Solving Trust Region Sub Problems with Combination of Cauchy and Newton Vectors for Unconstrained Optimization Problems

M. SARAJ

Department of Mathematics, Faculty of Mathematical Sciences and Computer
Shahid Chamran University, Ahvaz-Iran

Abstract: The trust region method is used to solve unconstrained optimization problems. On using a positive linear combination of Cauchy and Newton vectors, we could obtain an easy and inexpensive method for solving trust region subproblems. In this method, the linear combination has to be designed in such a way that, in each iteration, the Cauchy coefficient is assumed to be equal to one and then try to calculate and determine Newton coefficient in the sense that the vector magnitude obtained not to exceed the radius of trust region and the reduction obtained is to be more than cauchy’s reduction. We show that this method has global convergence property. For comparing this method with Cauchy method, the numerical results are presented.

Keywords: Trust region, Rank two update, Cauchy method, Newton method, Reduction algorithm family, and Global convergence.

1. INTRODUCTION

Many of optimization problems can be modeled in the form of unconstrained minimization problem, and such problems are very important in optimization. We define the general shape of these problems as follows:

$$\min f(x) : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$$ (1)

There are different methods for solving problem (1) such as: efficient trust region and quasi Newton methods. In trust region methods, to find the kth step in iteration k, we need to minimize a quadratic model of objective function (with considering forced condition). In fact, line search operation is replaced by the forced condition. On solving the following problem known as trust region subproblem, we can find \(p_k\) as kth step.

$$\min m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T G_k p$$

s.t $\|p\| \leq \Delta_k$

where \(f_k = f(x_k), g_k = \nabla f(x_k), G_k = \nabla^2 f(x_k)\) and \(\Delta_k\) is the radius of trust region.

With considering expenses in \(G_k\) which may not be +ve definite, we can therefore substitute \(G_k\) for \(B_k\) to obtain BFGS rank two update. An important property of the BFGS is that, if \(B_k\) is positive definite and \(\delta^y > 0\), then \(B_{k+1}\) is positive definite.
where
\[ \delta = x_{k+1} - x_k, \gamma = \nabla f(x_{k+1}) - \nabla f(x_k). \]

It is well known that in methods which use BFGS update (see, for instance [2,3]), in the case of not satisfying \( \delta^T \gamma > 0 \), we can make it satisfied by using a line search technique satisfied in wolf strong condition while using update. Other Advantage of using \( B_k \) is that, the \( B_k \) computation takes lesser time compare to \( G_k \). So, in iteration \( k \), with this assumption that \( B_k \) is positive definite, we can define the trust region sub problem as follows:

\[
\begin{align*}
\min m_k(p) &= f_k + g_k^T p + \frac{1}{2} p^T B_k p \\
\text{s.t.} \quad \|p\| &\leq \Delta_k
\end{align*}
\]

There are several methods for solving problem (2) which in all of them the obtained step \( p_k \) belongs to reduction algorithm family such as: Cauchy, Dogleg, and Two-dimensional subspace minimization methods (see [1]).

Cauchy method is the most primary method in which \( p_k \) vector is the multiple of \( g_k \).

The cauchy’s algorithm is not used often due to slow convergence practice, although, the Cauchy global convergences in solving problem (2) can be used as a criterion for new methods (see[1]). For increasing convergence’s rate it is needed that \( G_k \) or \( B_k \) to be used in \( p_k \).

In two dimensional subspace minimization, \( p_k \) vector is the linear combination of \( p_k^C \) (Cauchy vector) and \( p_k^N \) (Newton vector). To find Cauchy’s and Newton’s coefficient, a two variables problem must be solved, which is very expensive, so its practical usage is not suggested. Most of the methods for solving problem (2) are special form of this method. For example Cauchy and Dogleg methods. In general, trust region sub problem in two dimensional subspace minimization is as follows:

\[
\begin{align*}
\min m_k(p) &= f_k + g_k^T p + \frac{1}{2} p^T B_k p \\
\text{s.t.} \quad \|p\| &\leq \Delta_k, \quad p = \alpha p_k^C + \beta p_k^N
\end{align*}
\]

The method is called Borza method, and it is the special form of two dimensional subspace minimization problems. In the Borza method, we substitute \( \alpha = 1 \) and then try to find suitable \( \beta \).

