$ is a direction of steepest descent and uses the *optimal* step size. For instances of the QP in which the eigenvalues of the Hessian have different orders of magnitude, it is known that the steepest descent method can exhibit poor performance. Our research has shown that this poor performance is due to the choice of steplength not the search direction. Our attempts to alter the steplength of the steepest descent method in order to improve its performance gave birth to the new eigenstep method .

**The Conjugate Gradient Method.**

Early attempts to modify the steepest descent method led to the development of this method which can be shown to solve our QP in *n* steps using again our *standard iteration*. This method gets its name from the fact that it generates a set of vectors $s_0, s_1, ..., s_{n-1}$ that are conjugate with respect to Q that is, $s_i^T Q s_j = 0$ for $i \neq j$. In the case where Q = I the conjugate vectors are just orthogonal vectors. The method is initialized by being given, $x_0, s_0 = -g_0$, while $g_k \neq 0$

$$x_{k+1} = x_k - \alpha_k^* s_k \text{ and } s_{k+1} = -g_{k+1} + \frac{\|g_{k+1}\|^2}{\|g_k\|^2} s_k.$$

# 4. Recent Developments

In 1988 Barzilai and Borwein[2] presented alternatives to the optimal step size. They introduced a nonmonotone steplength which avoids the drawbacks of the steepest descent method. The Barzilai-Borwein method for the unconstrained minimization problem can be written as

$$x_{k+1} = x_k - \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T (g_k - g_{k-1})} g_k$$

where $s_{k-1} = x_k - x_{k-1}$ . For QP the Barzilai-Borwein method reduces to $x_{k+1} = x_k - \beta_{k-1} g_k$, where $\beta_{k-1}$ is the optimal choice of the steepest descent method at the previous iteration.

**Relaxed Steepest Descent Method (RSD)**

Inspired by the work of Barzilai and Borwein, Raydan and Svaiter[1] introduced two new methods which will be discussed in this paper.

The first method developed by Raydan and Svaiter in[1] is a modification of the steepest descent method applied to QP. They alter the optimal steplength using over and under relaxation parameters between 0 and 2. The authors have shown that by multiplying the optimal step size by a relaxation parameter between 0 and 2, one can improve the performance of the Cauchy or steepest descent method.

Recall that the classical steepest descent method applied to QP is

$$x_{k+1} = x_k - \alpha_k g_k$$

where $g_k = \nabla f(x_k) = Qx_k - b$ and the optimal choice of steplength $\alpha_k$ is given by

$$\alpha_k = \frac{g_k^T g_k}{g_k^T Q g_k}.$$

More specifically, Raydan and Svaiter modify the steepest descent method by introducing over and under relaxation parameters. Taking the relaxation parameters $\theta_k$ as a random scalar with a uniform distribution on[0 , 2], we get

$$x_{k+1} = x_k - \theta_k \alpha_k g_k$$

Notice that for $\theta_k = 1$, we obtain the classical steepest descent method, and also that for $\theta_k = 2$ , $f(x_{k+1}) = f(x_k)$.

Raydan and Svaiter prove in[1] that the sequence of points generated by the RSD method converge to the optimal solution and show that relaxation might be a suitable tool to accelerate the convergence of the steepest descent method . They illustrate their findings with numerical experiments.

**The Cauchy-Barzilai-Borwein method (CBB)**

Motivated by the Barzilai-Borwein method, Raydan and Svaiter propose a combination of the steepest descent method and the Barzilai-Borwein method for which a Q-linear rate of convergence can be established in a suitable norm. The new method is called the Cauchy-Barzilai-Borwein (CBB) method[1], which can be obtained as follows:

$$g_k = \nabla f(x_k) = Qx_k - b,$$

$$h_k = Qg_k,$$

$$t_k = \frac{g_k^T g_k}{g_k^T h_k},$$

$$x_{k+1} = x_k - 2t_k g_k + t_k^2 h_k.$$

They show that the CBB algorithm is much more efficient than the classical steepest descent and the RSD methods using extensive numerical experiments.

# 5. Examples and Numerical Experiments

**Example 5.1**

Consider the QP minimize

$$f(x) = \frac{1}{2} x^T \begin{bmatrix} 21 & 28 & 28 & 17 \\ 28 & 38 & 38 & 21 \\ 28 & 38 & 40 & 22 \\ 17 & 21 & 22 & 19 \end{bmatrix} x - \begin{bmatrix} 10 \\ -212 \\ 303 \\ -417 \end{bmatrix}^T x$$

Our solution is, $x^* = Q^{-1}b = \begin{bmatrix} 9484.124 \\ -7136.624 \\ 1347.687 \\ -2180.374 \end{bmatrix}$, $f(x^*) =$ -1462685.7187.

