Journal of Computational Mathematics, Vol.21, No.2, 2003, 229–236.

A SELF-ADAPTIVE TRUST REGION ALGORITHM*[1)]

Long Hei†
(*Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Sciences, Chinese Academy of Sciences, Beijing 100080, China*)
(*Department of Industrial Engineering and Management Sciences Northwestern University C230, 2145 Sheridan Road Evanston, Illinois 60208, USA*)

Abstract

In this paper we propose a self-adaptive trust region algorithm. The trust region radius is updated at a variable rate according to the ratio between the actual reduction and the predicted reduction of the objective function, rather than by simply enlarging or reducing the original trust region radius at a constant rate. We show that this new algorithm preserves the strong convergence property of traditional trust region methods. Numerical results are also presented.

Key words: Trust region, Unconstrained optimization, Nonlinear optimization.

1. Introduction

In this paper we study a new type of trust region method for solving the following unconstrained optimization problem:

$$\min_{x\in\Re^n} \quad f(x). \tag{1.1}$$

A trust region method calculates a trial step d_k by solving the trust region subproblem

$$\min_{d\in\Re^n} \quad \Phi_k(d) := g_k^T d + \frac{1}{2} d^T B_k d \tag{1.2}$$

$$s.t. \quad ||d||_2 \leq \Delta_k, \tag{1.3}$$

where $g_k = \nabla f(x_k)$ is the gradient of the objective function at the current approximate solution x_k, B_k is an $n \times n$ symmetric matrix approximating the Hessian of $f(x)$, and $\Delta_k > 0$ is the current trust region radius. Compared with the line search methods, one of the most important advantages of trust region methods is that B_k is allowed to be indefinite.

After obtaining a trial step d_k, which is an exact or approximate solution of subproblem (1.2)-(1.3), a trust region method computes the ratio ρ_k between the actual reduction in the objective function and the predicted reduction in the quadratic model of the objective function, that is,

$$\rho_k := \frac{Ared_k}{Pred_k} \tag{1.4}$$

$$= \frac{f(x_k) - f(x_k + d_k)}{\Phi_k(0) - \Phi_k(d_k)}. \tag{1.5}$$

Then the trust region radius Δ_k is updated according to the value of ρ_k. The common method for updating Δ_k is to enlarge it by a constant time (say, double):

$$\Delta_{k+1} = \beta_1 \Delta_k \qquad (\beta_1 > 1), \tag{1.6}$$

* Received July 1, 2000.
1) Supported partially by Chinese NSF grant.
† Email address: lhei@neruda.ece.nwu.edu.

if ρ_k is *satisfactory* enough, and to reduce it by a constant fraction (say, a half):

$$\Delta_{k+1} = \beta_2 \Delta_k \qquad (0 < \beta_2 < 1), \tag{1.7}$$

in the case ρ_k is not *satisfactory* enough. Now, if the trial step d_k is successful, one then accepts this step, and sets $x_{k+1} = x_k + d_k$; otherwise, the step d_k is rejected.

The trust region algorithm stated above is often used to solve problem (1.1). It converges globally and superlinearly. However, in the course of updating the trust region radius Δ_k, we do not make full use of the ratio ρ_k. In fact, the value of ρ_k, in some degree, reflects the extent to which the quadratic model $\Phi_k(d)$ approximates the objective function $f(x)$. Our goal in this paper is to design an algorithm in which Δ_k is updated at a variable rate according to the value of ρ_k directly.

2. Ideal Trust Region and R-Function

Now let us reconsider the idea of trust region algorithms. At the current solution x_k, if the trial step d_k is successful and the ratio ρ_k is satisfactory, one accepts the trial step and enlarges the trust region radius. On the contrary, if the trial step d_k is not successful and the ratio ρ_k is not satisfactory, d_k is rejected and Δ_k is reduced. As the ratio between the actual reduction $Ared_k$ and the predicted reduction $Pred_k$, ρ_k reflects the extent to which we are satisfied with the solution d_k of the subproblem (1.2)-(1.3), or to say, the extent to which the quadratic model $\Phi_k(d)$ approximates the original objective function $f(x)$.

