ACM 113 Introduction to Optimization - Problem Set 3

Ling Li, ling@cs.caltech.edu

May 8, 2001

3.1 Assume that $\{p_i\}_{i=1}^n$ are not linearly independent. Without loss of generality, there exit constants u_i such that

$$p_1 = \sum_{i=2}^{n} \mu_i p_i. \tag{1}$$

For $i = 2, 3, \ldots, n$, the conjugate condition gives

$$0 = p_i^T A p_1 = \mu_i p_i^T A p_i.$$

Since A is positive definite, $\mu_i = 0$. From (1), $p_1 = \mathbf{0}$, contradicting with $p_1 \neq \mathbf{0}$. So $\{p_i\}_{i=1}^n$ are linearly independent.

3.2 For the inner linear CG iteration, we have $A = \nabla^2 f_k$, $b = -\nabla f_k$, and $x^{(0)} = 0$.

(a) Thus $r^{(0)} = Ax^{(0)} - b = -b$, and

$$p^{(0)} = -r^{(0)} = b = -\nabla f_k. \tag{2}$$

So $x^{(1)} = x^{(0)} + \alpha^{(0)} p^{(0)} = -\alpha^{(0)} \nabla f_k$. When negative curvature is detected on the first step, $x^{(1)}$ is returned as p_k. Since we now set $\alpha^{(0)} = 1$, p_k is just the steepest descent direction.

(b) During the inner linear CG iteration,

$$\beta^{(i)} = \frac{{r^{(i)}}^T r^{(i)}}{{r^{(i-1)}}^T r^{(i-1)}}$$

is always positive. Properties of the linear CG ensure that for $i > j$,

$${r^{(i)}}^T p^{(j)} = {r^{(i)}}^T r^{(j)} = 0.$$

Together with (2), we have for $i > 0$,

$$b^T p^{(i)} = b^T \left(-r^{(i)} + \beta^{(i)} p^{(i-1)}\right) = {r^{(i)}}^T r^{(0)} + \beta^{(i)} b^T p^{(i-1)} = \beta^{(i)} b^T p^{(i-1)},$$

i.e., $b^T p^{(i)}$ and $b^T p^{(i-1)}$ have the same sign. Since $b^T p^{(0)} = b^T b > 0$, we know that $b^T p^{(i)} > 0$ for all i. We also have

$$\alpha^{(i)} = \frac{{r^{(i)}}^T r^{(i)}}{{p^{(i)}}^T A p^{(i)}} > 0,$$

as long as $p^{(i)^T} A p^{(i)} > 0$. Hence

$$b^T x^{(i+1)} = b^T \left(x^{(i)} + \alpha^{(i)} p^{(i)} \right) = b^T x^{(i)} + \alpha^{(i)} b^T p^{(i)} > b^T x^{(i)}.$$

Since $b^T x^{(0)} = 0$, we get $b^T x^{(i+1)} > 0$ for $i \geq 0$ as long as the negative curvature is met. Thus the truncations in the linear CG ensure that $-\nabla f_k^T p_k > 0$.

3.3 Hessian-free Newton methods.

(a) By the Taylor's series,

$$f(x_k + \epsilon) = f(x_k) + \epsilon f'(x_k) + \frac{\epsilon^2}{2} f''(x_k + t\epsilon), \quad t \in (0,1),$$

we have

$$f'(x_k) = \frac{f(x_k + \epsilon) - f(x_k)}{\epsilon} - \frac{\epsilon}{2} f''(x_k + t\epsilon).$$

If we use

$$\frac{f(x_k + \epsilon) - f(x_k)}{\epsilon} \tag{3}$$

to approximate $f'(x_k)$, we get a truncation error

$$T = \frac{\epsilon}{2} \left| f''(x_k + t\epsilon) \right| \leq \frac{L}{2} \epsilon.$$

