Trust Regions

Charles J. Geyer

October 22, 2005

1 Trust Region Theory

We follow Nocedal and Wright (1999, Chapter 4), using their notation. Fletcher (1987, Section 5.1) discusses the same algorithm, but with slight differences.

1.1 Main Loop

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a function we wish to minimize, and we do so by producing a sequence x_1, x_2, ... of points in $\mathbb{R}^n$ that converge to a solution.

This sequence of iterates is produced by the trust region main loop which repeatedly solves the *trust region subproblem*. First we form the Taylor series expansion of f about x_k

$$m_k(p) = f_k + g_k^T p + \tfrac{1}{2} p^T B_k p$$

where

$$\begin{aligned} f_k &= f(x_k) \\ g_k &= \nabla f(x_k) \\ B_k &= \nabla^2 f(x_k) \end{aligned}$$

and the idea is we "trust" the Taylor series expansion only so far, so we minimize m_k subject to the constraint $\|p\| \le \Delta_k$ where $\|\cdot\|$ denotes the Euclidean norm.

Suppose p_k is a solution to the trust region subproblem (about which more in Section 1.2 below). The adjustment of Δ_k is done as follows (Nocedal and Wright, 1999, Algorithm 4.1) based on

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(x_k) - m_k(x_k + p_k)} \tag{1}$$

which is the actual decrease in the objective function f in the step compared to the predicted decrease using m_k. If ρ_k is small or negative, then m_k is a bad model at $x_k + p_k$ so the step should not be used and the trust region radius should be adjusted. The complete trust region main loop body is

if $(\rho_k < 1/4)$ **then**
    $x_{k+1} := x_k$
    $\Delta_{k+1} = \|p_k\|/4$
else
    $x_{k+1} := x_k + p_k$
    if $(\rho_k > 3/4$ and $\|p_k\| = \Delta_k)$ **then**
        $\Delta_{k+1} = \min(2\Delta_k, \bar{\Delta})$
    fi
fi

where $\bar{\Delta}$ is the maximum allowed Δ_k.

1.2 Trust Region Subproblem

Now we follow Section 4.2 of Nocedal and Wright (1999). We drop subscripts writing m instead of m_k and so forth. So the subproblem is

$$\begin{aligned} \text{minimize} \quad & m(p) \stackrel{\text{def}}{=} f + g^T p + \tfrac{1}{2} p^T B p \\ \text{subject to} \quad & \|p\| \leq \Delta \end{aligned}$$

which minimizes a quadratic function over the closed ball of radius Δ centered at the origin.

A vector p^* is a global solution to the trust region subproblem if and only if

$$\|p^*\| \leq \Delta \tag{2a}$$

and there exists a scalar $\lambda \geq 0$ such that

$$(B + \lambda I) p^* = -g \tag{2b}$$

$$\lambda = 0 \quad \text{or} \quad \|p^*\| = \Delta \tag{2c}$$

$$B + \lambda I \text{ is positive semidefinite} \tag{2d}$$

(Nocedal and Wright, 1999, Theorem 4.3).

1.2.1 Unconstrained Solutions

The $\lambda = 0$ case is easy. If B is positive definite and $p^* = -B^{-1} g$ satisfies (2a), then that is the solution.

1.2.2 Constrained Solutions

The $\|p^*\| = \Delta$ case is more complicated. Define

$$p(\lambda) = -(B + \lambda I)^{-1} g = -\sum_{j=1}^{n} \frac{q_j^T g}{\lambda_j + \lambda} q_j \tag{3}$$

where λ_j are the eigenvalues of B and q_j are the corresponding orthonormal eigenvectors (this is valid only when $\lambda \neq -\lambda_j$ for all j). Then

$$\|p(\lambda)\|^2 = \sum_{j=1}^{n} \left(\frac{q_j^T g}{\lambda_j + \lambda} \right)^2 \tag{4}$$

The analysis again splits into several cases.

