The Steepest Descent Algorithm for Unconstrained Optimization and a Bisection Line-search Method

Robert M. Freund

February, 2004

1 The Algorithm

The problem we are interested in solving is:

$$\begin{array}{lll} P: & \text{minimize} & f(x) \\ & \text{s.t.} & x \in \Re^n, \end{array}$$

where $f(x)$ is differentiable. If $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

$$f(\bar{x} + d) \approx f(\bar{x}) + \nabla f(\bar{x})^T d$$

if d "small", i.e., if $\|d\|$ is small. Now notice that if the approximation in the above expression is good, then we want to choose d so that the inner product $\nabla f(\bar{x})^T d$ is as small as possible. Let us normalize d so that $\|d\| = 1$. Then among all directions d with norm $\|d\| = 1$, the direction

$$\tilde{d} = \frac{-\nabla f(\bar{x})}{\|\nabla f(\bar{x})\|}$$

makes the smallest inner product with the gradient $\nabla f(\bar{x})$. This fact follows from the following inequalities:

$$\nabla f(\bar{x})^T d \geq -\|\nabla f(\bar{x})\| \|d\| = \nabla f(\bar{x})^T \left(\frac{-\nabla f(\bar{x})}{\|\nabla f(\bar{x})\|} \right) = -\nabla f(\bar{x})^T \tilde{d}\ .$$

For this reason the un-normalized direction:

$$\bar{d} = -\nabla f(\bar{x})$$

is called the *direction of steepest descent* at the point $\bar{x}$.

Note that $\bar{d} = -\nabla f(\bar{x})$ is a descent direction as long as $\nabla f(\bar{x}) \neq 0$. To see this, simply observe that $\bar{d}^T \nabla f(\bar{x}) = -(\nabla f(\bar{x}))^T \nabla f(\bar{x}) < 0$ so long as $\nabla f(\bar{x}) \neq 0$.

A natural consequence of this is the following algorithm, called the *steepest descent algorithm*.

Steepest Descent Algorithm:

Step 0. Given x^0, set $k := 0$

Step 1. $d^k := -\nabla f(x^k)$. If $d^k = 0$, then stop.

Step 2. Solve $\min_\alpha f(x^k + \alpha d^k)$ for the stepsize α^k, perhaps chosen by an exact or inexact linesearch.

Step 3. Set $x^{k+1} \leftarrow x^k + \alpha^k d^k, k \leftarrow k + 1$. Go to **Step 1.**

Note from Step 2 and the fact that $d^k = -\nabla f(x^k)$ is a descent direction, it follows that $f(x^{k+1}) < f(x^k)$.

2 Global Convergence

We have the following theorem:

Convergence Theorem: Suppose that $f(\cdot) : \Re^n \to \Re$ is continuously differentiable on the set $S = \{x \in \Re^n \mid f(x) \leq f(x^0)\}$, and that S is a closed and bounded set. Then every point $\bar{x}$ that is a cluster point of the sequence $\{x^k\}$ satisfies $\nabla f(\bar{x}) = 0$.

Proof: The proof of this theorem is by contradiction. By the Weierstrass Theorem, at least one cluster point of the sequence $\{x^k\}$ must exist. Let $\bar{x}$ be any such cluster point. Without loss of generality, assume that $\lim_{k\to\infty} x^k = \bar{x}$, but that $\nabla f(\bar{x}) \neq 0$. This being the case, there is a value of $\bar{\alpha} > 0$ such that $\delta := f(\bar{x}) - f(\bar{x} + \bar{\alpha}\bar{d}) > 0$, where $\bar{d} = -\nabla f(\bar{x})$. Then also $(\bar{x} + \bar{\alpha}\bar{d}) \in intS$.

Now $\lim_{k\to\infty} d^k = \bar{d}$. Then since $(\bar{x} + \bar{\alpha}\bar{d}) \in intS$, and $(x^k + \bar{\alpha}d^k) \to (\bar{x} + \bar{\alpha}\bar{d})$, for k sufficiently large we have

$$f(x^k + \bar{\alpha}d^k) \le f(\bar{x} + \bar{\alpha}\bar{d}) + \frac{\delta}{2} = f(\bar{x}) - \delta + \frac{\delta}{2} = f(\bar{x}) - \frac{\delta}{2}.$$

However,

$$f(\bar{x}) < f(x^k + \alpha^k d^k) \le f(x^k + \bar{\alpha}d^k) \le f(\bar{x}) - \frac{\delta}{2},$$

which is of course a contradiction. Thus $\bar{d} = -\nabla f(\bar{x}) = 0$.
q.e.d.

3 The Rate of Convergence for the Case of a Quadratic Function

In this section we explore answers to the question of how fast the steepest descent algorithm converges. We say that an algorithm exhibits *linear convergence* in the objective function values if there is a constant $\delta < 1$ such that for all k sufficiently large, the iterates x^k satisfy:

$$\frac{f(x^{k+1}) - f(x^*)}{f(x^k) - f(x^*)} \le \delta,$$

where x^* is some optimal value of the problem P. The above statement says that the optimality gap shrinks by at least δ at each iteration. Notice that if $\delta = 0.1$, for example, then the iterates gain an extra digit of accuracy in the optimal objective function value at each iteration. If $\delta = 0.9$, for example, then the iterates gain an extra digit of accuracy in the optimal objective function value every 22 iterations, since $(0.9)^{22} \approx 0.10$.

The quantity δ above is called the *convergence constant*. We would like this constant to be smaller rather than larger.

We will show now that the steepest descent algorithm exhibits linear convergence, but that the convergence constant depends very much on the ratio of the largest to the smallest eigenvalue of the Hessian matrix $H(x)$ at the optimal solution $x = x^*$. In order to see how this arises, we will examine the case where the objective function $f(x)$ is itself a simple quadratic function of the form:

$$f(x) = \frac{1}{2}x^T Q x + q^T x,$$

where Q is a positive definite symmetric matrix. We will suppose that the eigenvalues of Q are

$$A = a_1 \geq a_2 \geq \ldots \geq a_n = a > 0,$$

i.e, A and a are the largest and smallest eigenvalues of Q.