The paper is organized as follows:

In section 2: We try to find suitable \( \beta \) that is called \( \beta^* \) and then introduce Borza vector ( \( p_k^{B} \)). In section 3: with the property of \( \beta^* \), the globally convergence of the method will be proved. In section 4: the algorithm of method is defined. Finally; the results of numerical experiments are reported for comparing this method with Cauchy in section 5. It is noticeable that the used norm is Euclidean norm.

2. FINDING METHOD AND VECTOR INTRODUCTION

With this assumption that \( p_k^C, p_k^N \) are Cauchy and Newton vectors in \( k \)th iteration respectively, therefore, the trust region sub problem in \( k \)th iteration which has to be solved is as follows:
\[
\begin{align*}
\min m_k(p) &= f_k + g_k^T p + \frac{1}{2} p^T B_k p \\
\text{s.t.} \quad \|p\| &\leq \Delta_k, \quad p = \alpha p_k^C + \beta p_k^N
\end{align*}
\]

By assuming \( \alpha = 1 \) and then search for suitable \( \beta \) called \( \beta^* \), we change the above problem as follows:

\[
\begin{align*}
\min m_k(p) &= f_k + g_k^T p + \frac{1}{2} p^T B_k p \\
\text{s.t.} \quad \|p_k^C + \beta p_k^N\| &\leq \Delta_k
\end{align*}
\]

To solve problem (4) we replace \( p \) by \( p_k^C, \beta p_k^N \) which converts the problem in to one variable unconstrained quadratic problem in terms of \( \beta \) as follows:

\[
\min m_k(\beta) = f_k + g_k^T (p_k^C + \beta p_k^N) + \frac{1}{2} (p_k^C + \beta p_k^N) B_k (p_k^C + \beta p_k^N)
\]

Which is further reduced to

\[
\min m_k(\beta) = f_k + g_k^T p_k^C + \frac{1}{2} p_k^C B_k p_k^C + \left(g_k^T p_k^N + p_k^C B_k p_k^N\right) \beta + \frac{1}{2} \beta^2 \left(p_k^N B_k p_k^N\right)
\]

since \( B_k \) is positive definite, the coefficient \( \beta^* \) is positive, therefore, \( m_k(\beta) \) function has global minimum called \( \beta^* \) which is given by

\[
\beta^* = \frac{-g_k^T p_k^N + g_k^T B_k p_k^N}{g_k^T B_k p_k^N}
\]

**Lemma**: The \( \beta^* \) which defined by (6) is not negative.

**Proof**: Since the linear combination is occurred, when \( \|p_k^C\| < \Delta_k \), \( p_k^C = -\frac{\|g_k\|^2}{g_k^T B_k g_k} g_k \) and, if \( \|p_k^C\| < \Delta_k \) then \( \|p_k^C\| = p_k^C \).

\[
g_k^T p_k^N + p_k^C B_k p_k^N = -g_k^T B_k^{-1} g_k + \frac{\|g_k\|^2}{g_k^T B_k g_k} g_k^T B_k B_k^{-1} g_k = -g_k^T B_k^{-1} g_k + \frac{\|g_k\|^4}{g_k^T B_k g_k} \leq 0
\]

where the final inequality follows from [1].

The above \( \beta^* \) will be the solution of (4) if

\[
\|p_k^C + \beta^* p_k^N\| \leq \Delta_k
\]
According to

\[ \left\| p_k^C + \beta^* p_k^N \right\| \leq \left\| p_k^C \right\| + \beta^* \left\| p_k^N \right\| \]  

(8)

And to find a trust region for \( \beta^* \), we let \( \left\| p_k^C \right\| + \beta^* \left\| p_k^N \right\| \leq \Delta_k \) which replies

\[ 0 \leq \beta^* \leq \frac{(\Delta_k - \left\| p_k^C \right\|)}{\left\| p_k^N \right\|} \]  

(9)

Figure 1: The Representation of Cauchy & Newton Vectors with their Linear Combination