Then if we want to T be small, we should use small ϵ. However, when calculating (3), we also have the roundoff error R. From

$$\begin{aligned} \text{float}\left(\text{float}(x) - \text{float}(y)\right) &= \left(\text{float}(x) - \text{float}(y)\right)(1 + \epsilon_{xy}) \\ &= (x + \epsilon_x - y - \epsilon_y)(1 + \epsilon_{xy}) \\ &= (x - y) + (x - y)\,\epsilon_{xy} + (\epsilon_x - \epsilon_y)(1 + \epsilon_{xy}), \end{aligned}$$

where $|\epsilon_x| \leq \epsilon_u |x|$, $|\epsilon_y| \leq \epsilon_u |y|$, and $|\epsilon_{xy}| \leq \epsilon_u$. With $x \stackrel{\text{def}}{=} f(x_k + \epsilon)$ and $y \stackrel{\text{def}}{=} f(x_k)$, (assume ϵ can be precisely represented by the machine, for example, $\epsilon = 2^{-n}$)

$$\begin{aligned} R &= \left| \frac{\text{float}\left(\text{float}(f(x_k + \epsilon)) - \text{float}(f(x_k))\right)}{\epsilon} - \frac{f(x_k + \epsilon) - f(x_k)}{\epsilon} \right| \\ &= \left| \frac{f(x_k + \epsilon) - f(x_k)}{\epsilon} \epsilon_{xy} + \frac{\epsilon_x - \epsilon_y}{\epsilon}(1 + \epsilon_{xy}) \right| \\ &\leq \left| f'(x_k + \xi\epsilon) \right| \epsilon_u + \frac{\epsilon_u \left(|f(x_k + \epsilon)| + |f(x_k)| \right)}{\epsilon}(1 + \epsilon_u) \\ &\leq \frac{2\epsilon_u L}{\epsilon} \left(1 + \epsilon_u + \frac{\epsilon}{2} \right), \end{aligned}$$

where $\xi \in (0,1)$. Thus the leading order roundoff error is

$$R \approx \frac{2\epsilon_u L}{\epsilon}. \tag{4}$$

The total error

$$E = T + R \approx \frac{L}{2}\epsilon + \frac{2\epsilon_u L}{\epsilon}. \tag{5}$$

Thus to minimize the total error, we'd better use $\epsilon = 2\sqrt{\epsilon_u}$, and the total error is $E \approx 2\sqrt{\epsilon_u}L \approx 10^{-8}L$. (Assume $\epsilon_u \approx 10^{-16}$.) When ϵ decreases from a large value, the error E first decreases, and after ϵ passes the optimal value, E increases.

(b) Note that $f(x)$ is a real-value function. With Taylor's series,*

$$f(x_k + i\epsilon) = f(x_k) + if'(x_k)\epsilon - \frac{f''(x_k)}{2}\epsilon^2 - \frac{if'''(x_k + i\xi\epsilon)}{3!}\epsilon^3, \quad \xi \in (0,1). \tag{6}$$

Thus

$$\mathrm{Im}\left[f(x_k + i\epsilon)\right] = f'(x_k)\epsilon - \frac{\mathrm{Re}\left[f'''(x_k + i\xi\epsilon)\right]}{3!}\epsilon^3,$$

or

$$f'(x_k) \approx \frac{\mathrm{Im}\left[f(x_k + i\epsilon)\right]}{\epsilon}$$

with truncation error $T = \frac{\epsilon^2}{3!}\left|\mathrm{Re}\left[f'''(x_k + i\xi\epsilon)\right]\right| \le \frac{\epsilon^2 L}{6}$. Now the roundoff error is

$$R \le \frac{\epsilon_u \left|\mathrm{Im}\left[f(x_k + i\epsilon)\right]\right|}{\epsilon} \approx \epsilon_u \left|f'(x_k)\right| \le \epsilon_u L.$$

(We omit the computation errors within the calculation of $\mathrm{Im}\left[f(x_k + i\epsilon)\right]$.) The total error is

$$E = T + R \approx \frac{\epsilon^2 L}{6} + \epsilon_u L. \tag{7}$$

Thus E decreases with ϵ decreases. For smaller error, we should use smaller ϵ, and then $E \approx \epsilon_u L \approx 10^{-16}L$.