We now think about the extreme case when ρ_k is $+\infty$, which means the computed step d_k is very successful. At this time we may, from the idealized point of view, enlarge the trust region radius Δ_k greatly, even to $+\infty$. In the other extreme case, say, when ρ_k is $-\infty$, which implies that the trial step d_k is so *bad* that it causes the objective function value to rise rapidly, it is then reasonable for us to imagine the trust region radius Δ_k should be reduced to a very small value, even near 0. These ideal cases, which we call *ideal trust region*, inspire us to study the following type of functions of ρ_k, named *R-function*:

Definition *Any one-dimensional function $R_\eta(t)$ that is defined in $\Re = (-\infty, +\infty)$ with the parameter $\eta \in (0, 1)$ is an* R-function *if and only if it satisfies:*
(i) *$R_\eta(t)$ is non-decreasing in $(-\infty, +\infty)$;*
(ii)

$$\lim_{t \to -\infty} R_\eta(t) = \beta \qquad (\textit{where } \beta \in [0,1) \textit{ is a small constant}); \tag{2.1}$$

(iii)

$$R_\eta(t) \le 1 - \gamma_1 \quad (\textit{for all } t < \eta, \textit{ where } \gamma_1 \in (0, 1-\beta) \textit{ is a constant}); \tag{2.2}$$

(iv)

$$R_\eta(\eta) = 1 + \gamma_2 \qquad (\textit{where } \gamma_2 \in (0, +\infty) \textit{ is a constant}); \tag{2.3}$$

(v)

$$\lim_{t \to +\infty} R_\eta(t) = M \qquad (\textit{where } M \in (1+\gamma_2, +\infty) \textit{ is a constant}). \tag{2.4}$$

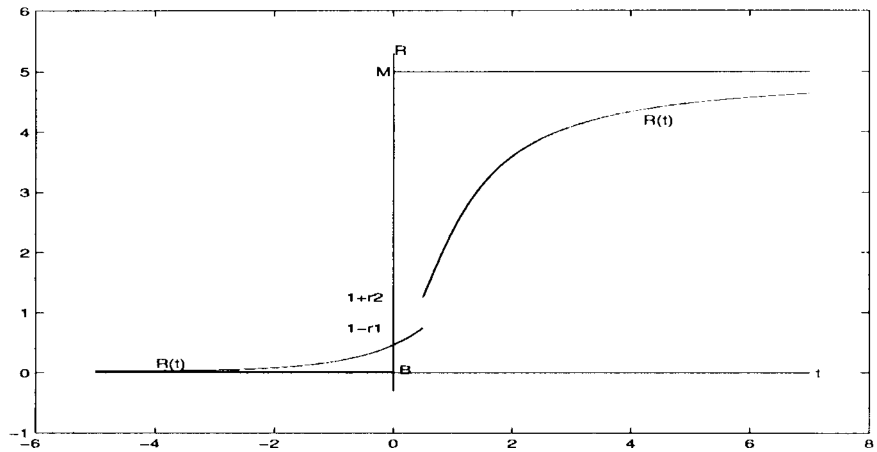
From this definition we can easily see some properties of R-functions:

Theorem 2.2. *An R-function $R_\eta(t)$ (where $\eta \in (0,1)$) satisfies:*

$$0 < \beta \le R_\eta(t) \le 1 - \gamma_1 < 1, \qquad \forall t \in (-\infty, \eta); \tag{2.5}$$

$$1 < 1 + \gamma_2 \le R_\eta(t) \le M < +\infty, \qquad \forall t \in [\eta, +\infty). \tag{2.6}$$

This theorem can be easily proven by the definition of R-functions. A typical figure of an R-function is shown in the following image. Due to Theorem 2.2, we can use $R_\eta(\rho_k)$ as the standard of enlarging or reducing the trust region radius. Just as we have already mentioned, the scale to which one updates the trust region radius is determined by the problem itself, so we call our new algorithm the *self-adaptive trust region algorithm*. As we will show later, Theorem 2.2 is important when we prove the convergence of the new algorithm.



3. The Self-Adaptive Trust Region Algorithm

We describe the self-adaptive trust region algorithm (SATR) as following:

Algorithm 3.1. *(SATR)*

Step 1. *Give* $x_1 \in \Re^n$, $B_1 \in \Re^{n\times n}$, $\Delta_1 > 0$, $\epsilon \geq 0$, $0 < \beta < 1$, $0 < \gamma_1 < 1-\beta$, $\gamma_2 > 0$, $M > 1+\gamma_2$, $c_1 > 1$, $0 < c_2 < 1$, *set* $k = 0$;

Step 2. *If* $||g_k||_2 \leq \epsilon$, *stop. Otherwise solve subproblem (1.2)-(1.3) for* d_k;

