Numerical Optimization
Week 4 Assignment

CHM564

1 Introduction

This week we test Trust-region algorithm on benchmark functions. We will discrible the performance of algorithm and analyze the parameters. Then, we will discuss the difference between the Line-search and Trust-region.

2 Implement Algorithm

2.1 Description of Trust-region

Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function. We will assume that the model function m_k that is used at each iterate x_k is quadratic. Moreover, m_k is based on the Taylor-series expansion of f.
To obtain each step, we seek a solution of the subproblem

$$\min_{p \in R^n} \; m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \qquad s.t. \; \|p\| \leq \Delta_k \tag{1}$$

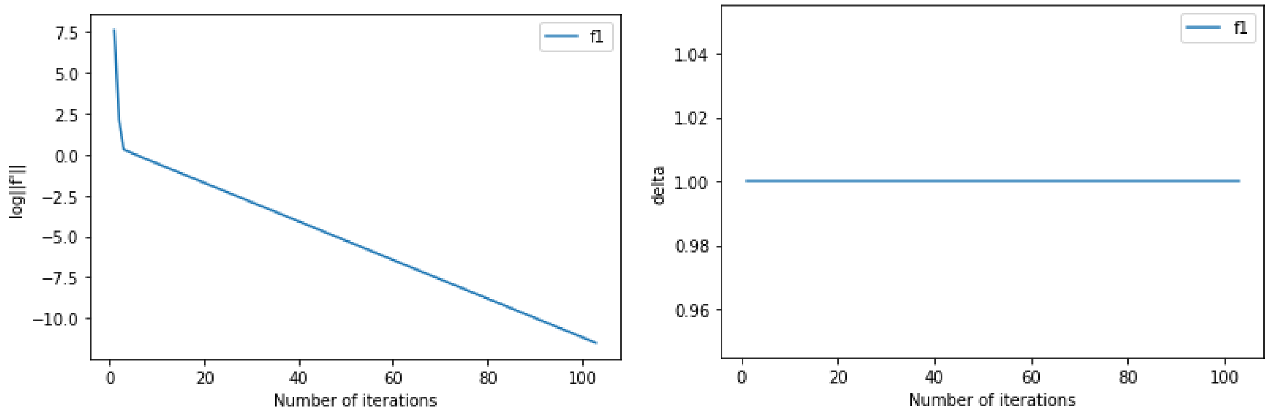
Where Δ_k is the trust-region radius. In this report, we define $\| \cdot \|$ to be the Euclidean norm, so that the solution p_k^* is the minimizer of m_k in the ball of radius Δ_k.
p should satisfy **Theorem 4.1:**
$The\ vector\ p^*\ is\ a\ global\ solution\ of\ (1).\ If\ and\ only\ if\ p^*\ is\ feasible\ and\ there\ i\ a\ scalar\ \lambda \geq 0\ such\ that\ the\ following\ conditions\ are\ satisfied:$

$$(B + \lambda I)p^* = -g$$
$$\lambda(\Delta - \|p^*\| = 0)$$
$$(B + \lambda I) is\ positive\ semidefinite$$

Another key ingredient is the strategy for choosing the trust-region radius Δ_k at each iteration. We base this choice on the agreement between the model function m_k and the objective function f at previous iterations. Given a step p_k we define the ratio

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}$$

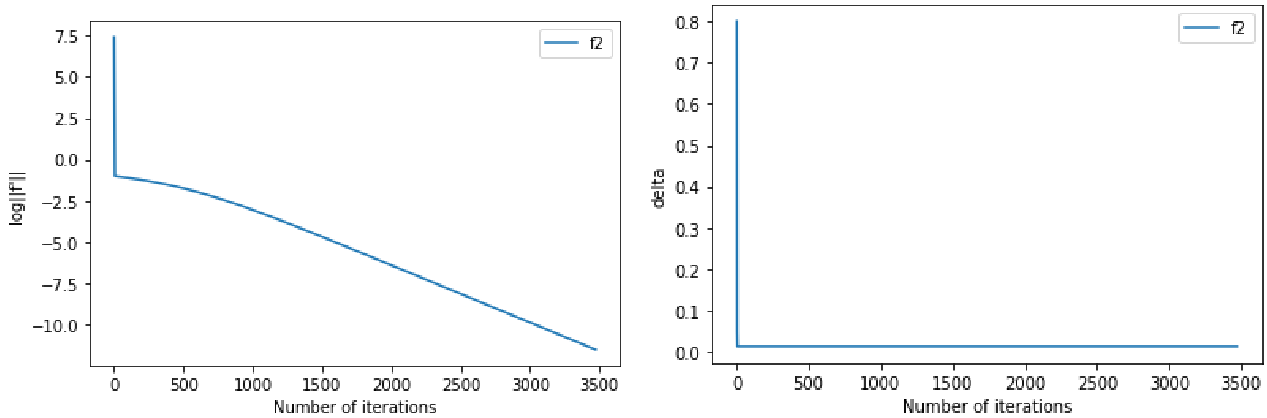
2.2 Implement Algorithm



(a) initial guess(1,1)