(c) For $f(x) = x^{9/2}$, the actual derivative is $f'(x) = \frac{9}{2}x^{7/2}$. Thus the real error can be obtained by measuring the difference between the estimated one and the real one. Figure 1 shows the error v.s. ϵ at $x = 1.5$. The slopes of those thick red lines imply that the real behavior of the error is just controlled by (5) or (7).

(d) We can generalize (3) as

$$\nabla^2 f(x)p \approx \frac{\nabla f(x + \epsilon p) - \nabla f(x)}{\epsilon},$$

since we have $\nabla f(x + \epsilon p) \approx \nabla f(x) + \epsilon \nabla^2 f(x)p$. Similarly to (a), if each component of the 1^{st}, 2^{nd} and 3^{rd} derivatives of f is bounded by L, the truncation error for each component of $\nabla^2 f(x)p$ is

$$T \le \frac{\epsilon}{2}L\left(\sum_i |p_i|\right)^2 \le \frac{\epsilon n}{2}L\,\|p\|^2,$$

*Define $g(x) = f(x_k + ix)$, $x \in \mathbb{R}$. Then $g^{(n)}(x) = i^n f^{(n)}(x_k + ix)$. Thus from

$$g(\epsilon) = g(0) + g'(0)\epsilon + \frac{g''(0)}{2}\epsilon^2 + \frac{g'''(\xi\epsilon)}{3!}\epsilon^3, \quad \xi \in (0,1),$$

we have (6).

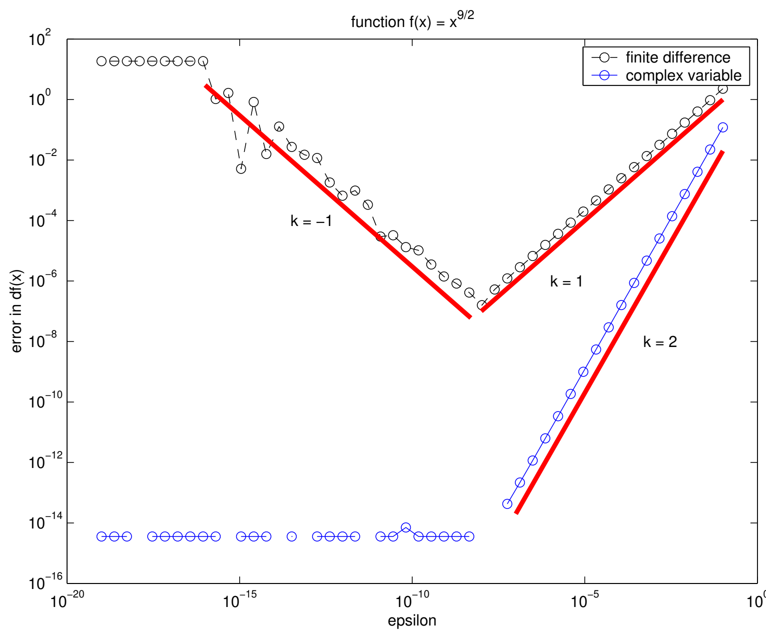


Figure 1: The total error E when calculating the first derivative of $f(x) = x^{9/2}$ at $x = 1.5$. Two methods, the finite difference method and the complex variable method are used. The slopes of those thick red lines imply how the error changes with respect to ϵ. For example, the right-most red line implies that $E \propto \epsilon^2$, which coincides with (7) when $\epsilon^2 \gg \epsilon_u$. (Note: some points are 'lost' because they are machine zero and can not be shown in a `loglog` plot.)

where n is the dimension of p. The roundoff error, which I do not want to go to the details, is

$$R \approx \frac{2\epsilon_u L}{\epsilon}\left(1 + \epsilon_u + \frac{\epsilon\sqrt{n}}{2}\|p\|\right)$$

Usually $\epsilon\sqrt{n}\|p\| \ll 1$, so we also have (4) here. Thus

$$E = T + R \approx \frac{\epsilon n}{2} L \|p\|^2 + \frac{2\epsilon_u L}{\epsilon},$$

and the optimal ϵ is $\epsilon \approx \frac{2}{\|p\|}\sqrt{\frac{\epsilon_u}{n}} \approx \frac{\sqrt{\epsilon}}{\|p\|}$.