The Easy Case Let $\lambda_{\min}$ denote the minimum eigenvalue of B. If

$$(q_j^T g) \neq 0, \qquad \text{for some } j \text{ such that } \lambda_j = \lambda_{\min} \tag{5}$$

then $p(\lambda)$ is a continuous, strictly decreasing function on the open interval $(-\lambda_{\min}, \infty)$ and goes to ∞ as $\lambda \to -\lambda_{\min}$ and to zero as $\lambda \to \infty$. Thus a λ^* such that $\|p(\lambda^*)\|^2 = \Delta^2$ exists and is unique, equation (4) can be used to find it, and $p(\lambda^*)$ is the solution to the trust region subproblem.

The Hard Case In the other case, when (5) is false, (4) is continuous at $\lambda = \lambda_{\min}$ and now defines a continuous strictly decreasing function on the closed interval $[-\lambda_{\min}, \infty)$, and λ must be in this interval in order for (2d) to hold.

Now the analysis splits into two subcases yet again.

Hard Case: Easy Part If (5) is false but

$$\|p(-\lambda_{\min})\|^2 > \Delta^2, \tag{6}$$

then there is still a unique λ^* in $(-\lambda_{\min}, \infty)$ such that $\|p(\lambda^*)\|^2 = \Delta^2$, equation (4) can be used to find it, and $p(\lambda^*)$ is the solution to the trust region subproblem.

Hard Case: Hard Part Otherwise, when (5) and (6) are both false, we must have $\lambda = -\lambda_{\min}$. Then $B + \lambda I$ is singular, and (3) can no longer be used. Now solutions of (2b) are non-unique, and we have

$$p_{\text{hard}}(\tau) = - \sum_{\substack{j=1 \\ \lambda_j \neq \lambda_{\min}}}^{n} \frac{q_j^T g}{\lambda_j - \lambda_{\min}} q_j + \sum_{\substack{j=1 \\ \lambda_j = \lambda_{\min}}}^{n} \tau_j q_j \tag{7}$$

is a solution of (2b) for every vector τ. Since the first sum on the right hand side of (7) has norm less than Δ by the falsity of (6), we can choose τ to make $\|p_{\text{hard}}(\tau)\| = \Delta$.

Note that the solution p^* obtained in this case is non-unique. Even if there is only one term in the second sum in (7), τ_j of opposite signs produce $p_{\text{hard}}(\tau)$ of the same length. When there is more than one term in the second sum in (7), there are even more possibilities of nonuniqueness. This nonuniqueness is not an issue, because (at least as far as the subproblem is concerned) one solution is just as good as another.

1.2.3 Numerical Stability

We need to examine how all this case splitting works when the arithmetic is inexact (as it is in a computer). Let us take the eigendecomposition of B that gives us equation (3) or equation (7) as authoritative. We know the eigendecomposition is inexact, but we do not attempt to correct for its inexactness.

If all of the λ_j as calculated (inexactly) by the eigendecomposition routine are strictly positive, then (3) with $\lambda = 0$ gives an "unconstrained" solution, and this solution has squared norm (4) that is either greater than Δ^2 or not. If not, we have found an inexact solution. If greater, we decide to impose the constraint. This decision is stable in the sense that we will not want to undo it later.

With inexact arithmetic, we are unlikely ever to get the "hard" case. Nevertheless, we allow for the possibility. Define

$$C_1 = \sum_{\substack{j=1 \\ \lambda_j \neq \lambda_{\min}}}^{n} \left(\frac{q_j^T g}{\lambda_j - \lambda_{\min}} \right)^2 \tag{8a}$$

$$C_2 = \sum_{\substack{j=1 \\ \lambda_j = \lambda_{\min}}}^{n} \left(q_j^T g \right)^2 \tag{8b}$$

Then the constrained cases are distinguished as follows.

- *easy case:* $C_2 \neq 0$.
- *hard-easy case:* $C_2 = 0$ and $C_1 > \Delta^2$.
- *hard-hard case:* $C_2 = 0$ and $C_1 \leq \Delta^2$.

The "hard-hard case" is now obvious. τ is adjusted so that the second term on the right hand side in (7) has squared length $\Delta^2 - C_1$.

In the other two cases we must find a zero of the function of λ given by

$$\|p(\lambda)\|^2 - \Delta \tag{9a}$$

or (what is equivalent) a zero of the function defined by

$$\phi(\lambda) = \frac{1}{\|p(\lambda)\|} - \frac{1}{\Delta} \tag{9b}$$

which is better behaved (Nocedal and Wright, 1999, Chapter 4). Both are monotone, (9a) strictly decreasing and (9b) strictly increasing, but (9b) is also nearly linear.