(b) initial $\Delta = 1$

Figure 1: Test on Function1



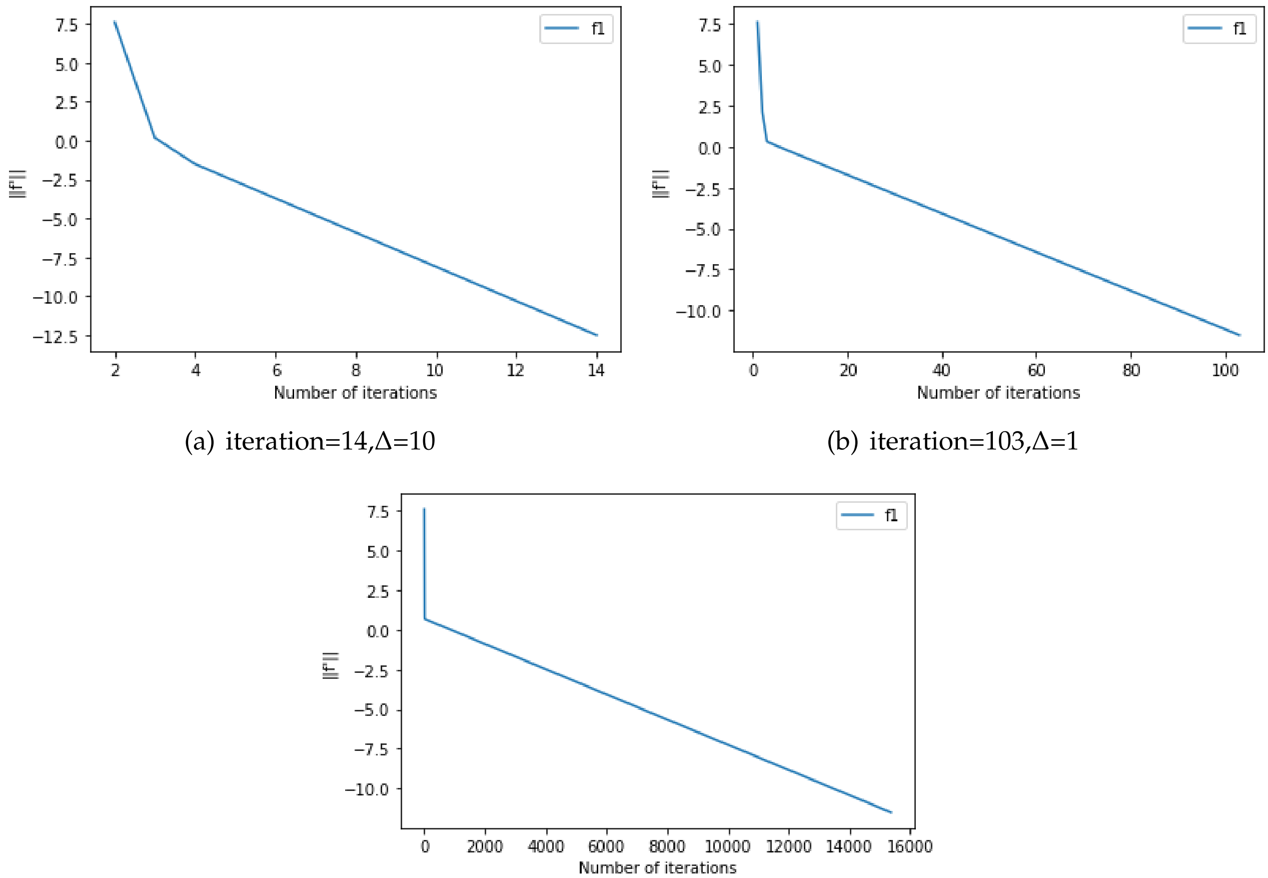
(a) initial guess(2,2)