Step 3. *Compute* $\rho_k = \frac{Ared_k}{Pred_k}$. *Set*

$$x_{k+1} = \begin{cases} x_k + d_k, & \text{if } \rho_k > c_1; \\ x_k, & \text{otherwise} \end{cases} \tag{3.1}$$

and

$$\Delta_{k+1} = R_{c_2}(\rho_k)||d_k||_2. \tag{3.2}$$

Step 4. *Update* B_k, *set* $k = k+1$. *Go to Step 2.*

The commonly used values of c_1 and c_2 are, respectively, 0 and 0.25. c_1 is usually set as 0 because it causes any trial step d_k to be accepted as long as d_k reduces the objective function value, thereby ensuring that our algorithm will never reject any *good* point whose function value has been computed. This technique is especially important in the case when function values of $f(x)$ are difficult to compute. c_2 is the standard of judging whether ρ_k is satisfactory enough or not. Alternative methods to give c_1 and c_2 are mentioned in [1]. One can also see [6], [7], [8],[9] for help. In the algorithm (SATR), the parameters β, M, γ_1, γ_2, ϵ are give by users.

It is easy to see that the traditional trust region algorithm is equivalent to the case when we set

$$R_{c_2}(\rho_k) = \begin{cases} \beta_2 \in (0,1), & \text{if } \rho_k < c_2; \\ \beta_1 \in (1,+\infty), & \text{otherwise.} \end{cases} \tag{3.3}$$

4. The Global Convergence Property of Algorithm (SATR)

To analyze the new algorithm (SATR) we make the following assumptions:

Assumption 4.1.

(**i**) *The sequence $\{x_k\}$ generated by Algorithm 3.1 is bounded, that is,*

$$x_k \in S \tag{4.1}$$

for all k, where S is a closed convex set in $\Re^n$;

(**ii**) *$\nabla f(x)$ is uniformly continuous in $\Re^n$;*

(**iii**) *The solution d_k of subproblem (1.2)-(1.3) satisfies:*

$$\Phi_k(0) - \Phi_k(d_k) \geq \delta ||g_k||_2 \min \left[\Delta_k, \frac{||g_k||_2}{1 + ||B_k||_2}\right], \tag{4.2}$$

where δ is a positive constant;

(**iv**)

$$\sum_{k=1}^{+\infty} \frac{1}{M_k} = +\infty, \tag{4.3}$$

where

$$M_k := \max_{1 \leq i \leq k} ||B_i||_2 + 1. \tag{4.4}$$

As we will see, the proof to the convergence of Algorithm 3.1 is mainly based on inequality (4.2), which we call the *sufficient model reduction condition.* In fact Powell (in [4]) has proved that the exact solution d_k of subproblem (1.2)-(1.3) satisfies (4.2) with $\delta = \frac{1}{2}$. In [5] Nocedal and Yuan also presented an algorithm, by using which the approximate solution d_k of (1.2)-(1.3) would satisfy (4.2). Here we regard inequality (4.2) as one of our basic assumptions.

Under Assumption 4.1, we will prove that the sequence $\{x_k\}$ generated by Algorithm 3.1 is globally convergent in the sense that

$$\liminf_{k \to +\infty} ||g_k||_2 = 0. \tag{4.5}$$

We proceed by contradiction. If (4.5) were not true, there would exist a constant $\epsilon > 0$ such that $||g_k||_2 \geq \epsilon > 0$. First, we have the following lemma:

Lemma 4.2. *If $g(x)$ is uniformly continuous and the sequence $\{x_k\}$ generated by Algorithm 3.1 satisfies*

$$||g_k||_2 \geq \epsilon > 0, \tag{4.6}$$

where $\epsilon > 0$ is a constant, then there exists a constant $\gamma > 0$ such that

$$||d_k||_2 \geq \frac{\gamma}{M_k} \qquad (k = 1, 2, 3, ...), \tag{4.7}$$

where M_k is defined by (4.4).

Proof. If the lemma were not true, there would exist a subsequence $\{k_i\}$ such that

$$||d_{k_i}||_2 M_{k_i} \to 0, \quad i \to +\infty. \tag{4.8}$$

The above limit and (4.4) show that

$$\begin{aligned} \Phi_{k_i}(0) - \Phi_{k_i}(d_{k_i}) &= -g_{k_i}^T d_{k_i} - \frac{1}{2} d_{k_i}^T B_{k_i} d_{k_i} \\ &= O(||d_{k_i}||_2). \end{aligned} \tag{4.9}$$

(4.9) and (4.2) give

$$Pred_{k_i} \geq \delta\epsilon\Delta_{k_i}, \quad \text{for all large } i. \tag{4.10}$$