The optimal solution of P is easily computed as:

$$x^* = -Q^{-1}q$$

and direct substitution shows that the optimal objective function value is:

$$f(x^*) = -\frac{1}{2}q^T Q^{-1} q.$$

For convenience, let x denote the current point in the steepest descent algorithm. We have:

$$f(x) = \frac{1}{2}x^T Q x + q^T x$$

and let d denote the current direction, which is the negative of the gradient, i.e.,

$$d = -\nabla f(x) = -Qx - q.$$

Now let us compute the next iterate of the steepest descent algorithm. If α is the generic step-length, then

$$f(x + \alpha d) = \frac{1}{2}(x + \alpha d)^T Q (x + \alpha d) + q^T (x + \alpha d)$$

$$= \frac{1}{2}x^TQx + \alpha d^TQx + \frac{1}{2}\alpha^2 d^TQd + q^Tx + \alpha q^Td$$
$$= f(x) - \alpha d^Td + \frac{1}{2}\alpha^2 d^TQd.$$

Optimizing the value of α in this last expression yields

$$\alpha = \frac{d^Td}{d^TQd},$$

and the next iterate of the algorithm then is

$$x^{'} = x + \alpha d = x + \frac{d^Td}{d^TQd}d,$$

and

$$f(x^{'}) = f(x+\alpha d) = f(x) - \alpha d^Td + \frac{1}{2}\alpha^2 d^TQd = f(x) - \frac{1}{2}\frac{(d^Td)^2}{d^TQd}.$$

Therefore,

$$\frac{f(x^{'}) - f(x^*)}{f(x) - f(x^*)} = \frac{f(x) - \frac{1}{2}\frac{(d^Td)^2}{d^TQd} - f(x^*)}{f(x) - f(x^*)}$$
$$= 1 - \frac{\frac{1}{2}\frac{(d^Td)^2}{d^TQd}}{\frac{1}{2}x^TQx + q^Tx + \frac{1}{2}q^TQ^{-1}q}$$
$$= 1 - \frac{\frac{1}{2}\frac{(d^Td)^2}{d^TQd}}{\frac{1}{2}(Qx+q)^TQ^{-1}(Qx+q)}$$
$$= 1 - \frac{(d^Td)^2}{(d^TQd)(d^TQ^{-1}d)}$$
$$= 1 - \frac{1}{\beta}$$

where

$$\beta = \frac{(d^TQd)(d^TQ^{-1}d)}{(d^Td)^2}.$$

In order for the convergence constant to be good, which will translate to fast linear convergence, we would like the quantity β to be small. The following result provides an upper bound on the value of β.

Kantorovich Inequality: Let A and a be the largest and the smallest eigenvalues of Q, respectively. Then

$$\beta \leq \frac{(A+a)^2}{4Aa}.$$

We will prove this inequality later. For now, let us apply this inequality to the above analysis. Continuing, we have

$$\frac{f(x^{'}) - f(x^*)}{f(x) - f(x^*)} = 1 - \frac{1}{\beta} \leq 1 - \frac{4Aa}{(A+a)^2} = \frac{(A-a)^2}{(A+a)^2} = \left(\frac{A/a - 1}{A/a + 1}\right)^2 =: \delta \ .$$

Note by definition that A/a is always at least 1. If A/a is small (not much bigger than 1), then the convergence constant δ will be much smaller than 1. However, if A/a is large, then the convergence constant δ will be only slightly smaller than 1. Table 1 shows some sample values. Note that the number of iterations needed to reduce the optimality gap by a factor of 10 grows linearly in the ratio A/a.

4 Examples

4.1 Example 1: Typical Behavior

Consider the function $f(x_1, x_2) = 5x_1^2 + x_2^2 + 4x_1x_2 - 14x_1 - 6x_2 + 20$. This function has its optimal solution at $x^* = (x_1^*, x_2^*) = (1, 1)$ and $f(1, 1) = 10$.

In step 1 of the steepest descent algorithm, we need to compute

$$d^k = -\nabla f(x_1^k, x_2^k) = \begin{pmatrix} -10x_1^k - 4x_2^k + 14 \\ -2x_2^k - 4x_1^k + 6 \end{pmatrix} = \begin{pmatrix} d_1^k \\ d_2^k \end{pmatrix}$$

A	a	Upper Bound on $\delta = \left(\frac{A/a-1}{A/a+1}\right)^2$	Number of Iterations to Reduce the Optimality Gap by a factor of 10
1.1	1.0	0.0023	1
3.0	1.0	0.25	2
10.0	1.0	0.67	6
100.0	1.0	0.96	58
200.0	1.0	0.98	116
400.0	1.0	0.99	231

Table 1: Sensitivity of Steepest Descent Convergence Rate to the Eigenvalue Ratio

and in step 2 we need to solve $\alpha^k = \arg\min_\alpha h(\alpha) = f(x^k + \alpha d^k)$. In this example we will be able to derive an analytic expression for α^k. Notice that

$$\begin{aligned} h(\alpha) &= f(x^k + \alpha d^k) \\ &= 5(x_1^k + \alpha d_1^k)^2 + (x_2^k + \alpha d_2^k)^2 + 4(x_1^k + \alpha d_1^k)(x_2^k + \alpha d_2^k) - \\ &\quad -14(x_1^k + \alpha d_1^k) - 6(x_2^k + \alpha d_2^k) + 20, \end{aligned}$$

and this is a simple quadratic function of the scalar α. It is minimized at

$$\alpha^k = \frac{(d_1^k)^2 + (d_2^k)^2}{2(5(d_1^k)^2 + (d_2^k)^2 + 4d_1^k d_2^k)}$$

Using the steepest descent algorithm to minimize $f(x)$ starting from $x^1 = (x_1^1, x_2^1) = (0, 10)$, and using a tolerance of $\epsilon = 10^{-6}$, we compute the iterates shown in Table 2 and in Figure 2:

For a convex quadratic function $f(x) = \frac{1}{2}x^T Qx - c^T x$, the contours of the function values will be shaped like ellipsoids, and the gradient vector $\nabla f(x)$ at any point x will be perpendicular to the contour line passing through x, see Figure 1.