(e) The complex method is generalized as

$$\nabla^2 f(x)p \approx \frac{\operatorname{Im}\left[\nabla f(x + i\epsilon p)\right]}{\epsilon}. \tag{8}$$

Further assuming the 4$^{\text{th}}$ derivative of f is bounded by L, we have

$$T \le \frac{\epsilon^2}{3!} L \left(\sum_i |p_i|\right)^3 \le \frac{\epsilon^2}{6} L \left(\sqrt{n}\|p\|\right)^3 \approx \epsilon^2 L \|p\|^3,$$

and $R \le \epsilon_u L$. So

$$E = T + R \approx \epsilon^2 L \|p\|^3 + \epsilon_u L$$

and we want ϵ to be as small as possible.

Table 1: Minimum found by the truncated Newton CG methods

	Hessian inversion			Infinite difference			Complex variable		
σ	k	$\|x_k\|$	f_k	k	$\|x_k\|$	f_k	k	$\|x_k\|$	f_k
0	1	0	0	2	1.4647 E^{-18}	1.0726 E^{-36}	1	0	0
1	8	8.7970 E^{-13}	3.8694 E^{-25}	8	8.2596 E^{-13}	3.4110 E^{-25}	5	2.0446 E^{-14}	2.0902 E^{-28}
10	11	1.1555 E^{-15}	6.6761 E^{-31}	11	1.6368 E^{-14}	1.3395 E^{-28}	7[a]	0	0

[a] $k = 6$ when the quadratic convergence is demarded.

3.4 The objective function is

$$f(x) = \frac{1}{2}x^T x + \frac{\sigma}{4}\left(x^T Ax\right)^2,$$

and A is a symmetric matrix. Then

$$\begin{aligned}\nabla f(x) &= x + \sigma\left(x^T Ax\right) Ax,\\ \nabla^2 f(x) &= I_4 + 2\sigma\,(Ax)\,(Ax)^T + \sigma\left(x^T Ax\right) A.\end{aligned}$$

It is easy to know that $x = \mathbf{0}$ is the global minimizer of $f(x)$. Thus we can use the norm of x_k as a criterion of how close x_k is to the minimizer.

We use $\epsilon = 10^{-12}$ for stopping condition of the outer Newton iteration, i.e.,

$$\|\nabla f_k\| \le \epsilon\,(1 + |f_k|)\,.$$

We use $\epsilon = 10^{-8}$ when calculating $\nabla^2 f_k p_k$ by Hessian-free methods. The results from the computer experiments are in Table 1. For the sake of short, we use E^n representing $\times 10^n$.

It is a little surprising that different convergence rates (linear, superlinear, and quadratic) come to the (almost) same results. It is also surprising that the Newton method converges faster when the complex variable approximation (8) is used, than when the exact inverse of $\nabla^2 f_k$ is used.

3.5 First, we will get a formula for τ when the intersect with the trust region is needed. Then some results and plots from the experiment are shown.

(a) During the inner linear CG, let $p^{(j)}$ and $d^{(j)}$ denote the CG solution and search direction. When negative curvature is detected at step i, that is, ${d^{(i)}}^T B_k d^{(i)} \le 0$, we have to find a τ such that $p = p^{(i)} + \tau d^{(i)}$ minimizes $m_k(p)$ with $\|p\| \le \Delta_k$. Since $p^{(i)}$ is spanned by $\left\{d^{(j)}\right\}_{j<i}$, ${p^{(i)}}^T Ad^{(i)} = 0$. Thus

$$\begin{aligned}m_k(p) &= f_k + \nabla f_k^T p + \frac{1}{2}p^T B_k p\\ &= \frac{1}{2}{d^{(i)}}^T B_k d^{(i)} \cdot \tau^2 + \nabla f_k^T d^{(i)} \cdot \tau + m_k\left(p^{(i)}\right).\end{aligned}$$

From Problem 2, we know $\nabla f_k^T d^{(i)} < 0$. Recall that ${d^{(i)}}^T B_k d^{(i)} \le 0$. So $m_k(p)$ gets its minimum when $\tau \ge 0$ and $\|p\| = \Delta_k$.