The eigenvalues of Q are $\lambda_0 = 110.7270$, $\lambda_1 = 5.9953$ $\lambda_2 = 1.2585$, $\lambda_3 = 0.0191$

We begin our algorithm with $x_0 = \begin{bmatrix} -50 \\ -15 \\ 78 \\ 23 \end{bmatrix}$.

**Iteration 0**

$s_0 = -[\ 1095 \quad 1689 \quad 1353 \quad 01405\ ]^T$,

$\alpha_0 = \frac{1}{\lambda_0} = \frac{1}{110.727}$

$x_1 = [\ -59.9 \quad -30.3 \quad 65.8 \quad 10.3\ ]^T$,

$f(x_1) = -16169.9$

$g_1 = [\ -97.6 \quad 101.7 \quad -271.4 \quad 406.64\ ]^T$,

$\|g_1\| = 508.8$

**Iteration 1:**

$s_1 = -[\ -97.6 \quad 101.7 \quad -271.4 \quad 406.64\ ]^T$,

$\alpha_1 = \frac{1}{\lambda_1} = \frac{1}{5.9953}$

$x_2 = [\ -43.6 \quad -47.2 \quad 111.1 \quad -57.5\ ]^T$,

$f(x_2) = -44851.9$

$g_2 = [\ -115.7 \quad 209.5 \quad -140.9 \quad 34.8\ ]^T$,

$\|g_2\| = 279.9$

**Iteration 2:**

$s_2 = -[\ -115.7 \quad 209.5 \quad -140.9 \quad 34.8\ ]^T$,

$\alpha_2 = \frac{1}{\lambda_2} = \frac{1}{1.2585}$

$x_3 = [\ 48.3 \quad -213.7 \quad 223.0 \quad -85.2\ ]^T$,

$f(x_3) = -97075.59$

$g_3 = [\ -180.7 \quad 132.6 \quad -21.5 \quad 40.1\ ]^T$,

$\|g_3\| = 228.7$

**Iteration 3:**

$s_3 = -[\ -180.7 \quad 132.6 \quad -21.5 \quad 40.1\ ]^T$,

$\alpha_3 = \frac{1}{\lambda_3} = \frac{1}{0.0191}$

$x_4 = [\ -9484.1 \quad -7136.6 \quad 1347.7 \quad -2180.4\ ]^T$,

$f(x_4) = -1462685.7187$

$g_4 = [\ 0.0000 \quad 0.0000 \quad 0.0000 \quad 0.0000\ ]^T$,

$\|g_4\| = 0.0000$.

Also for this example the steepest descent method terminated after 14695 iterations with a CPU time of 84.98 seconds compared to a CPU time of 0.22 seconds using the eigenstep method.

**Example 5.2**

In this example in which we illustrate with figure 5.1 we compare the performance of; the steepest descent method, the conjugate gradient method , the CBB method, the RSD method and the eigenstep method (E) in the solution of

$$f(x) = \frac{1}{2}x^T \begin{bmatrix} 2 & 1 \\ 1 & 10 \end{bmatrix} x - [\ -4 \quad -6\ ]\, x.$$

Our minimizer is $x^* = [\ -1.78947 \quad -0.42105\ ]^T$

Note that all 3 methods have the same search direction after the first iteration. The steepest descent method and the conjugate gradient method also have the same step size, $\alpha^*$ and arrive at the same point after the first iteration, however the eigenstep method has a shorter step after the first iteration. Next, the conjugate gradient method and the eigenstep method terminate with $x^*$ after iteration two and the steepest descent method produces a zig zag route to $x^*$ terminating after 31iterations to $x^*$ with a tolerance of $\|g_{30}\| < 10^{-6}$.

The following graph illustrates the CBB RSD and eigenstep methods also applied to example 5.2.