k	x_1^k	x_2^k	d_1^k	d_2^k	$\|\|d^k\|\|_2$	α^k	$f(x^k)$
1	0.000000	10.000000	-26.000000	-14.000000	29.52964612	0.0866	60.000000
2	-2.252782	8.786963	1.379968	-2.562798	2.91071234	2.1800	22.222576
3	0.755548	3.200064	-6.355739	-3.422321	7.21856659	0.0866	12.987827
4	0.204852	2.903535	0.337335	-0.626480	0.71152803	2.1800	10.730379
5	0.940243	1.537809	-1.553670	-0.836592	1.76458951	0.0866	10.178542
6	0.805625	1.465322	0.082462	-0.153144	0.17393410	2.1800	10.043645
7	0.985392	1.131468	-0.379797	-0.204506	0.43135657	0.0866	10.010669
8	0.952485	1.113749	0.020158	-0.037436	0.04251845	2.1800	10.002608
9	0.996429	1.032138	-0.092842	-0.049992	0.10544577	0.0866	10.000638
10	0.988385	1.027806	0.004928	-0.009151	0.01039370	2.1800	10.000156
11	0.999127	1.007856	-0.022695	-0.012221	0.02577638	0.0866	10.000038
12	0.997161	1.006797	0.001205	-0.002237	0.00254076	2.1800	10.000009
13	0.999787	1.001920	-0.005548	-0.002987	0.00630107	0.0866	10.000002
14	0.999306	1.001662	0.000294	-0.000547	0.00062109	2.1800	10.000001
15	0.999948	1.000469	-0.001356	-0.000730	0.00154031	0.0866	10.000000
16	0.999830	1.000406	0.000072	-0.000134	0.00015183	2.1800	10.000000
17	0.999987	1.000115	-0.000332	-0.000179	0.00037653	0.0866	10.000000
18	0.999959	1.000099	0.000018	-0.000033	0.00003711	2.1800	10.000000
19	0.999997	1.000028	-0.000081	-0.000044	0.00009204	0.0866	10.000000
20	0.999990	1.000024	0.000004	-0.000008	0.00000907	2.1803	10.000000
21	0.999999	1.000007	-0.000020	-0.000011	0.00002250	0.0866	10.000000
22	0.999998	1.000006	0.000001	-0.000002	0.00000222	2.1817	10.000000
23	1.000000	1.000002	-0.000005	-0.000003	0.00000550	0.0866	10.000000
24	0.999999	1.000001	0.000000	-0.000000	0.00000054	0.0000	10.000000

Table 2: Iterations of the example in Subsection 4.1.

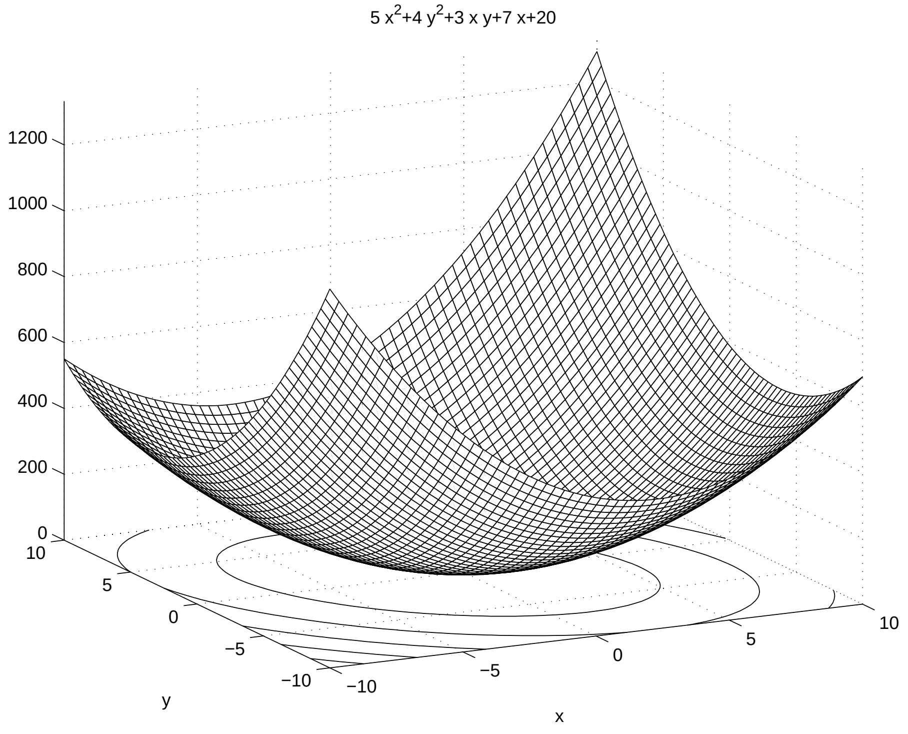

Figure 1: The contours of a convex quadratic function are ellipsoids.

4.2 Example 2: Hem-Stitching

Consider the function

$$f(x) = \frac{1}{2}x^T Q x - c^T x + 10$$

where Q and c are given by:

$$Q = \begin{pmatrix} 20 & 5 \\ 5 & 2 \end{pmatrix}$$

and

$$c = \begin{pmatrix} 14 \\ 6 \end{pmatrix}$$

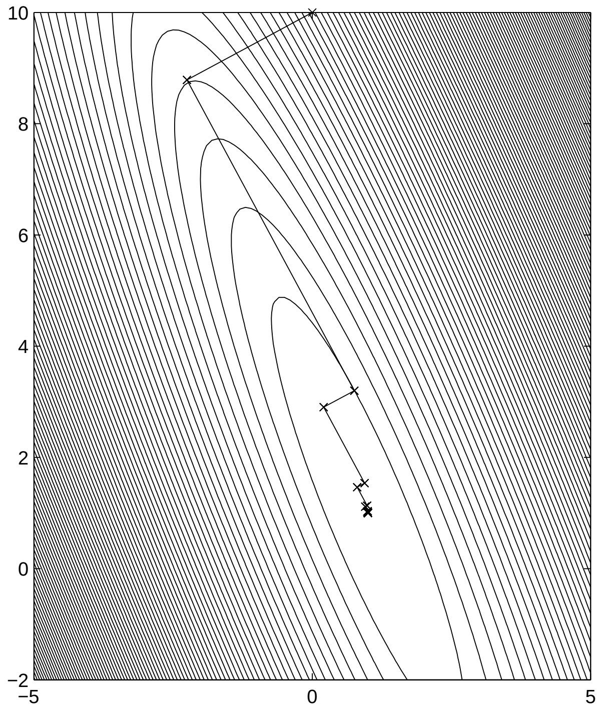

Figure 2: Behavior of the steepest descent algorithm for the example of Subsection 4.1.

For this problem, $\frac{A}{a} = 30.234$, and so the linear convergence rate δ of the steepest descent algorithm is bounded above by $\delta = \left(\frac{A/a-1}{A/a+1}\right)^2 = 0.8760$.

If we apply the steepest descent algorithm to minimize $f(x)$ starting from $x^1 = (40, -100)$, we obtain the iteration values shown in Table 3 and shown graphically in Figure 3.