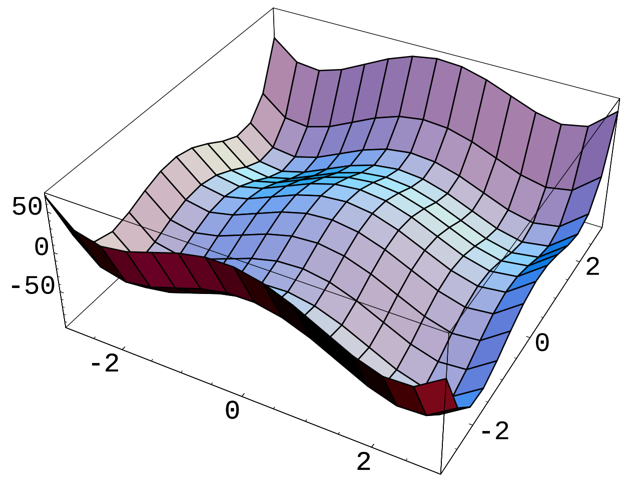


Figure 2: The function $f(x) = 2x_1^4 + 3x_2^4 - 20(x_1^2 + x_2^2) + 2x_1(x_2 - 1)$. It is easy to see that this function has 4 local minima, and also has some saddle points, which are non-minimizing stationary.

From $\|p\|^2 = \left\|p^{(i)}\right\|^2 + 2\tau p^{(i)T} d^{(i)} + \tau^2 \left\|d^{(i)}\right\|^2$, $\tau \geq 0$, and $\|p\| = \Delta_k$, we have[†]

$$\tau = \frac{-p^{(i)T} d^{(i)} + \sqrt{\left(p^{(i)T} d^{(i)}\right)^2 + \left\|d^{(i)}\right\|^2 \left(\Delta_k^2 - \left\|p^{(i)}\right\|^2\right)}}{\left\|d^{(i)}\right\|^2}.$$

(b) The objective function I used to be minimized is

$$f(x) = 2x_1^4 + 3x_2^4 - 20(x_1^2 + x_2^2) + 2x_1(x_2 - 1).$$

We have

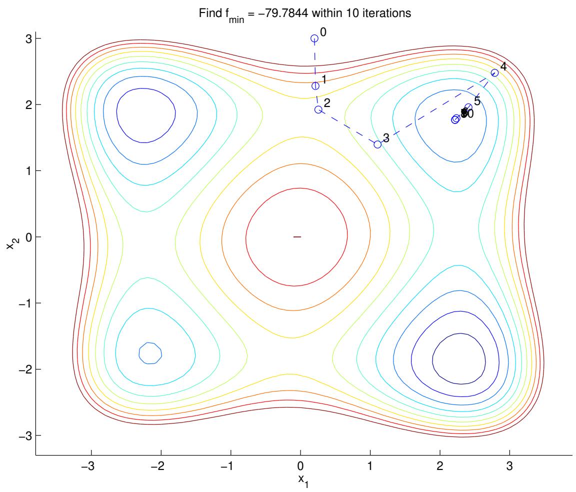
$$\nabla f(x) = \begin{bmatrix} 8x_1^3 - 40x_1 + 2x_2 - 2 \\ 2x_1 + 12x_2^3 - 40x_2 \end{bmatrix}, \quad \text{and } \nabla^2 f(x) = \begin{bmatrix} 24x_1^2 - 40 & 2 \\ 2 & 36x_2^2 - 40 \end{bmatrix}.$$

With $\eta = 0.2$, $\Delta_0 = 1$ and $\Delta_{\max} = 5$, some results from different initial points are show in Figure 3. We can see that since the function has 4 local minima, the result is very sensitive to the initial point. However, the saddle points do not prevent the trust region method from finding a local minimum.