**Numerical experiments for smaller problems**

In table 5.1 we used matlab to randomly generate matrix Q and implement the RSD, CBB, steepest descent (SD), conjugate gradient (CG), and eigenstep (EM) methods for small problems ( $n \le 10$ ). For the eigenstep method, we pre-calculated the eigenvalues of the Q, we then sorted them in descending order and started our algorithm applying the main iterative step (see definition 2.1)starting with the largest eigenvalue. The stop rule is either the number of iterations is greater than 10000 or the norm of ( $x_{k+1} - x_k$ ) is less than the tolerance. The condition number is the ratio of the maximum eigenvalue to the minimum eigenvalue of Q. The data under each method are the number of iterations required to obtain the optimal solution. The CPU times were too small to be relevant.

If the tolerance is not required to be very small (say $10^{-4}$~$10^{-6}$), the number of iterations of the eigenstep and conjugate gradient methods are equal to the dimension of the problem. For most practical applications, these tolerances are adequate.

Theoretically we have proven and we know that the conjugate gradient and eigenstep methods terminate in at most *n* iterations. The reason that finite termination is not satisfied with a very small tolerance (say $10^{-12}$~$10^{-20}$), is that the floating number computation and rounded arithmetic

leads to subtle discrepancies (see[6, p.389]).

**Numerical experiments for large problems**

In table 5.2 we again used Matlab to implement the Barziai-Borwien (BB), RSD, CBB, steepest descent (SD), conjugate gradient (CG), and eigenstep (EM) methods to solve larger problems. We first used matlab to randomly generated the matrix $Q$, the vector $b$ and the initial point $x_0$ and saved this data in a binary data file. Before running the methods, all data are read from the data file to ensure all methods are used to solve the same problem. In each iteration, to obtain good performances for these large problems, we coded a step in the algorithm that ensures that the function value is decreasing. The CPU time of Eigenstep is the sum of the time of iterations and the time of precalculating the eigenvalues. All these experiments were performed by a PC: Intel Pentium 4, CPU 3.2GHz 1.0GB of RAM.

