The Steepest Descent Algorithm for Unconstrained Optimization

Robert M. Freund

February, 2014

©2014 Massachusetts Institute of Technology. All rights reserved.
1 Steepest Descent Algorithm

The problem we are interested in solving is:

\[ P : \text{minimize} \quad f(x) \]
\[ \text{s.t.} \quad x \in \mathbb{R}^n, \]

where \( f(x) \) is differentiable. If \( x = \bar{x} \) is a given point, \( f(x) \) can be approximated by its first-order Taylor expansion at \( \bar{x} \):

\[ f(x) \approx f(\bar{x}) + \nabla f(\bar{x})^T (x - \bar{x}). \]

Now let \( d \neq 0 \) be a direction, and in consideration of choosing a new point \( x := \bar{x} + d \), we define:

\[ h(d) := f(\bar{x}) + \nabla f(\bar{x})^T d, \]

whereby we have:

\[ h(d) := f(\bar{x}) + \nabla f(\bar{x})^T d \approx f(\bar{x} + d). \]

If \( d \) is “small,” then \( h(d) \approx f(\bar{x} + d) \). We would like to choose \( d \) so that the normalized inner product \( \nabla f(\bar{x})^T d/\|d\| \) is as negative as possible. The direction

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

makes the most negative (normalized) inner product with the gradient \( \nabla f(\bar{x}) \).

To see this, let \( d \neq 0 \) be any direction. Then:

\[ \nabla f(\bar{x})^T d/\|d\| \geq -\|\nabla f(\bar{x})\| = \nabla f(\bar{x})^T \left( \frac{-\nabla f(\bar{x})}{\|\nabla f(\bar{x})\|} \right) = \nabla f(\bar{x})^T \bar{d}/\|\bar{d}\|, \]

where the first inequality is an application of the Cauchy-Schwartz inequality. For this reason the direction:

\[ \bar{d} = -\nabla f(\bar{x}) \]
is called the direction of steepest descent at the point \( \bar{x} \). It should be noted that the above reasoning depends implicitly on the use of the Euclidean norm to measure the norm of the gradient and the direction, where we define the Euclidean norm to be:

\[
\|v\| := \|v\|_2 := \sqrt{v^T v}.
\]

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 \( \nabla f(\bar{x}) \neq 0 \).

If we compute the search direction \( \bar{d} \) at each iteration as the steepest descent direction, we call the algorithm the steepest descent algorithm, whose formal description is presented in Algorithm 1.

\begin{algorithm}
\caption{Steepest Descent Algorithm}
\begin{algorithmic}
\STATE Initialize at \( x^0 \), and set \( k \leftarrow 0 \).
\STATE At iteration \( k \):
\STATE \hspace{1em} 1. \( d^k := -\nabla f(x^k) \). If \( d^k = 0 \), then stop.
\STATE \hspace{1em} 2. Choose steps-size \( \alpha^k \) (typically by performing an exact or inexact line-search).
\STATE \hspace{1em} 3. Set \( x^{k+1} \leftarrow x^k + \alpha^k d^k \), \( k \leftarrow k + 1 \).
\end{algorithmic}
\end{algorithm}

Regarding the step-size in Step (2.) of Algorithm 1, if the step-size is determined by computing:

\[
\alpha^k := \arg\min_{\alpha} f(x^k + \alpha d^k),
\]

then we say that the step-size has been determined by exact line-search. The exact determination of the optimal value of \( \alpha \) above may not be possible. If the step-size is determined by computing:

\[
\alpha^k \approx \arg\min_{\alpha} f(x^k + \alpha d^k),
\]

then we say that the step-size has been determined by inexact line-search.

There are a variety of metrics that can be used in the approximation criterion for an inexact line-search, of course.
Note from Step (1.) and the fact that $d^k = -\nabla f(x^k)$ is a descent direction, it follows that $f(x^{k+1}) < f(x^k)$ so long as sufficient care is taken in the choice of the step-size $\alpha^k$ in Step (2.).

2 Global Convergence

We have the following convergence result.