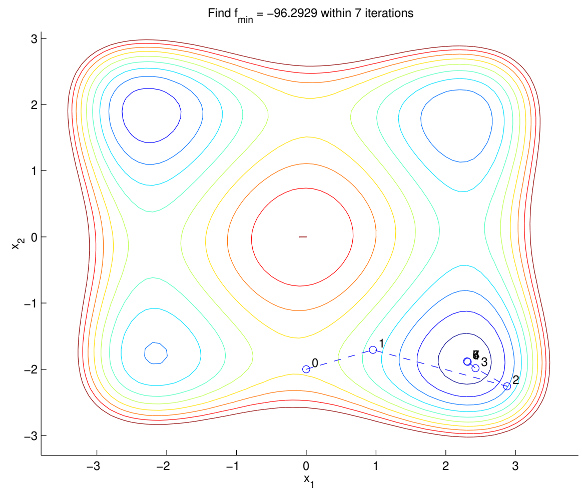
[†] Since $p^{(i)}$ is spanned by $\left\{d^{(j)}\right\}_{j<i}$, $p^{(i)T} r^{(i)} = 0$. From $p^{(0)T} d^{(0)} = 0$, and $\alpha^{(j)} > 0$ for $j < i$, we get

$$\begin{aligned} p^{(i)T} d^{(i)} &= p^{(i)T} \left(-r^{(i)} + \beta^{(i)} d^{(i-1)}\right) \\ &= \beta^{(i)} \left(p^{(i-1)} + \alpha^{(i-1)} d^{(i-1)}\right)^T d^{(i-1)} \\ &> \beta^{(i)} p^{(i-1)T} d^{(i-1)} > \cdots > \beta^{(1)} p^{(0)T} d^{(0)} = 0. \end{aligned}$$

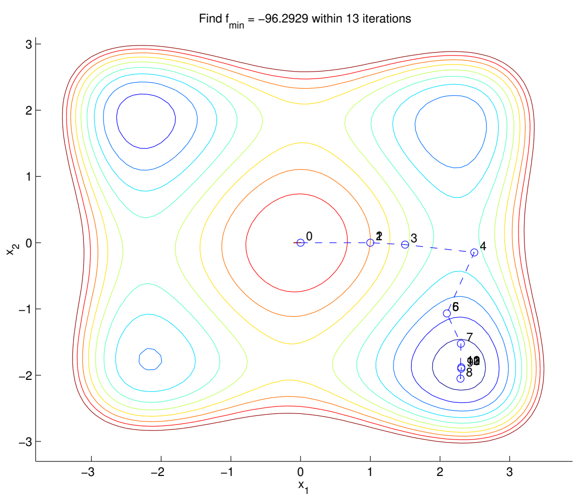
Thus the other solution to the equation $\|p\| = \Delta_k$ is negative.



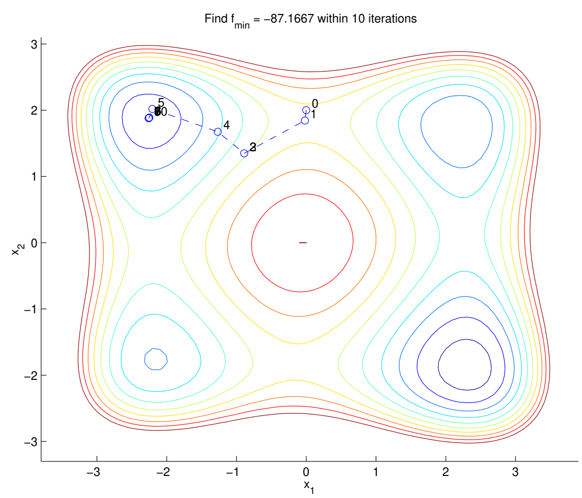
(a) From $(0.2, 3)^T$



(b) From $(0, -2)^T$



(c) From $(0, 0)^T$



(d) From $(0, 2)^T$

Figure 3: The trajectories of several runs, from different initial points.