(b) initial $\Delta = 0.8$

Figure 2: Test on Function2

We can see that, the algorithm shows well on f_1, f_2. But in fact, the conditions of program on these functuons are not same.
It's obviously that if ρ_k is close to 1, there is good agreement between the model m_k and the function f over this step, so it is safe to expand the trust region for the next iteration. If ρ_k is positive but significantly smaller than 1, we do not alter the trust region, but if it is close to zero or negative, we shrink the trust region by reducing Δ_k at the next iteration.
The condition to update x_k on f_1 is **if** $\rho_k < \frac{1}{4}$. However, its not suitable for f_2. It can't satisfy this condition, so x_k can't be updated, then it will lose in circle.

2.3 Analysis of Parameters



(a) iteration=14,Δ=10

(b) iteration=103,Δ=1

(c) iteration=15364,Δ=0.5

Figure 3: Test on Function1

From Figure 3, we can see that, for f_1, it is sensitively about parameters and always can converage. And the number of iteration become larger with smaller Δ. Its easy to understand, with bigger region, x_k can fall further.
Also, I change the ρ on f_1. The results are same. It shows f_1 may not sensitive on ρ. But it doesn't means ρ is unimportant. For f_2, if ρ smaller, the program will lose in circle.

3 Theoretical part

3.1 Desciption of p

From Theorem 4.1, we know that p_k must satisfy these three conditions. We can divide the process in two situations.
1) When $\lambda = 0$, p^* should satisfy:

$$Bp^* = -g$$
$$\|p^*\| \neq \Delta$$
$$B \ is \ positive \ semidefinite$$

2) When $\lambda \neq 0$,

$$(B + \lambda I)p^* = -g$$
$$\|p^*\| = \Delta$$
$$(B + \lambda I) is\ positive\ semidefinite$$

So, our aim is to find λ which can make $\|p(\lambda)\| = \| - (B + \lambda I)^{-1} g\| = \Delta$

B is positive definite, so $B = Q\Lambda Q^T$ and $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ and $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$, then we can get:

$$p(\lambda)^2 = -\sum_i^n \frac{(q_j^T g)^2}{(\lambda_j + \lambda)^2}$$

Now, we want to calculate λ

1)When $q_i^T g \neq 0$
There are many methods to find λ, in this report, we used *Newton*.

For $B + \lambda^{(l)} I = R^T R$ by Cholesky

$$R^T R p_l = -g$$

if $\|p_l\| \leq \Delta$, stop. else $R^T q_l = p_l$. We can calculate λ

$$\lambda^{(l+1)} = \lambda^{(l)} + (\frac{\|p_l\|}{\|q_l\|})^2 (\frac{\|p_l\| - \Delta}{\Delta})$$

Also, we can use *bijection* to find λ

Firstly, we should find initial interval[λ_0,λ_1], which satisfy$\|p(\lambda_0)\| > \Delta$ and $\|p(\lambda_1)\| < \Delta$.
Then, we choose the middle $\lambda' = \frac{\lambda_0 + \lambda_1}{2}$, repeat it.
The result is nearly as same as *Newton*, but it took more time.

2)When $q_i^T g = 0$
We always called this situation as *hardcase*. To solve this problem, we add a scalar τ.
For any scalar τ, we have

$$\|p\|^2 = \sum_{j:\lambda_j \neq \lambda_i} \frac{(q_j^T g)^2}{(\lambda_j + \lambda)^2} + \tau^2$$

3.2 Trust-region and Line-search

The line search method and the trust region method are both important methods to ensure the overall convergence in the optimization algorithm.

Their common purpose is to find the displacement of each iteration step in the optimization algorithm. So that it can update a new iteration point.
The difference between these algorithms is how to determine the update amount of the optimization variable in each iteration. The line-search method determines the direction

first and then decide the step size, but the trust-region method determines the maximum step size, and then the direction and actual step size. Compared with trust-region method, line-search may be easier and simple. However, trust-region is much more accuary than line-search. For line-search, if one step gets mistake, it will be difficult to correct. But turst-region can modify in its region , even stop updating the points until the next suitable step.

4 Summary

In this report, we discuss the trust-region method and test it on f_1, f_2. It always shows well. Also we analyze the parameters, we can see that this algorithm is sensitive about it on different functions. And then we describe the different situations on how to find p_k. Finally, we discuss the line-search method and trust-region method respectively. We get conclusion, line-search is easier than trust-region, but trust-region is much more accuary.