4.3 Example 3: Rapid Convergence

Consider the function

$$f(x) = \frac{1}{2}x^T Qx - c^T x + 10$$

k	x_1^k	x_2^k	$\|\|d^k\|\|_2$	α^k	$f(x^k)$	$\frac{f(x^k)-f(x^*)}{f(x^{k-1})-f(x^*)}$
1	40.000000	-100.000000	286.06293014	0.0506	6050.000000	
2	25.542693	-99.696700	77.69702948	0.4509	3981.695128	0.658079
3	26.277558	-64.668130	188.25191488	0.0506	2620.587793	0.658079
4	16.763512	-64.468535	51.13075844	0.4509	1724.872077	0.658079
5	17.247111	-41.416980	123.88457127	0.0506	1135.420663	0.658079
6	10.986120	-41.285630	33.64806192	0.4509	747.515255	0.658079
7	11.304366	-26.115894	81.52579489	0.0506	492.242977	0.658079
8	7.184142	-26.029455	22.14307211	0.4509	324.253734	0.658079
9	7.393573	-16.046575	53.65038732	0.0506	213.703595	0.658079
10	4.682141	-15.989692	14.57188362	0.4509	140.952906	0.658079
20	0.460997	0.948466	1.79847660	0.4509	3.066216	0.658079
30	-0.059980	3.038991	0.22196980	0.4509	0.965823	0.658079
40	-0.124280	3.297005	0.02739574	0.4509	0.933828	0.658079
50	-0.132216	3.328850	0.00338121	0.4509	0.933341	0.658079
60	-0.133195	3.332780	0.00041731	0.4509	0.933333	0.658078
70	-0.133316	3.333265	0.00005151	0.4509	0.933333	0.658025
80	-0.133331	3.333325	0.00000636	0.4509	0.933333	0.654656
90	-0.133333	3.333332	0.00000078	0.0000	0.933333	0.000000

Table 3: Iteration values for example of Subsection 4.2, which shows hemstitching.

where Q and c are given by:

$$Q = \begin{pmatrix} 20 & 5 \\ 5 & 16 \end{pmatrix}$$

and

$$c = \begin{pmatrix} 14 \\ 6 \end{pmatrix}$$

For this problem, $\frac{A}{a} = 1.8541$, and so the linear convergence rate δ of the steepest descent algorithm is bounded above by $\delta = \left(\frac{\frac{A}{a}-1}{\frac{A}{a}+1}\right)^2 = 0.0896$.

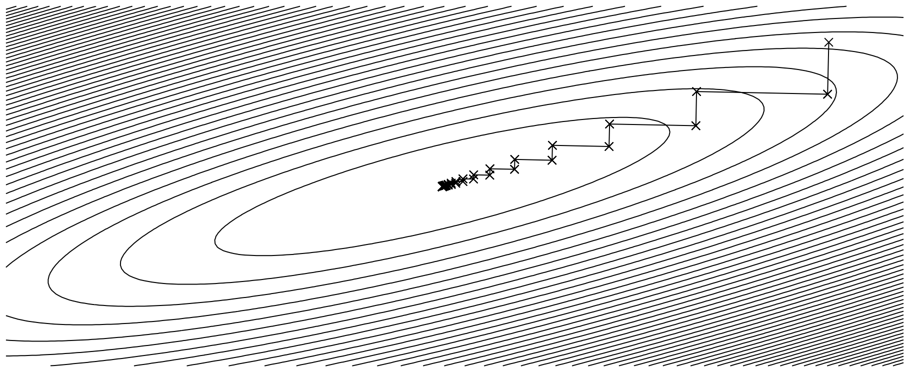

Figure 3: Hem-stitching in the example of Subsection 4.2.

If we apply the steepest descent algorithm to minimize $f(x)$ starting from $x^1 = (40, -100)$, we obtain the iteration values shown in Table 4 and shown graphically in Figure 4.

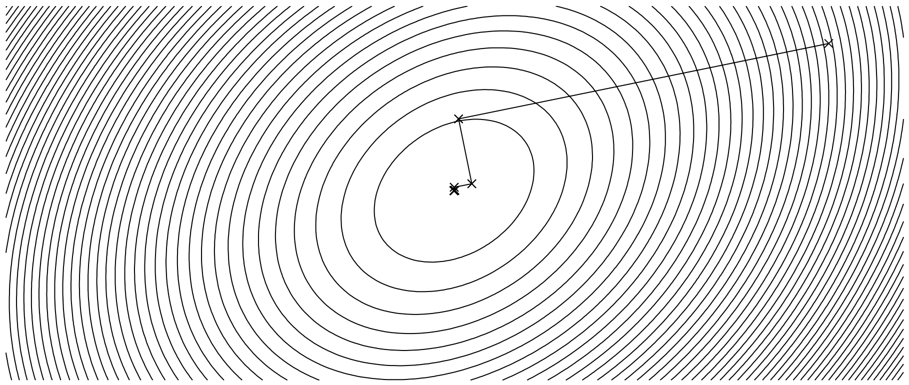

Figure 4: Rapid convergence in the example of Subsection 4.3.

k	x_1^k	x_2^k	$\|\|d^k\|\|_2$	α^k	$f(x^k)$	$\frac{f(x^k)-f(x^*)}{f(x^{k-1})-f(x^*)}$
1	40.000000	−100.000000	1434.79336491	0.0704	76050.000000	
2	19.867118	−1.025060	385.96252652	0.0459	3591.615327	0.047166
3	2.513241	−4.555081	67.67315150	0.0704	174.058930	0.047166
4	1.563658	0.113150	18.20422450	0.0459	12.867208	0.047166
5	0.745149	−0.053347	3.19185713	0.0704	5.264475	0.047166
6	0.700361	0.166834	0.85861649	0.0459	4.905886	0.047166
7	0.661755	0.158981	0.15054644	0.0704	4.888973	0.047166
8	0.659643	0.169366	0.04049732	0.0459	4.888175	0.047166
9	0.657822	0.168996	0.00710064	0.0704	4.888137	0.047166
10	0.657722	0.169486	0.00191009	0.0459	4.888136	0.047166
11	0.657636	0.169468	0.00033491	0.0704	4.888136	0.047166
12	0.657632	0.169491	0.00009009	0.0459	4.888136	0.047161
13	0.657628	0.169490	0.00001580	0.0704	4.888136	0.047068
14	0.657627	0.169492	0.00000425	0.0459	4.888136	0.045002
15	0.657627	0.169491	0.00000075	0.0000	4.888136	0.000000

Table 4: Iteration values for the example in Subsection 4.3, which exhibits rapid convergence.

4.4 Example 4: A nonquadratic function

Let

$$f(x) = x_1 - 0.6x_2 + 4x_3 + 0.25x_4 - \sum_{i=1}^{4} \log(x_i) - \log(5 - \sum_{i=1}^{4} x_i) \ .$$

Table 5 shows values of the iterates of steepest descent applied to this function.

