Numerical Optimization
Week 2 Assignment

CHM564

Contents

1 Introduction

In this report, we will test three algorithms,*BFGS*, *newton* and *dogleg* with some functions. Good algorithms should possess Robustness,Efficiency and Accuracy, so we will test these properties to show which algorithm preforms better.

2 Three properties

2.1 Theoretical basis

To test these algortithms, we'd better to use convex functions. Now, we proof function3 is convex function(d=1).
Proof :

$$
\begin{aligned}
f_3(x) = \log(\epsilon + x^2) \Rightarrow f_3''(x) &= \frac{2\epsilon - 2x^2}{(\epsilon + x^2)^2} > 0 \\
&\Rightarrow 2\epsilon - 2x^2 > 0 \\
&\Rightarrow |x| < \sqrt{\epsilon}
\end{aligned}
$$

If $|x| \geq \sqrt{\epsilon}$, the Hessian will be negetive or zero. Then we can't make sure the direction of *Newton* is decent. So, we will use $|x| < 10^{-8}$

For function4, we will show its Lipschitz and for large x the Hessian is close to 0.
proof :First, we will prove the gradient of f_4 can be bounded by a constant.

$$\bigtriangledown f_4(x) = \begin{bmatrix} \frac{\exp(qx_1)-100}{\exp(qx_1)+1} \\ \frac{\exp(qx_2)-100}{\exp(qx_2)+1} \\ \dots \\ \frac{\exp(qx_d)-100}{\exp(qx_d)+1} \end{bmatrix} = \begin{bmatrix} 1 - \frac{101}{1+\exp(qx_1)} \\ 1 - \frac{101}{1+\exp(qx_2)} \\ \dots \\ 1 - \frac{101}{1+\exp(qx_d)} \end{bmatrix}$$

We, can see that, each elements is smaller than -100, so

$$|| \bigtriangledown f_4(x)||_2 \leq 100\sqrt{d}$$

Take $100\sqrt{d}$ to be Lipschitz constant.Now, we prove f_4 is globally Lipschitz. Then, we use Hessian of f_4:

$$\begin{aligned} \lim_{x\to\infty} \bigtriangledown^2 f_4(x) &= \lim_{x\to\infty} \frac{(101q\exp(qx))'}{(1+\exp(qx)^2)'} \\ &= \lim_{x\to\infty} \frac{101q^2\exp(qx)}{2+2\exp(qx)q\exp(qx)} \\ &= \lim_{x\to\infty} \frac{101q}{2+2\exp(qx)} \\ &= 0 \end{aligned}$$

2.2 Accuracy

We know the optimum of function1, x-value is $(0,0)$. Also, we can easily get its also the optimum value of function3. And from Assignment1, we know for function4, the optimum is about $(4.608 * 10^{-8}, 4.608 * 10^{-8})$.

From these tables, we can get *Newton* has the best accuary, but it needs a closed initial point. We can see that, if we choose $(100,100)$ as initial point, the results is worst than *BFGS*. Although *BFGS* is also affected, the result is still acceptable. However, the *dogleg* is the suitable algorithm for a far point. So, for the firt function, if we can get a closed initial point, the *Newton* is the best, conversely, is *Dogleg*. The *BFGS* is the most stable.
For function3, we can see that the *BFGS* shows the best. Cause the others need Hessian to iterate, and the element of Hessian is too small and easy to have error.

Table 1: Algorithms test on function1

Algorithm		$(x_1, x_2)^a = (1, 1)$	$(x_1, x_2) = (100, 100)$
BFGS	$x*^b$	$(1.08 * 10^{-19}, 0)$	$(-4.33 * 10^{-19}, -4.33 * 10^{-19})$
	$y*^b$	0	0
Newton	$x*$	$(0, 0)$	$(1.42 * 10^{-14}, 1.39 * 10^{-17})$
	$y*$	0	0
Dogleg	$x*$	$(-4.33 * 10^{-19}, -4.33 * 10^{-19})$	$(0, 0)$
	$y*$	0	0

[a]This is input value x.[b]This is the output value.