Theorem 2.2 shows that there exist an upper bound and a lower bound for R-function $R_\eta(t)$, thus by the criterion of updating the trust region radius in Algorithm SATR (formula (3.2)), we have

$$\Delta_{k_i} = O(||d_{k_i}||_2). \tag{4.11}$$

Then (4.8) and (4.11) imply

$$\Delta_{k_i} M_{k_i} \to 0, \quad i \to +\infty. \tag{4.12}$$

Thus we can assume that

$$\Delta_{k_i} < \Delta_{k_{i-1}} \tag{4.13}$$

holds for all i because $\{M_k\}$ is monotonely increasing. (4.9), (4.10) and the uniform continuity of $g(x)$ give

$$\begin{aligned} Ared_{k_i} &= f(x_{k_i}) - f(x_{k_i} + d_{k_i}) = -g_{k_i}^T d_{k_i} + o(||d_{k_i}||_2) \\ &= Pred_{k_i} + o(||d_{k_i}||_2). \end{aligned} \tag{4.14}$$

Hence

$$\lim_{i \to +\infty} \rho_{k_i} = \frac{Ared_{k_i}}{Pred_{k_i}} = 1. \tag{4.15}$$

That is, for large enough i, $\Delta_{k_i} \geq ||d_{k_{i-1}}||_2 = \Delta_{k_{i-1}}$, which contradicts (4.13). So (4.7) holds for $||d_k||_2 \leq \Delta_k$. **Q.E.D.**

Here we also have to cite another famous lemma from [2]:

Lemma 4.3. *Let $\{\Delta_k\}$, $\{M_k\}$ be two arbitrary positive sequences. If there exist constants $\tau > 0$, $\beta_1 > 0$, $0 < \beta_2 < 1$ and a subsequence I of $\{1, 2, 3, ...\}$ such that*

$$\Delta_{k+1} \leq \beta_1 \Delta_k, \qquad \forall k \in I; \tag{4.16}$$

$$\Delta_{k+1} \leq \beta_2 \Delta_k, \qquad \forall k \notin I; \tag{4.17}$$

$$\Delta_k \geq \frac{\tau}{M_k}, \qquad \forall k; \tag{4.18}$$

$$M_{k+1} \geq M_k, \qquad \forall k; \tag{4.19}$$

$$\sum_{k \in I} \frac{1}{M_k} < +\infty, \tag{4.20}$$

then

$$\sum_{k=1}^{+\infty} \frac{1}{M_k} < +\infty. \tag{4.21}$$

Now we are going to prove our main theorem:

Theorem 4.4. *The sequence $\{x_k\}$ generated by Algorithm 3.1 satisfies*

$$\liminf_{k \to +\infty} ||g_k||_2 = 0 \tag{4.22}$$

under Assumption 4.1.

Proof. If (4.22) were not true, $\{f(x_k)\}$ is bounded below and there exists $\epsilon > 0$, from Lemma 4.2, such that (4.6) and (4.7) hold. Define

$$I = \{k : \rho_k \geq c_2\}. \tag{4.23}$$

Assumption 4.1(iii) and Lemma 4.2 give

$$\begin{aligned} +\infty &> \sum_{k=1}^{+\infty}[f(x_k)-f(x_{k+1})] \\ &\geq \sum_{k\in I}[f(x_k)-f(x_{k+1})] \\ &\geq \sum_{k\in I} c_2 Pred_k \\ &\geq \sum_{k\in I}\delta c_2\epsilon\min\left\{\Delta_k,\frac{\epsilon}{M_k}\right\} \\ &\geq \sum_{k\in I}\delta c_2\epsilon\frac{\min\{\gamma,\epsilon\}}{M_k}. \end{aligned} \tag{4.24}$$

Thus $\sum_{k\in I}\frac{1}{M_k}<+\infty$. Applying Lemma 4.3 we have

$$\sum_{k=1}^{+\infty}\frac{1}{M_k}<+\infty, \tag{4.25}$$

which contradicts Assumption 4.1(iv). Hence the theorem is true. **Q.E.D.**

5. Numerical Results

We have implemented the new algorithm and compared it with the traditional trust region algorithm. In both the traditional trust region algorithm and the new algorithm SATR, the trial step is computed by the novel algorithm for solving the subproblem (1.2)-(1.3) proposed by Nocedal and Yuan in [5] (Algorithm 2.6), and B_k is updated by the BFGS formula. However we do not update B_k if the curvature condition

$$s_k^T y_k > 0 \tag{5.1}$$

fails, where

$$\begin{aligned} s_k &= x_{k+1}-x_k, && (5.2)\\ y_k &= g_{k+1}-g_k. && (5.3) \end{aligned}$$