4.5 Example 5: An analytic example

Suppose that

$$f(x) = \frac{1}{2}x^T Q x + q^T x$$

k	x_1^k	x_2^k	x_3^k	x_4^k	$\|\|d^k\|\|_2$	$f(x^k)$	$\frac{f(x^k)-f(x^*)}{f(x^{k-1})-f(x^*)}$
1	1.000000	1.000000	1.000000	1.000000	4.17402683	4.650000	
2	0.802973	1.118216	0.211893	0.950743	1.06742574	2.276855	0.219505
3	0.634501	1.710704	0.332224	1.121308	1.88344095	1.939389	0.494371
4	0.616707	1.735137	0.205759	1.108079	0.46928947	1.797957	0.571354
5	0.545466	1.972851	0.267358	1.054072	1.15992055	1.739209	0.688369
6	0.543856	1.986648	0.204195	1.044882	0.31575186	1.698071	0.682999
7	0.553037	2.121519	0.241186	0.991524	0.80724427	1.674964	0.739299
8	0.547600	2.129461	0.202091	0.983563	0.23764416	1.657486	0.733266
9	0.526273	2.205845	0.229776	0.938380	0.58321024	1.646479	0.770919
10	0.525822	2.212935	0.202342	0.933770	0.18499125	1.637766	0.764765
20	0.511244	2.406747	0.200739	0.841546	0.06214411	1.612480	0.803953
30	0.503892	2.468138	0.200271	0.813922	0.02150306	1.609795	0.806953
40	0.501351	2.489042	0.200095	0.804762	0.00743136	1.609480	0.807905
50	0.500467	2.496222	0.200033	0.801639	0.00256658	1.609443	0.808228
60	0.500161	2.498696	0.200011	0.800565	0.00088621	1.609439	0.808338
70	0.500056	2.499550	0.200004	0.800195	0.00030597	1.609438	0.808376
80	0.500019	2.499845	0.200001	0.800067	0.00010564	1.609438	0.808381
90	0.500007	2.499946	0.200000	0.800023	0.00003647	1.609438	0.808271
100	0.500002	2.499981	0.200000	0.800008	0.00001257	1.609438	0.806288
110	0.500001	2.499993	0.200000	0.800003	0.00000448	1.609438	0.813620
120	0.500000	2.499998	0.200000	0.800001	0.00000145	1.609438	0.606005
121	0.500000	2.499998	0.200000	0.800001	0.00000455	1.609438	0.468218
122	0.500000	2.499998	0.200000	0.800001	0.00000098	1.609438	0.000000

Table 5: Iteration values for the example of Subsection 4.4.

where

$$Q = \begin{pmatrix} +4 & -2 \\ -2 & +2 \end{pmatrix} \quad \text{and} \quad q = \begin{pmatrix} +2 \\ -2 \end{pmatrix}.$$

Then

$$\nabla f(x) = \begin{pmatrix} +4 & -2 \\ -2 & +2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} +2 \\ -2 \end{pmatrix}$$

and so

$$x^* = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

and

$$f(x^*) = -1.$$

Direct computation shows that the eigenvalues of Q are $A = 3 + \sqrt{5}$ and $a = 3 - \sqrt{5}$, whereby the bound on the convergence constant is

$$\delta = \left(\frac{A/a - 1}{A/a + 1} \right)^2 \leq 0.556.$$

Suppose that $x^0 = (0, 0)$. Then we have:

$$x^1 = (-0.4, 0.4), \qquad x^2 = (0, 0.8)$$

and the even numbered iterates satisfy

$$x^{2n} = (0, 1 - 0.2^n) \quad \text{and} \quad f(x^{2n}) = (1 - 0.2^n)^2 - 2 + 2(0.2)^n$$

and so

$$f(x^{2n}) - f(x^*) = (0.2)^{2n}.$$

Therefore, starting from the point $x^0 = (0, 0)$, the optimality gap goes down by a factor of 0.04 after every two iterations of the algorithm.

5 Final Remarks

- The proof of convergence is basic in that it only relies on fundamental properties of the continuity of the gradient and on the closedness and boundedness of the level sets of the function $f(x)$.
- The analysis of the rate of convergence is quite elegant.

- Kantorovich won the Nobel Prize in Economics in 1975 for his contributions to the application of linear programming to economic theory.
- The bound on the rate of convergence *is* attained in practice quite often, which is too bad.
- The ratio of the largest to the smallest eigenvalue of a matrix is called the *condition number* of the matrix. The condition number plays an extremely important role in the theory and the computation of quantities involving the matrix Q.
- What about non-quadratic functions? Most functions behave as near-quadratic functions in a neighborhood of the optimal solution. The analysis of the non-quadratic case gets very involved; fortunately, the key intuition is obtained by analyzing the quadratic case.

6 A Bisection Algorithm for a Line-Search of a Convex Function

Suppose that $f(x)$ is a continuously differentiable convex function, and that we seek to solve:

$$\bar{\alpha} := \arg\min_{\alpha} f(\bar{x} + \alpha \bar{d}),$$

where $\bar{x}$ is our current iterate, and $\bar{d}$ is the current direction generated by an algorithm that seeks to minimize $f(x)$. Suppose further that $\bar{d}$ is a descent direction of $f(x)$ at $x = \bar{x}$, namely:

$$f(\bar{x} + \epsilon \bar{d}) < f(\bar{x}) \text{ for all } \epsilon > 0 \text{ and sufficiently small }.$$

Let

$$h(\alpha) := f(\bar{x} + \alpha \bar{d}),$$

whereby $h(\alpha)$ is a convex function in the scalar variable α, and our problem is to solve for

$$\bar{\alpha} = \arg\min_{\alpha} h(\alpha).$$

We therefore seek a value $\bar{\alpha}$ for which

$$h^{'}(\bar{\alpha}) = 0.$$

It is elementary to show that

$$h^{'}(\alpha) = \nabla f(\bar{x} + \alpha \bar{d})^T \bar{d}.$$

Proposition 6.1 $h^{'}(0) < 0$.
q.e.d.

Because $h(\alpha)$ is a convex function of α, we also have:

Proposition 6.2 $h^{'}(\alpha)$ *is a monotone increasing function of* α.
q.e.d.

Figure 5 shows an example of convex function of two variables to be optimized. Figure 6 shows the function $h(\alpha)$ obtained by restricting the function of Figure 5 to the line shown in that figure. Note from Figure 6 that $h(\alpha)$ is convex. Therefore its first derivative $h^{'}(\alpha)$ will be a monotonically increasing function. This is shown in Figure 7.