We would like to bracket the zero, finding an interval containing the zero. We know that $\lambda = -\lambda_{\min}$ is a lower bound. We have

$$\phi(-\lambda_{\min}) = \frac{1}{\sqrt{C_1}} - \frac{1}{\Delta} < 0$$

in the "hard-easy case" and

$$\phi(-\lambda_{\min}) = -\frac{1}{\Delta}$$

in the "easy case". A better lower bound uses

$$\|p(\lambda)\|^2 \geq \frac{C_2}{(\lambda_{\min} + \lambda)^2}$$

from which (setting the right hand side equal to Δ^2 and solving for λ) we get

$$\lambda_{\text{dn}} = \frac{\sqrt{C_2}}{\Delta} - \lambda_{\min}$$

as a lower bound for the λ that makes $\|p(\lambda)\| = \Delta$.

To get an upper bound, we note that if we define

$$C_3 = \sum_{j=1}^{n} \left(q_j^T g \right)^2 \tag{10}$$

then we have

$$\|p(\lambda)\|^2 \le \frac{C_3}{(\lambda_{\min} + \lambda)^2}$$

Setting the right hand side equal to Δ^2 and solving for λ gives

$$\lambda_{\text{up}} = \frac{\sqrt{C_3}}{\Delta} - \lambda_{\min}$$

as an upper bound. So now we know there is a solution in $[\lambda_{\text{dn}}, \lambda_{\text{up}}]$.

1.3 Rescaling

When the variables are ill-scaled, the trust region is badly designed and Nocedal and Wright (1999, Section 4.4) recommend changing the trust region subproblem to

$$\begin{aligned} \text{minimize} \quad & m(p) \stackrel{\text{def}}{=} f + g^T p + \tfrac{1}{2} p^T B p \\ \text{subject to} \quad & \|Dp\| \le \Delta \end{aligned} \tag{11}$$

where D is a strictly positive definite diagonal matrix (as will be seen below, D can be any invertible matrix).

We claim that this is equivalent to running our original algorithm (with no D or, what is equivalent, with D the identity matrix) on the function $\tilde{f}$ defined by

$$\tilde{f}(\tilde{x}) = f(D^{-1} \tilde{x})$$

Consider a point $\tilde{x}_k = D x_k$. Here, if g and B denote the gradient and Hessian of f, then

$$\tilde{g} = D^{-1} g \tag{12a}$$

$$\widetilde{B} = D^{-1} B D^{-1} \tag{12b}$$

are the gradient and Hessian of f. The trust region subproblem (of the original kind with no D) for $\tilde{f}$ (for an iterate centered at $\tilde{x}_k$ using trust region "radius" Δ_k) is

$$\begin{aligned} \text{minimize} \quad & \widetilde{m}(w) \stackrel{\text{def}}{=} f + \tilde{g}^T w + \tfrac{1}{2} w^T \widetilde{B} w \\ & \qquad\quad f + g^T D^{-1} w + \tfrac{1}{2} w^T D^{-1} B D^{-1} w \\ \text{subject to} \quad & \|w\| \le \Delta_k \end{aligned} \tag{13}$$

Let w^* be a solution to (13), and define $p^* = D^{-1}w^*$. Then p^* solves (11).

Let x_1 is the starting point for the trust region problem of minimizing f using rescaling D. Define $\tilde{x}_1 = D^{-1}x_1$, and consider it the starting point for the trust region problem of minimizing $\tilde{f}$ using no rescaling. Let $\tilde{x}_1$, $\tilde{x}_2$, ... be the sequence of iterates produced in the latter problem. Then we have

$$\begin{aligned} \tilde{x}_{k+1} - \tilde{x}_k &= w_k^* \\ &= Dp_k^* \\ &= D(x_{k+1} - x_k) \end{aligned}$$

whenever we have an accepted step (so $x_{k+1} \neq x_k$).