Table 2: Algorithms test on function3

Algorithm		$(x_1, x_2) = (10^{-20}, 10^{-20})$	$(x_1, x_2) = (10^{-9}, 10^{-9})$
BFGS	$x*$	$(9.99 * 10^{-21}, 1.27 * 10^{-25})$	$(-1.98 * 10^{-25}, -8.05 * 10^{-30})$
	$y*$	-36.841361	-36.841361
Newton	$x*$	$(10^{-21}, -9.99 * 10^{-27})$	$(9.98 * 10^{-11}, -2.22 * 10^{-11})$
	$y*$	-36.841361	-36.83634
Dogleg	$x*$	$-$	$-$
	$y*$	$-$	$-$

Table 3: Algorithms test on function4

Algorithm		$(x_1, x_2) = (-10^{-7}, -10^{-7})$	$(x_1, x_2) = (10^{-7}, 10^{-7})$
BFGS	$x*$	$-$	$-$
	$y*$	$-$	$-$
Newton	$x*$	$(6.26 * 10^{-7}, 6.26 * 10^{-7})$	$(6.26 * 10^{-8}, 6.26 * 10^{-8})$
	$y*$	0.000001	0
Dogleg	$x*$	$(4.61 * 10^{-8}, 4.61 * 10^{-8})$	$(4.61 * 10^{-8}, 4.61 * 10^{-8})$
	$y*$	0	0

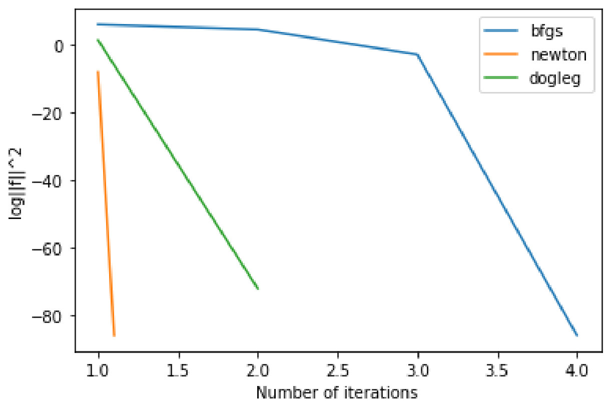
2.3 Robustness

To test Robustness, we need to see if the algorithm is suitable for different situations. From above tables, we can get *Newton* is better than *Dogleg* and *BFGS*. For function3 *Dogleg* can't iterate successfully and get a worse result. For function4 *BFGS* also unsuccessful due to precision loss.

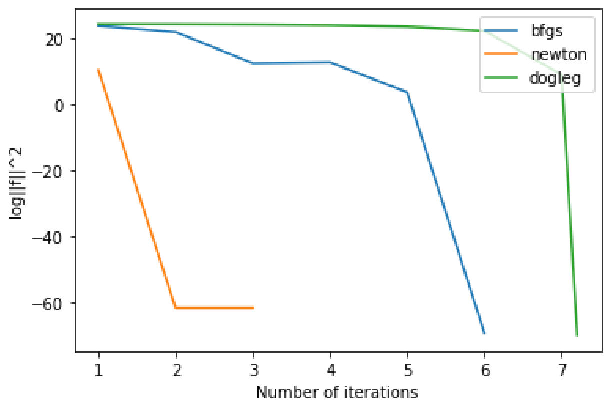
Newton and *BFGS* are Line Search method and *Dogleg* is Trust-Region Method. *Newton* has a fast convergence speed and small number of iterations. However initial value will have a big impact on the result, if the function is non-convex, its easily fall in saddle points. *BFGS* approximate inverse of Hessian matrix with positive definite matrix, so the speed of a single iteration will faster than *Newton*. But it means the results will not as accurate as *Newton*.

2.4 Efficiency

From figures, we can see that, *Newton*'s converage rate always faster than others, its super linear. But if we use a far point as initial point, the *dogleg* will become slower than *BFGS*, its because model function may not be a good approximation of f when x is far from the optimum. If we choose initial points near the optimum, *dogleg* also can have a fast speed, but still slower than *Newton*. *BFGS* may have a good speed of each iteration, but compared with *Newton*, it needs iterate more times. Overall, its worse.
Figure3 shows function4's different coveragerate. Gradient of x smaller than 0 is totally different with x bigger than 0. It can see directly that choos inital points in different intervals will also have different convarage rate even the distance to the best value is the same.