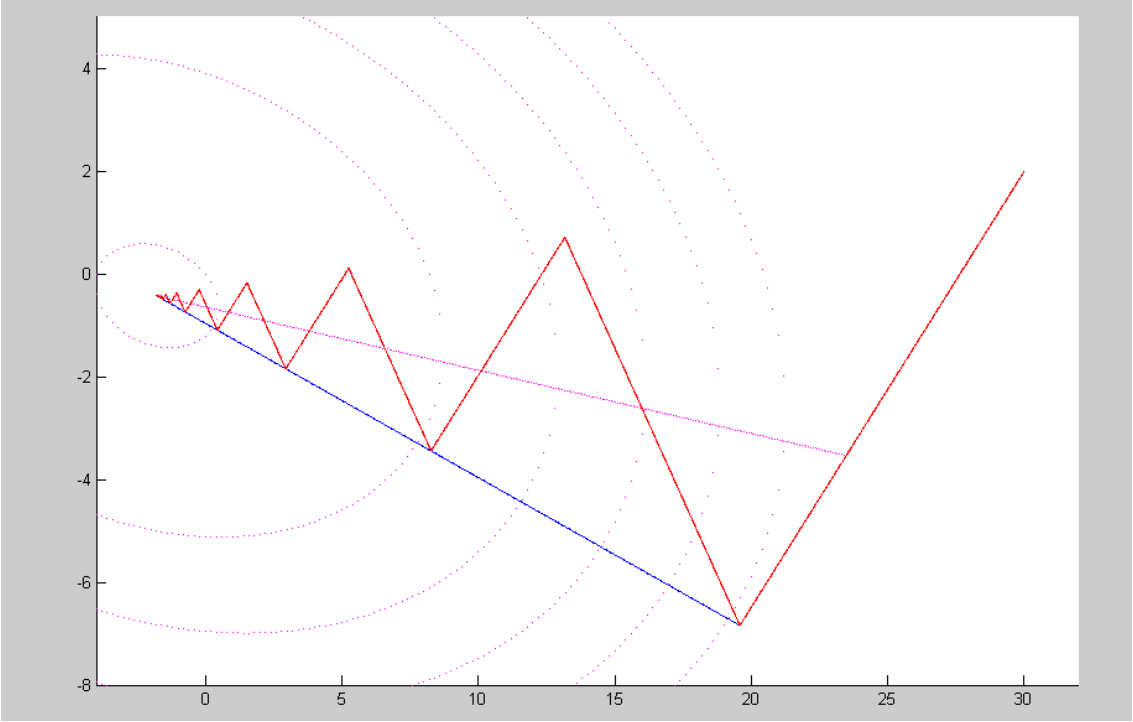


**Figure 5.1.** Steepest descent **(red),** conjugate gradient **(blue),** eigenstep **(purple)** methods

**Table 5.1.** A comparison of five methods with n=2, n=4 and n=10

| Example | Conditio | Tolerance | RSD | CBB | SD | CG | EM |
|---|---|---|---|---|---|---|---|
| 1 n = 2 | 5.4 | 10-4 | 26 | 6 | 37 | 2 | 2 |
| | | 10-6 | 44 | 6 | 49 | 2 | 2 |
| | | 10-12 | 54 | 6 | 85 | 2 | 2 |
| | | 10-20 | * | 7 | 105 | 4 | 7 |
| 2 n = 4 | 372 | 10-4 | 261 | 18 | 954 | 4 | 5 |
| | | 10-6 | 420 | 18 | 1310 | 4 | 5 |
| | | 10-12 | 810 | 19 | 2288 | >10000 | 331 |
| | | 10-20 | >10000 | 25 | >10000 | >10000 | 1667 |
| 3 n = 10 | 836 | 10-4 | 473 | 61 | 1867 | 11 | 12 |
| | | 10-6 | 657 | 69 | 2741 | 11 | 12 |
| | | 10-12 | 1279 | 101 | 5299 | 34 | >10000 |
| | | 10-20 | 1924 | 136 | >10000 | >10000 | >10000 |
| 4 n = 10 | 1000 | 10-4 | 248 | 23 | 113 | 11 | 12 |
| | | 10-6 | 365-649 | 24 | 169 | 11 | 12 |
| | | 10-12 | 1279 | 35 | 341 | 12 | >10000 |
| | | 10-20 | 1913 | 40 | >10000 | 15 | >10000 |
| 5 n = 10 | 1000 | 10-4 | 182 | 15 | 213 | 10 | 12 |
| | | 10-6 | 397 | 19 | 325 | 10 | 12 |
| | | 10-12 | 1009 | 31 | 659 | 11 | 15 |
| | | 10-20 | 2307 | 39 | >10000 | 13 | 16 |

**Table 5.2.** A comparison of six methods with n=100, n=1000, n=2000, n=3000

| | n | Toler | BB | | RSD | | SD | | CBB | | CG | | EM | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | CPU | Iter | CPU | Iter | CPU | Iter | CPU | Iter | CPU | Iter | CPU | Iter |
| 1 | 100 | $10^{-4}$ | 0.000 | 32 | 0.062 | 257 | 0.031 | 84 | 0.000 | 13 | 0.000 | 15 | 0.00+0.00 | 26 |
| 2 | 1000 | $10^{-4}$ | 0.515 | 37 | 3.922 | 285 | 2.078 | 152 | 0.453 | 26 | 0.437 | 31 | 0.20+1.71 | 20 |
| 3 | 1000 | $10^{-4}$ | 0.875 | 61 | 9.062 | 586 | 8.61 | 606 | 0.75 | 43 | 0.578 | 39 | 0.59+1.70 | 59 |
| 4 | 2000 | $10^{-4}$ | 5.734 | 77 | 36.88 | 477 | 63.03 | 818 | 6.609 | 64 | 2.453 | 33 | 3.45+15.297 | 60 |
| 5 | 3000 | $10^{-4}$ | 15.14 | 54 | 100.0 | 529 | 100.4 | 510 | 17.81 | 63 | 6.5 | 31 | 9.1+54.34 | 65 |