Because $h^{'}(\alpha)$ is a monotonically increasing function, we can approximately compute $\bar{\alpha}$, the point that satisfies $h^{'}(\bar{\alpha}) = 0$, by a suitable bisection method. Suppose that we know a value $\hat{\alpha}$ that $h^{'}(\hat{\alpha}) > 0$. Since $h^{'}(0) < 0$ and $h^{'}(\hat{\alpha}) > 0$, the mid-value $\tilde{\alpha} = \frac{0+\hat{\alpha}}{2}$ is a suitable test-point. Note the following:

- If $h^{'}(\tilde{\alpha}) = 0$, we are done.
- If $h^{'}(\tilde{\alpha}) > 0$, we can now bracket $\bar{\alpha}$ in the interval $(0, \tilde{\alpha})$.
- If $h^{'}(\tilde{\alpha}) < 0$, we can now bracket $\bar{\alpha}$ in the interval $(\tilde{\alpha}, \hat{\alpha})$.

This leads to the following *bisection algorithm* for minimizing $h(\alpha) = f(\bar{x} + \alpha \bar{d})$ by solving the equation $h^{'}(\alpha) \approx 0$.

Step 0. Set $k = 0$. Set $\alpha_l := 0$ and $\alpha_u := \hat{\alpha}$.

Step k. Set $\tilde{\alpha} = \frac{\alpha_u + \alpha_l}{2}$ and compute $h^{'}(\tilde{\alpha})$.

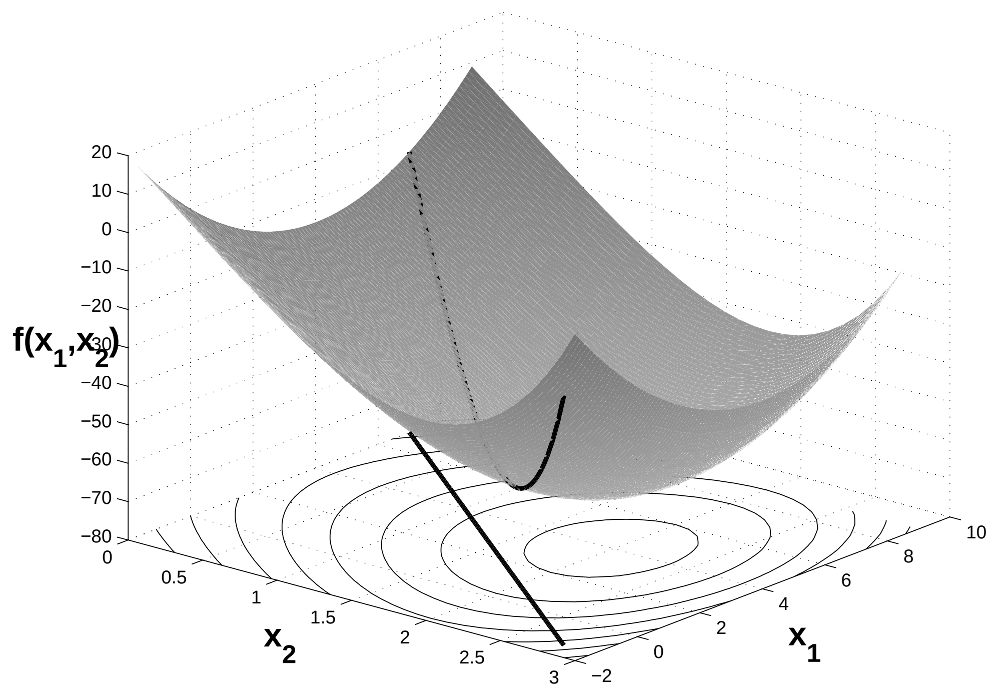

Figure 5: A convex function to be optimized.

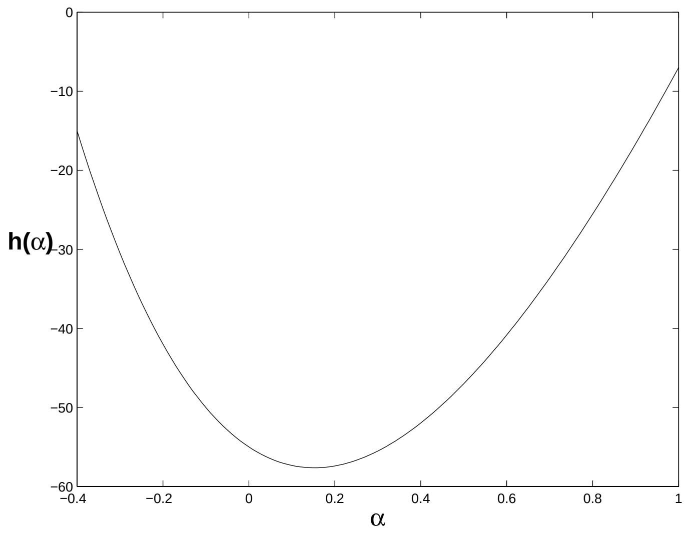

Figure 6: The 1-dimensional function $h(\alpha)$.

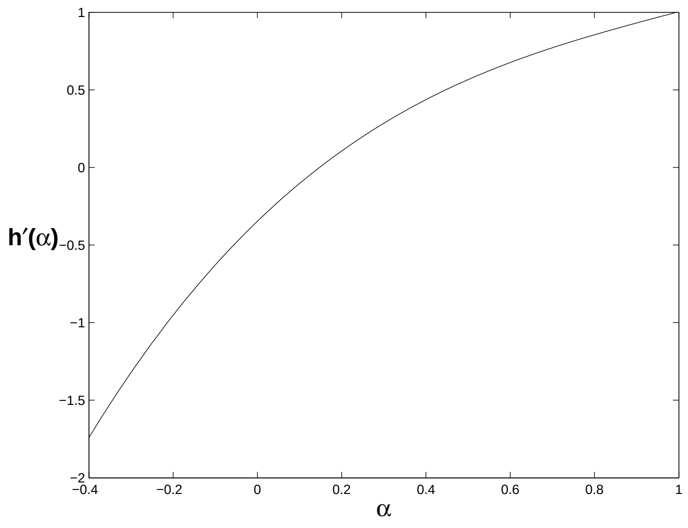

Figure 7: The function $h^{'}(\alpha)$ is monotonically increasing.

- If $h^{'}(\tilde{\alpha}) > 0$, re-set $\alpha_u := \tilde{\alpha}$. Set $k \leftarrow k + 1$.
- If $h^{'}(\tilde{\alpha}) < 0$, re-set $\alpha_l := \tilde{\alpha}$. Set $k \leftarrow k + 1$.
- If $h^{'}(\tilde{\alpha}) = 0$, stop.

