Optimization Method Coding Project 1: Newton Methods

Li, Ziyao [1]

Abstract

This work focuses on the realization of Newton and Quasi-Newton algorithms. Two test functions are evaluated in this work, namely Box Three-Dimensional Function and Penalty II Function. A Minimal Surface Problem is also solved using Quasi-Newton method. Numerical Experiments are conducted on three formerly mentioned problems. Generally Newton methods takes less iterations than Quasi-Newton methods. SR1 performs poorly on all three problems, making it the least appealing method among all tested algorithms.

1. Line Search

Two types of line search methods, exact line search (ELS) and inexact line search (ILS) are realized. Here, the exact line search method is mainly discussed. The 0.618 exact search method is realized for conducting ELS. During the process of realizing the algorithm, several problems emerged.

One is that the prior knowledge of α_0 is really hard to select. No matter how small the α_0 is selected to be, when the gradient g_k of the current point x_k is extremely large or the determinant of the Hesse matrix G_k have relatively small eigenvalues, the function value at $x_{k+1} = x_k + \alpha_0 d_k$ can be very large and may overflow. To solve this problem, a detector is added to solve overflow problem. When an overflow is detected, we simply cut the original α_0 into α_0/t with given constant t, and then solve the problem.

The second problem is the multi-modal problem on α. When the searched α leads to $f(x_k + \alpha d) > f_k$, then the step length is not correctly found, and the function must be multi-modal. In this situation, adopting the same strategy, that is, cut the original α_0 into α_0/t, we can search for a valley region closer to 0, making it more likely to be the valley which $\alpha = 0$ is in. Note that such a valley always exists if d_k is the decreasing direction. Continuously doing so until $\alpha < \epsilon$ indicates that d_k is not a decreasing direction, and the algorithm fails. Figure 1 shows a real situation when the second advancement works.

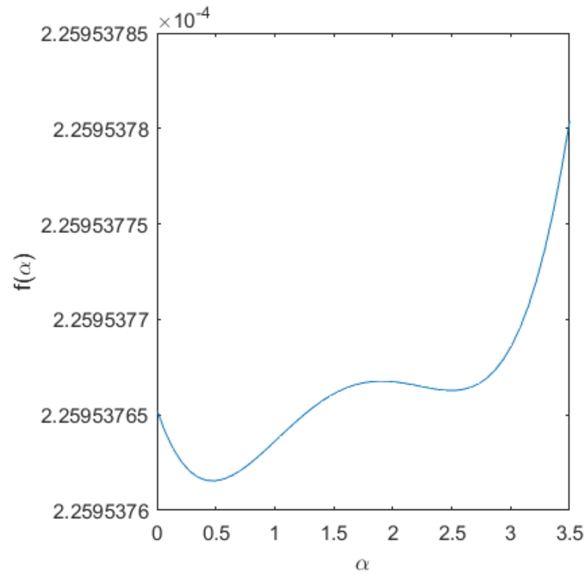

Figure 1. Graph of $\tilde{f}(\alpha) = f(x + \alpha d)$ when a multi-modal situation occurs in minimizing Penalty II Function.

2. Numerical Performance

Performance of five different methods are tested. Relevant results are shown in Table 1, 2, 3 and 5. Note that the number in the bracket indicates the relative stop criterion. For example, $8(4)$ indicates the algorithm converges after 8 iterations (times of function evaluations) under the criterion

$$\begin{aligned} |f_k - f_{k-1}| &\leq \epsilon \\ ||g_k||_\infty &\leq \epsilon \\ \epsilon &= 10^{-4} \end{aligned}$$

with default $\epsilon = 10^{-8}$.

In the Box 3D problem, although in the first iteration the Hesse matrix is not positive definite and no measures are taken, Damped Newton method gives the best performance.

[1] SID: 1500017776, Yuanpei College, Peking University.

Table 1. Numerical performance (iterations) of Box 3D Function Minimization Problem.

M	3	5	10	15	20
DAMP	8	7	7	9	8
LM	97	8(4)	1439	953	760
SR1-B	19	3(1)	3(0)	3(0)	11
SR1-H	19	3(1)	3(0)	3(0)	11
DFG-B	19	9	13	14	11
DFG-H	19	9	13	14	11
BFGS-B	19	9	13	14	11
BFGS-H	19	9	13	14	11

Table 2. Numerical performance (feva) of Box 3D Function Minimization Problem.

M	3	5	10	15	20
DAMP	411	359	357	455	408
LM	5429	408(4)	69827	48613	39526
SR1-B	924	151(1)	163(0)	185(0)	545
SR1-H	924	151(1)	163(0)	185(0)	545
DFG-B	1022	480	672	801	580
DFG-H	1022	480	672	801	580
BFGS-B	944	451	632	705	556
BFGS-H	944	451	632	705	556

When alternating the Hesse matrix into a positive definite one using LM algorithm, the performance suffers. SR1 method also performs very poorly in this problem. The algorithm fails after several iterations meeting a problem that the direction calculated with the method cease to be a decreasing direction. DFG and BFGS methods give similar performances. By the way, it makes no difference in numerical performance whether a B_k matrix or a H_k one is adopted in the method in this problem.