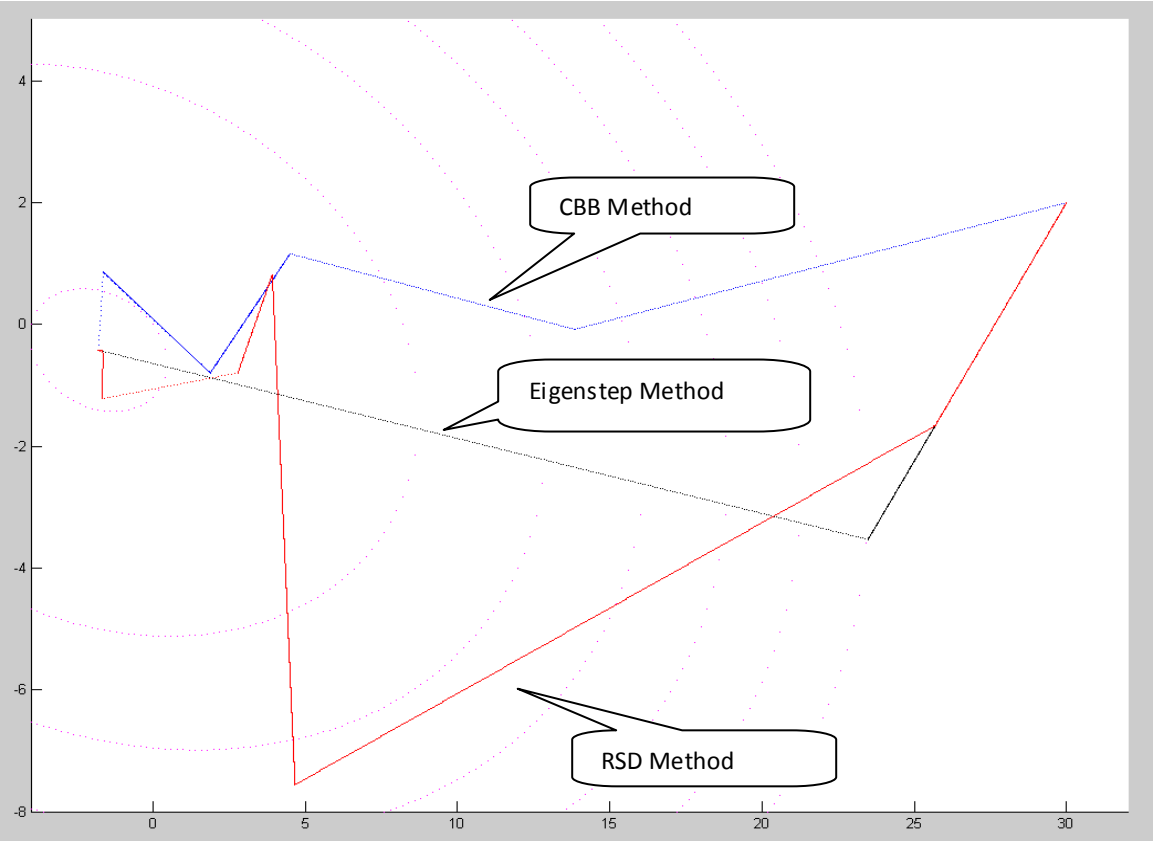


**Figure 5.2.** RSD, CBB and eigenstep methods

# 6. Conclusions

In this paper we presented theorem 2.2 which validates the proposed eigenstep method to solve our model problem. We compared the eigenstep method to other well known first order methods. The conjugate gradient method and the eigenstep method both terminate in at most ***n*** iterations and our experiments show that both methods are superior in performance to the classical steepest descent method. Future work may include modifying the eigenstep method to deal with more general functions. For example, for a function where the Hessian is not constant one might consider using using the reciprocal of the largest eigenvalue as a step size to move efficiently to the next point. In fact modifications of the conjugate gradient method along with step size formulas by Fletcher-Reeves and Polak-Ribiere[6, p.399] are used today to deal with more general problems since these methods have low storage requirements.

# REFERENCES

[1] Raydan, M., and Svaiter,B.F.,"Relaxed steepest descent and Cauchy-Barzilai-Borwein method", *International Journal of Computational Optimization and Applications*, Vol.21, No.2, pp155-167, February 2002.

[2] Barzilai, J., and Borwein,J.M.,"Two point step size gradient method", *IMA Journal of Numerical Analysis*, Vol.8, No.1, pp.141-148, 1988.

[3] Dai,Y.H., and Liao,L.Z.,"R-Linear convergence of the Barzilai and Borwein gradient method", *IMA Journal of Numerical Analysis*, Vol.22 , No.1,pp.1-10, January 2002

[4] Friedlander, A., Martinez, J.M., Molina, B., Raydan, M.,

"Gradient Method With Retards and Generalizations", *SIAM Journal on Numerical Analysis*, Volume 36, number 1, pp.275-289,1999.

[5] Raydan, M., "On the Barzilai and Borwein choice of steplength for the gradient method", *IMA Journal of Numerical Analysis*, Vol.13, No.1,pp. 321-326, 1993.

[6] Nash,S., Sofer,A.," Linear and Nonlinear Programming", McGraw-Hill publishers, 1996

[7] Johnson, L., Riess, R., Arnold, J.," Introduction to Linear Algebra", Addison-Wesley publishers, third edition, 1993.