Proposition 6.3 *After every iteration of the bisection algorithm, the current interval $[\alpha_l, \alpha_u]$ must contain a point $\bar{\alpha}$ such that $h^{'}(\bar{\alpha}) = 0$.*

Proposition 6.4 *At the k^{th} iteration of the bisection algorithm, the length of the current interval $[\alpha_l, \alpha_u]$ is*

$$L = \left(\frac{1}{2}\right)^k (\hat{\alpha}).$$

Proposition 6.5 *A value of α such that $|\alpha - \bar{\alpha}| \le \epsilon$ can be found in at most*

$$\left\lceil \log_2 \left(\frac{\hat{\alpha}}{\epsilon}\right) \right\rceil$$

steps of the bisection algorithm.

6.1 Computing $\hat{\alpha}$ for which $h^{'}(\hat{\alpha}) > 0$

Suppose that we do not have available a convenient value $\hat{\alpha}$ for which $h^{'}(\hat{\alpha}) > 0$. One way to proceed is to pick an initial "guess" of $\hat{\alpha}$ and compute $h^{'}(\hat{\alpha})$. If $h^{'}(\hat{\alpha}) > 0$, then proceed to the bisection algorithm; if $h^{'}(\hat{\alpha}) \le 0$, then re-set $\hat{\alpha} \leftarrow 2\hat{\alpha}$ and repeat the process.

6.2 Stopping Criteria for the Bisection Algorithm

In practice, we need to run the bisection algorithm with a stopping criterion. Some relevant stopping criteria are:

- Stop after a fixed number of iterations. That is stop when $k = \bar{K}$, where $\bar{K}$ specified by the user.
- Stop when the interval becomes small. That is, stop when $\alpha_u - \alpha_l \le \epsilon$, where ϵ is specified by the user.
- Stop when $|h^{'}(\tilde{\alpha})|$ becomes small. That is, stop when $|h^{'}(\tilde{\alpha})| \le \epsilon$, where ϵ is specified by the user.

This third stopping criterion typically yields the best results in practice.

6.3 Modification of the Bisection Algorithm when the Domain of $f(x)$ is Restricted

The discussion and analysis of the bisection algorithm has presumed that our optimization problem is

$$P: \quad \text{minimize} \quad f(x)$$

$$\text{s.t.} \qquad x \in \Re^n.$$

Given a point $\bar{x}$ and a direction $\bar{d}$, the line-search problem then is

$$LS: \quad \text{minimize} \quad h(\alpha) := f(\bar{x} + \alpha\bar{d})$$

$$\text{s.t.} \qquad \alpha \in \Re.$$

Suppose instead that the domain of definition of $f(x)$ is an open set $X \subset \Re^n$. Then our optimization problem is:

$$P: \quad \text{minimize} \quad f(x)$$

$$\text{s.t.} \qquad x \in X,$$

and the line-search problem then is

$$LS: \quad \text{minimize} \quad h(\alpha) := f(\bar{x} + \alpha\bar{d})$$

$$\text{s.t.} \qquad \bar{x} + \alpha\bar{d} \in X.$$

In this case, we must ensure that all iterate values of α in the bisection algorithm satisfy $\bar{x} + \alpha\bar{d} \in X$. As an example, consider the following problem:

$$P: \quad \text{minimize} \quad f(x) := -\sum_{i=1}^{m} \ln(b_i - A_i x)$$

$$\text{s.t.} \qquad b - Ax > 0.$$

Here the domain of $f(x)$ is $X = \{x \in \Re^n \mid b - Ax > 0\}$. Given a point $\bar{x} \in X$ and a direction $\bar{d}$, the line-search problem is:

$$\begin{array}{lll} LS: & \text{minimize} & h(\alpha) := f(\bar{x} + \alpha\bar{d}) = -\sum_{i=1}^{m} \ln(b_i - A_i(\bar{x} + \alpha\bar{d})) \\ & \text{s.t.} & b - A(\bar{x} + \alpha\bar{d}) > 0. \end{array}$$

Standard arithmetic manipulation can be used to establish that

$$b - A(\bar{x} + \alpha\bar{d}) > 0 \ \text{ if and only if } \ \check{\alpha} < \alpha < \hat{\alpha}$$

where

$$\check{\alpha} := -\min_{A_i\bar{d}<0} \left\{ \frac{b_i - A_i\bar{x}}{-A_i\bar{d}} \right\} \quad \text{and} \quad \hat{\alpha} := \min_{A_i\bar{d}>0} \left\{ \frac{b_i - A_i\bar{x}}{A_i\bar{d}} \right\},$$

and the line-search problem then is:

$$\begin{array}{lll} LS: & \text{minimize} & h(\alpha) := -\sum_{i=1}^{m} \ln(b_i - A_i(\bar{x} + \alpha\bar{d})) \\ & \text{s.t.} & \check{\alpha} < \alpha < \hat{\alpha}. \end{array}$$

7 Proof of Kantorovich Inequality

Kantorovich Inequality: Let A and a be the largest and the smallest eigenvalues of Q, respectively. Then

$$\beta \leq \frac{(A+a)^2}{4Aa}.$$

Proof: Let $Q = RDR^T$, and then $Q^{-1} = RD^{-1}R^T$, where $R = R^T$ is an orthonormal matrix, and the eigenvalues of Q are

$$0 < a = a_1 \leq a_2 \leq \ldots \leq a_n = A,$$

and,

$$D = \begin{pmatrix} a_1 & 0 & \ldots & 0 \\ 0 & a_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_n \end{pmatrix}.$$

Then

$$\beta = \frac{(d^TRDR^Td)(d^TRD^{-1}R^Td)}{(d^TRR^Td)(d^TRR^Td)} = \frac{f^TDff^TD^{-1}f}{f^Tff^Tf}$$

where $f = R^Td$. Let $\lambda_i = \frac{f_i^2}{f^Tf}$. Then $\lambda_i \geq 0$ and $\sum\limits_{i=1}^{n} \lambda_i = 1$, and

$$\beta = \sum_{i=1}^{n} \lambda_i a_i \sum_{i=1}^{n} \lambda_i \left(\frac{1}{a_i}\right) = \frac{\sum\limits_{i=1}^{n} \lambda_i \left(\frac{1}{a_i}\right)}{\left(\frac{1}{\sum\limits_{i=1}^{n} \lambda_i a_i}\right)}.$$

The largest value of β is when $\lambda_1 + \lambda_n = 1$, see the illustration in Figure 8. Therefore,

$$\beta \leq \frac{\lambda_1 \frac{1}{a} + \lambda_n \frac{1}{A}}{\frac{1}{\lambda_1 a + \lambda_n A}} = \frac{(\lambda_1 a + \lambda_n A)(\lambda_1 A + \lambda_n a)}{Aa} \leq \frac{(\frac{1}{2}A + \frac{1}{2}a)(\frac{1}{2}a + \frac{1}{2}A)}{Aa} = \frac{(A+a)^2}{4Aa}.$$

q.e.d.