2 Termination Criteria

Although not part of the trust region algorithm proper, termination criteria are important. Fletcher (1987) discusses them on pp. 22–23 and 57 ff. The conventional criteria are to terminate when any of

$$f(x_k) - f(x_{k+1}) \tag{14a}$$

$$x_k - x_{k+1} \tag{14b}$$

$$\nabla f(x_k) \tag{14c}$$

are small. All are used in practice. Note that (14b) and (14c) are vectors, so what "small" means for them is more complicated.

Fletcher (1987, pp. 57 ff.) makes the point that it is desirable that an optimization algorithm be invariant under rescaling, perhaps under arbitrary affine transformation of the domain of the objective function, perhaps only under diagonal scaling. A trust region algorithm is not invariant under rescaling unless the scaling matrix D introduced in Section 1.3 exactly matches the scaling. Nevertheless, invariant convergence tests still make sense and only (14a) among those considered above is invariant.

Fletcher also suggests

$$(x_{k+1} - x_k)^T \nabla f(x^k) \tag{14d}$$

which is invariant when the step is a Newton step.

It is also possible for $\Delta_k \to 0$ as $k \to \infty$ (see Fletcher, 1987, proof of Theorem 5.1.1). Thus it is also necessary to consider termination when

$$\Delta_k \tag{14e}$$

is small.

3 Algorithm

3.1 Order of Operations

The algorithm has one main loop and (as described) does

- one objective function evaluation (value, gradient, and Hessian) and
- one adjustment of Δ_k and x_k

per iteration. The main tricky bit is to decide where in the loop each part of the computation occurs. We can break the body of the loop into the following parts.

- Solve trust region subproblem.
- Evaluate objective function and derivatives at $x_k + p_k$.
- Adjust current point x_k and trust radius Δ_k.
- Test for termination with x_{k+1} the solution.

The following considerations influence the order in which these steps are done.

- Evaluation of $f(x_k + p_k)$ must come between the solution of the subproblem (which produces p_k) and the adjustment of x_k and Δ_k, because the adjustment depends on (1), which depends on p_k.
- Thus all that remains to be decided is where in the Solve-Evaluate-Adjust cycle to put the termination test.
  - Solve-Test-Evaluate-Adjust and Solve-Evaluate-Test-Adjust are senseless, because we can't decide that $x_k + p_k$ is the solution until after we decide whether or not to set $x_{k+1} = x_k + p_k$ in the Adjust step, and it makes no sense to do the work of producing p_k and not let $x_k + p_k$ be considered for the solution.
  - Thus the order should be Solve-Evaluate-Adjust-Test as in the list above.

3.2 Termination Test

Criteria for termination.

(a) The change in the objective function value $|f(x_k) - f(x_{k+1})|$ is less than the termination criterion ϵ_1 in any step.

(b) The change in the second-order Taylor-series model of the objective function value $\left|(x_k - x_{k+1})^T \left(g_k + \frac{1}{2} B_k (x_k - x_{k-1})\right)\right|$ is less than the termination criterion ϵ_3 in any step.

(c) The trust region radius Δ_k has shrunk to less than the termination criterion ϵ_4 in any step.

Condition (b) is also invariant under affine transformations of the parameter space. It seems to make more sense than using the first-order series condition (14d) discussed above, especially since we are already calculating it, because it is the denominator in (1).

Condition (c) appears to be redundant, since when the trust region shrinks to too small, this will force (a) and (b) to be small too.

Now we have a nice unification of ideas. The object tested in (a) is the numerator in (1), the object tested in (b) is the denominator in (1). When either is small, the ratio (1) may be wildly erroneous due to inexact computer arithmetic, but then we terminate the algorithm anyway.

However, this analysis tells us that our separation of work to do into "Adjust" and "Test" steps is erroneous. When we should terminate, we cannot adjust because we cannot (or should not) calculate the "Adjust" criterion (1). The reason why we originally wanted "Test" after "Adjust" is that we thought that it made no sense to terminate when the "Adjust" procedure rejects the step. But now we see that if it rejects the step based on a value of (1) that is reliable, then we won't terminate anyway because we have decided to terminate if and only if we declare (1) unreliable.

References

Fletcher, R. (1987). *Practical Methods of Optimization*, second edition. John Wiley.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. Springer-Verlag.