For the new trust region algorithm SATR, we set

$$R_{c_2}(\rho_k)=\begin{cases} \frac{2}{\pi}(M-1-\gamma_2)\arctan(\rho_k-c_2)+(1+\gamma_2), & \text{if } \rho_k\geq c_2;\\ (1-\gamma_1-\beta)(e^{\rho_k-c_2}+\frac{\beta}{1-\gamma_1-\beta}), & \text{otherwise.}\end{cases} \tag{5.4}$$

It it easy to test that this $R_{c_2}(\rho_k)$ is an R-function. We implemented two versions, one, called Version 1, using parameters $\gamma_1=\gamma_2=0.01$, the other, called Version 2, using $\gamma_1=\gamma_2=0.15$, and both using $\beta=0.1$, $M=5$ and $c_2=0.25$. We choose the initial trust region radius $\Delta_1=1$ for both algorithms. The algorithms terminate if, at the k-th step, the norm of the gradient $||g_k||_2\leq 10^{-8}$. The global convergence results for the traditional trust region algorithm and Theorem 4.4 ensure both algorithms to end after a limited number of iterations.

We tested the algorithms on some of the problems given by Moré, Garbow and Hillstrom in [3]. These are small problems, with the number of variables ranging from 2 to 20. We have used the same numbering system as in [3]. The results are given in Table 1. We only list the number of iterations because this equals the number of function and gradient evaluations (i.e. there is exactly one function and gradient evaluation per iteration).

We observe from the table that Algorithm SATR performs better than the traditional trust region algorithm in most of the problems. To make further studies, we tested them on some of the problems with larger number of variables ranging from 20 to 50 in [3]. The results are listed in Table 2. Comparing the corresponding problems in Table 1 (with less variables) with those in Table 2 (with more variables), we may conclude that our new algorithm SATR is more efficient in solving large problems.

The algorithms were coded in MATLAB, and the tests were performed with MATLAB 5.0.

Acknowledgement. I would like to thank my supervisor, Professor Ya-xiang Yuan for his encouragement and help.

Table 1. Results on some of the problems of Moré, Garbow and Hillstrom

		Traditional Trust Region BFGS	SATR Version 1	SATR Version 2
	n	Iter	Iter	Iter
1	3	28	27	26
2	6	43	47	42
3	3	9	7	7
5	3	30	34	35
6	6	16	16	16
7	9	81	78	68
8	8	72	216	109
10	2	79	54	53
12	3	45	43	41
13	20	49	52	53
14	14	160	149	148
15	16	98	125	123
16	2	17	16	17
17	4	59	58	55
18	8	43	34	35

Table 2. Results on some medium-size problems

		Traditional Trust Region BFGS	SATR Version 1	SATR Version 2
	n	Iter	Iter	Iter
6	20	28	28	28
	40	35	37	37
	50	36	36	36
8	20	204	148	165
	40	157	219	112
	50	255	298	139
13	20	49	52	53
	40	57	57	57
	50	54	56	57
14	20	197	169	192
	40	313	292	275
	50	388	326	318
15	20	94	118	89
	40	165	159	138
18	20	102	102	101

References

[1] Y. Yuan, *On the convergence of trust region algorithms* (in Chinese), Mathematica Numerica Sinica, August 1994, 333-346.

[2] Y. Yuan and W.Sun, *Theories and Methods for Optimization* (in Chinese), Science Press, Beijing, 1997, 563-565.

[3] J. J. Moré, B. S. Garbow and K. E. Hillstrom, *Testing unconstrained optimization software*, ACM transactions on Mathematical Software 7, 1981, 17-41.

[4] M. J. D. Powell, *Convergence properties of a class of minimization algorithms*, in: O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds., Nonlinear Programming 2, Academic press, New York, 1975, 1-27.

[5] J. Nocedal and Y. Yuan, *Combining trust region and line search techniques*, in: Y. Yuan, ed., Advances in Nonlinear Programming (Kluwer), 1998, 153–175.

[6] R. Fletcher, *A model algorithm for composite NDO problem*, *Math. Prog. Study*, **17**(1982), 67-76.

[7] R. Fletcher, *Practical Methods of Optimization* (second edition), John Wiley and Sons, Chichester, 1987.

[8] J. J. Moré, *Recent developments in algorithms and software for trust region methods*, in: A. Bachem, M. Grätschel and B. Korte, eds., Math. Prog.: The State of the Art, Springer-Verlag, Berlin, 1983, 258-287.

[9] M. J. D. Powell, *On the global convergence of trust region algorithms for unconstrained optimization*, *Math. Prog.*, **29**(1984), 297-303.