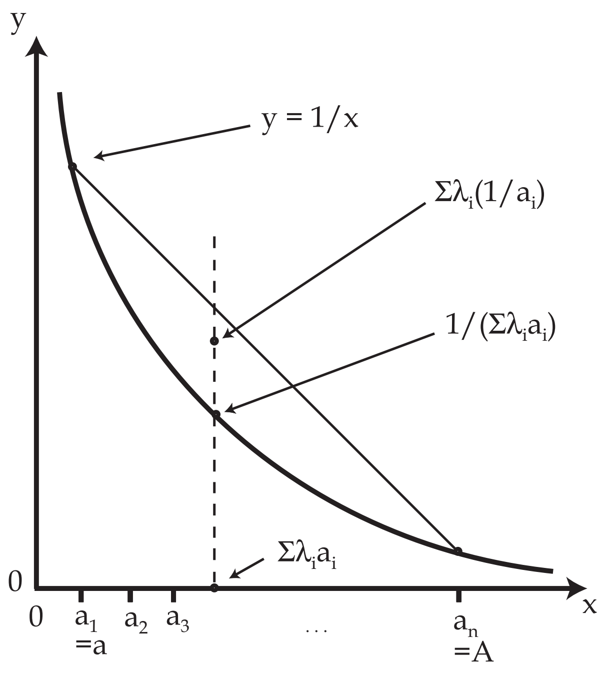

Figure 8: Illustration for proof of the Kantorovich Inequality.

8 Steepest Descent Exercises

NOTE ON COMPUTATION EXERCISES: You may use any machine and any programming language. We recommend, however, that you use MATLAB on Athena or on your own computer. (The teaching assistant is prepared to help you get started in MATLAB on Athena.) Some of the specific details in the computation exercises are purposely left to your own discretion. For example, in cases where you must use a line-search routine, you must choose your own tolerance for the line-search. Also, you must choose your own stopping criteria for your computation. Please include your code or pseudo-code in your write-up.

1. (Steepest Descent) Suppose that x_k and x_{k+1} are two consecutive points generated by the steepest descent algorithm with exact line-search. Show that $\nabla f(x_k)^T \nabla f(x_{k+1}) = 0$.

2. (Steepest Descent) Suppose that we seek to minimize

$$f(x_1, x_2) = 5x_1^2 + 5x_2^2 - x_1 x_2 - 11x_1 + 11x_2 + 11.$$

   (a) Find a point satisfying the first-order necessary conditions for a solution.

   (b) Show that this point is a global minimum of $f(x)$.

   (c) What would be the worst rate of convergence for the steepest descent algorithm for this problem?

   (d) Starting at $(x_1, x_2) = (0, 0)$, at most how many steepest descent iterations would it take to reduce the function value to 10^{-11}?

3. (Steepest Descent) Suppose we seek to minimize

$$f(x) = \frac{1}{2} x^T H x + c^T x + 13,$$

where

$$H = \begin{pmatrix} 10 & -9 \\ -9 & 10 \end{pmatrix} \quad \text{and} \quad c = \begin{pmatrix} 4 \\ -15 \end{pmatrix}.$$

Implement the steepest descent algorithm on this problem, using the following starting points:

- $x_0 = (0, 0)^T$.
- $x_0 = (-0.4, 0)^T$.
- $x_0 = (10, 0)^T$.
- $x_0 = (11, 0)^T$.

As it turns out, the optimal solution to this problem is $x^* = (5, 6)^T$, with $f(x^*) = -22$. What linear convergence constants do you observe for each of the above starting points?

4. (Steepest Descent) Suppose we seek to minimize

$$f(x) = \frac{1}{2}x^T Hx + c^T x,$$

where

$$H = \begin{pmatrix} 10 & -18 & 2 \\ -18 & 40 & -1 \\ 2 & -1 & 3 \end{pmatrix} \quad \text{and} \quad c = \begin{pmatrix} 12 \\ -47 \\ -8 \end{pmatrix}.$$

Implement the steepest descent algorithm on this problem, using the following starting points.

- $x_0 = (0, 0, 0)^T$.
- $x_0 = (15.09, 7.66, -6.56)^T$.
- $x_0 = (11.77, 6.42, -4.28)^T$.
- $x_0 = (4.46, 2.25, 1.85)^T$.

As it turns out, the optimal solution to this problem is $x^* = (4, 3, 1)^T$, with $f(x^*) = -50.5$. What linear convergence constants do you observe for each of the above starting points?

5. (Steepest Descent) Suppose that we seek to minimize the following function:

$$f(x_1, x_2) = -9x_1 - 10x_2 + \theta(-\ln(100 - x_1 - x_2) - \ln(x_1) - \ln(x_2) - \ln(50 - x_1 + x_2)),$$

where θ is a given parameter. Note that the domain of this function is $X = \{(x_1, x_2) \mid x_1 > 0, x_2 > 0, x_1 + x_2 < 100, x_1 - x_2 < 50\}$. Implement the steepest descent algorithm for this problem, using the bisection algorithm for your line-search, with the stopping criterion that $|h^{'}(\tilde{\alpha})| \leq \epsilon = 10^{-6}$. Run your algorithm for $\theta = 10$ and for $\theta = 100$, using the following starting points.

- $x_0 = (8, 90)^T$.
- $x_0 = (1, 40)^T$.
- $x_0 = (15, 68.69)^T$.
- $x_0 = (10, 20)^T$.

What linear convergence constants do you observe for each of the above starting points?

Helpful Hint: Note that $f(x_1, x_2)$ is only defined on the domain $X = \{(x_1, x_2) \mid x_1 > 0, x_2 > 0, x_1 + x_2 < 100, x_1 - x_2 < 50\}$. If you are at a point $\bar{x} = (\bar{x}_1, \bar{x}_2)$ and you have computed a direction $\bar{d} = (\bar{d}_1, \bar{d}_2)$, then note that a value of the upper bound $\hat{\alpha}$ is effectively given by the largest value of α for which the following constraints are satisfied:

$$\bar{x}_1 + \alpha\bar{d}_1 > 0, \ \ \bar{x}_2 + \alpha\bar{d}_2 > 0, \ \ \bar{x}_1 + \alpha\bar{d}_1 + \bar{x}_2 + \alpha\bar{d}_2 < 100, \ \ \bar{x}_1 + \alpha\bar{d}_1 - \bar{x}_2 - \alpha\bar{d}_2 < 50.$$

The largest value of α satisfying these conditions can easily be computed by applying appropriate logic.

Important Note: Please keep the value of θ as a specific command in your code. The reason for this is that later in the semester, we will consider an algorithm where we iteratively change the value of θ. It will then be useful for your code to have a line in it where θ can be set and re-set.