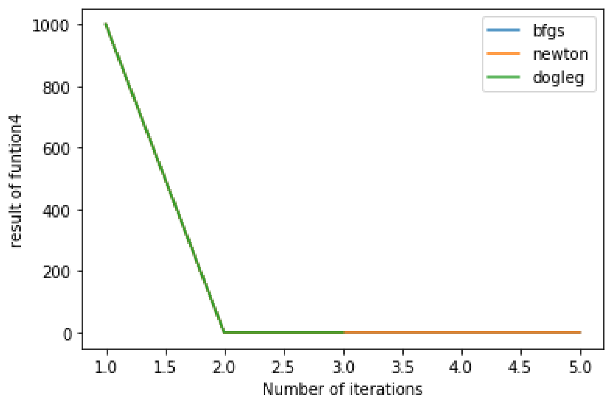


(a) Input x(1,1)

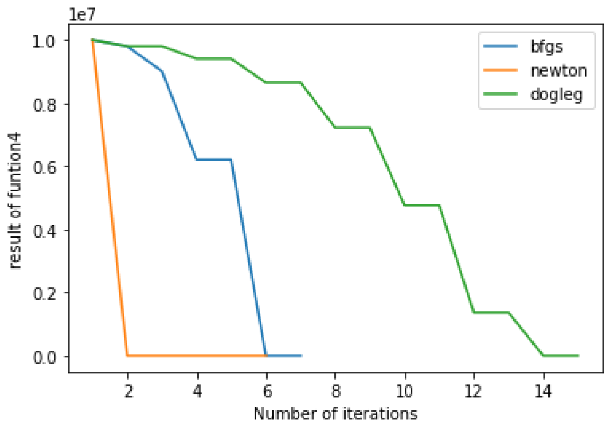


(b) Input x(100,100)

Figure 1: Converage rate of Function1

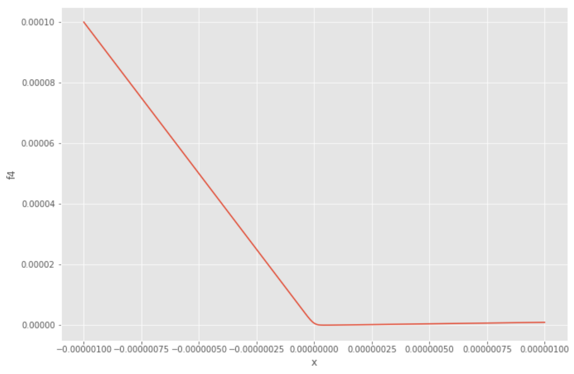


(a) Input x(1,1)

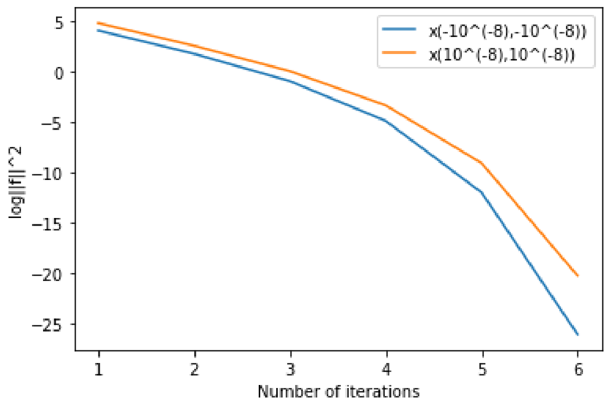


(b) Input x(100,100)

Figure 2: Function1's result of each iteration



(a) Function4 with d=1



(b) Converage rate of Dogleg

Figure 3: Plot for function4

3 Summary

This report disscused the pros and cons of *BFGS*, *newton* and *dogleg*, and test their properties respectively.
Newton seems is the best, but its depend too much on inital point. *BFGS* doesn't depend on inital point and needn't use Hessian, it is fast in each iteration, but overall its still a little slow. In addition, the accuracy and robustness are worse than *Newton*. In a convex function, if we can't make sure the approximate of optimum point, it will be better to use *BFGS* first.*Dogleg* is the only Trust-Region Method in these algorithms. Most of situations, it always performed well. But speed of converage may a little slow.