The Borza vector in iteration \( k \) is as follows:

\[
p_k^B = \begin{cases} 
  p_k^C + \beta^* p_k^N & \text{if } 0 \leq \beta^* \leq \frac{(\Delta_k - \left\| p_k^C \right\|)}{\left\| p_k^N \right\|} \\
  p_k^C & \text{otherwise}
\end{cases}
\]  

(10)

3. ALGORITHM

Given \( \bar{\Delta}, \Delta_0 \in (0, \bar{\Delta}) \) and for \( \eta \in \left[ 0, \frac{1}{4} \right) \)

For \( k = 0, 1, 2, \ldots \)

Compute \( p_k^B \)

\[
p_k^B = \frac{f(x_k) - f(x_{k+1})}{m_k(0) - m_k(p_k^B)}
\]
if \( \rho_k < \frac{1}{4} \)

\[ \Delta_{k+1} = \frac{1}{4} \| p_k^\theta \| \]

else

if \( \rho_k \geq \frac{3}{4} \) and \( \| p_k^\theta \| \)

\[ \Delta_{k+1} = \min(2\Delta_k, \Delta) \]

else

\[ \Delta_{k+1} = \Delta_k \]

if \( \rho_k > \eta \)

\[ x_{k+1} = x_k + p_k^\theta \]

else

\[ x_{k+1} = x_k \]

end (for)

**4. GLOBAL CONVERGENCE**

The following two, are the sufficient condition for global convergence [1].

\[ \| p_k^\theta \| \leq \Delta_k \] (11)

\[ m_k(p_k^\theta) \leq m_k(p_k^C) \] (12)

From (7), (8), (9), and since we have

\[ \| p_k^\theta \| = \| p_k^C + \beta^* p_k^N \| \leq \Delta_k . \]

And to prove (12) we first find all \( \beta \) which satisfies in given relation and, then we show that \( \beta^* \) belongs to their family.

\[ m_k(p_k^C + \beta p_k^N) \leq m_k(p_k^C) \Rightarrow f_k + g_k^T (p_k^C + \beta p_k^N) + \frac{1}{2} (p_k^C + \beta p_k^N)^T B_k (p_k^C + \beta p_k^N) \leq f_k + g_k^T p_k^C + \frac{1}{2} p_k^C^T B_k p_k^C \Rightarrow \]

\[ \frac{1}{2} \beta^2 p_k^N^T B_k p_k^N \beta + g_k^T p_k^N + p_k^C^T B_k p_k^N \leq 0 \Rightarrow \]

\[ 0 \leq \beta \leq \frac{-2(g_k^T p_k^N + p_k^C^T B_k p_k^N)}{p_k^N^T B_k p_k^N} \] (13)
By comparing (6), (13) it is concluded that:
\[ m_k(p_k^b) = m_k(p_k^c + \beta^* p_k^N) \leq m_k(p_k^c). \]

5. NUMERICAL RESULTS

In this section, we present the numerical results for the suggested algorithm. All the computations are performed on MATLAB. We compare the Borza method with Cauchy method. $\eta = 0$, $\Delta = \infty$, and the computation terminates when $\|g_i\| \leq 10^{-6}$.

**Example** (Powell function)

\[ f(x) = x_1^4 + x_1 x_2 + (1 + x_2)^2 \]

<table>
<thead>
<tr>
<th>Initial point</th>
<th>Optimized point</th>
<th>Cauchy iteration</th>
<th>Borza iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1.1]</td>
<td>[.06959, -1.3479]</td>
<td>20</td>
<td>16</td>
</tr>
<tr>
<td>[10,10]</td>
<td>[.06959, -1.3479]</td>
<td>38</td>
<td>22</td>
</tr>
<tr>
<td>[100,100]</td>
<td>[.06959, -1.3479]</td>
<td>155</td>
<td>38</td>
</tr>
<tr>
<td>[200, -200]</td>
<td>[.06959, -1.3479]</td>
<td>407</td>
<td>36</td>
</tr>
</tbody>
</table>

Tables of Comparing Cauchy with Borza Method for Different Iterations.

REFERENCES