In the Penalty II problem, Damped Newton method cannot perform after a few iterations due to non-decreasing directions, while a simple LM adjustment gives the best performance among all tested methods. SR1 meets the same problem emerges in Box 3D. Performance of DFG and BFGS are poor in terms of converging speed, while a slightly lowered criterion leads to great improvements. Whether an H_k method or a B_k one is starting to make some differences, however, the differences are slight and the performances are far from stable in two methods to compare which one is better.

3. Minimal Surface Problem

Minimal Surface Problem is the Problem 4.3 in *P153*. The problem calculates the minimal surface of a function v on a convex set $D \in R^2$ (or higher dimension) under boundary constraints. In this problem, a differential form of the problem is considered, and the convex set D is limited to

Table 3. Numerical performance (iterations) of Penalty II Function Minimization Problem.

N	2	4	6	8	10
DAMP	-	4(4)	3(5)	103	4(3)
LM	5	221	186	104	74
SR1-B	1(0)	1(-1)	7(5)	2(1)	2(0)
SR1-H	1(0)	1(-1)	7(5)	2(1)	2(0)
DFG-B	8	350	1112	408	763
DFG-H	8	544	1413	647	637
BFGS-B	8	373	1123	862	745
BFGS-H	8	391	1355	617	785

Table 4. Numerical performance (feva) of Penalty II Function Minimization Problem.

N	2	4	6	8	10
DAMP	-	212(4)	161(5)	5056	240(3)
LM	253	9838	8676	5113	3827
SR1-B	49(0)	49(-1)	339(5)	97(1)	97(0)
SR1-H	49(0)	49(-1)	339(5)	97(1)	97(0)
DFG-B	377	25601	80624	29354	53704
DFG-H	377	39163	101924	46186	44685
BFGS-B	385	17723	53124	40780	35421
BFGS-H	385	18596	64215	29362	37254

a square (rectangular) region for the convenience of grid-differentiation.

Figure 3 and 3 shows an instance of the process. Figure 3 shows the initial value of v and Figure 3 shows the final output. It proves the algorithm to function since the plane in Figurefig3 is obvious the minimal solution.

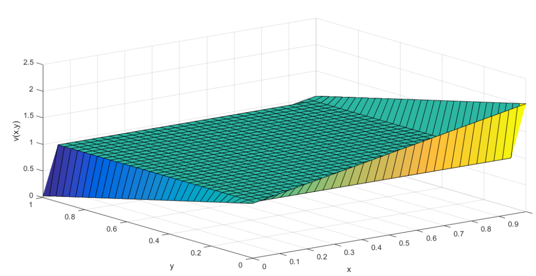

Figure 2. Graph of linear boundary constraints and initial points with $v(x, y) = 1$.

A comparison of numerical performance of 5 pre-defined types are compared using different algorithms. Detailed definition of the boundary constraints can be found in the relevant codes with coded types. The grid is defined to be 32×32, i.e. the problem is a 961-dimensional optimization problem. The stop criterion is set to be 10^{-6}.

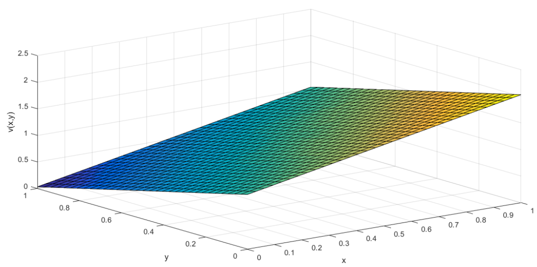

Figure 3. Graph of linear boundary constraints and calculated minimal surface.

Table 5. Numerical performance (function evaluation times) of Penalty II Function Minimization Problem.

TYPE	0	1	2	3	4
DAMP(ITER)	5	5	6	5	5
DFG(ITER)	102	85	89	97	69
BFGS(ITER)	101	84	88	96	70
DAMP(FEVA)	5971	5985	6985	5983	5985
DFG(FEVA)	5694	4554	4759	5250	3504
BFGS(FEVA)	4741	4078	4289	4681	3423

The results shows that Newton methods uses a lot less iterations to reach convergence. However, since a differential method is used to approximate the Hesse matrix at a given point, the times of evaluating f is at the same level as Quasi-Newton methods. SR1 still doesn't perform well on this problem, failing to figure out decreasing directions after approximately 300 iterations, when the situation is far from achieving stop criterion. DFG and BFGS also performs similarly.