**Theorem 2.1 (Convergence of Steepest Descent Algorithm)** Suppose that $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on the set $S = \{ x \in \mathbb{R}^n | f(x) \leq f(x^0) \}$, and that $S$ is a closed and bounded set. Let $\{x^k\}$ be the sequence of iterates of Algorithm 1 using exact line-search. 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 (Theorem 1 of the first lecture note), 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 $\delta := f(\bar{x}) - f(\bar{x} + \bar{\alpha}d) > 0$, where $d = -\nabla f(\bar{x})$. Then also $(\bar{x} + \bar{\alpha}d) \in \text{int}S$.

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

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

However,

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

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

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. Recall 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:
\[
\frac{f(x^{k+1}) - f(x^*)}{f(x^k) - f(x^*)} \leq \delta ,
\]

where \( x^* \) is some optimal value of the problem \( P \). The above statement says that the optimality gap shrinks by at least a multiplicative factor of \( \delta \) 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 \( \delta \) above is called the (linear) 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 a strictly convex 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
\[
0 < a = a_1 \leq a_2 \leq \ldots \leq a_n = A .
\]
Here \( A \) and \( a \) are the largest and smallest eigenvalues of \( Q \), respectively.

Let \( x^* \) be the optimal solution of \( P \). Then
\[
0 = \nabla f(x^*) = Q x^* + q ,
\]
whereby it follows that:
\[
x^* = -Q^{-1} q ,
\]
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, using an exact line-search to determine the step-size. If $\alpha$ is the generic step-size, then

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

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

$$= f(x) - \alpha d^T d + \frac{1}{2}\alpha^2 d^T Qd .$$

Optimizing the value of $\alpha$ in this last expression yields

$$\alpha = \frac{d^T d}{d^T Qd} ,$$

and the next iterate of the algorithm then is

$$x' := x + \alpha d = x + \frac{d^T d}{d^T Qd} d ,$$

and

$$f(x') = f(x + \alpha d) = f(x) - \alpha d^T d + \frac{1}{2}\alpha^2 d^T Qd = f(x) - \frac{1}{2} \left( \frac{d^T d}{d^T Qd} \right)^2 .$$
Therefore,

\[
\frac{f(x') - f(x^*)}{f(x) - f(x^*)} = \frac{f(x) - \frac{1}{2} (d^T d)^2}{f(x) - f(x^*)}
\]

\[
= 1 - \frac{1}{2} \left( x^T Q x + q^T Q^{-1} q \right)
\]

\[
= 1 - \frac{1}{2} \left( (Q x + q)^T Q^{-1} (Q x + q) \right)
\]

\[
= 1 - \frac{(d^T d)^2}{(d^T Q d)(d^T Q^{-1} d)}
\]

\[
= 1 - \beta ,
\]

where

\[
\beta := \frac{(d^T Q d)(d^T Q^{-1} d)}{(d^T d)^2} .
\]

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

**Proposition 3.1 (Kantorovich Inequality)** Let \( A \) and \( a \) be the largest and the smallest eigenvalues of \( Q \), respectively. Then for all \( d \neq 0 \) it holds that:

\[
\beta := \frac{(d^T Q d)(d^T Q^{-1} d)}{(d^T d)^2} \leq \frac{(A + a)^2}{4 A a} .
\]

We will prove this inequality later. For now, let us apply this inequality to the above analysis. Continuing, we have:
Table 1: The Steepest Descent convergence rate $\delta$ as a function of the eigenvalue ratio $A/a$.

\begin{align*}
\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 \\
&= \left( 1 - \frac{2}{\frac{A}{a} + 1} \right)^2 : = \delta .
\end{align*}

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 $\delta$ will be much smaller than 1. However, if $A/a$ is large, then the convergence constant $\delta$ will be only slightly smaller than 1. Table 1 shows some computed values of $\delta$ as a function of the ratio $A/a$. Note that the number of iterations needed to reduce the optimality gap by a factor of 10 grows linearly in the ratio $A/a$. 

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
Eigenvalue Ratio & Upper Bound on & Number of Iterations \\
$A/a$ & $\delta = \left( 1 - \frac{2}{\frac{A}{a} + 1} \right)^2$ & to Reduce the Optimality Gap \\
& & by a factor of 10 \\
\hline
1.1 & 0.0023 & 1 \\
3.0 & 0.25 & 2 \\
10.0 & 0.67 & 6 \\
100.0 & 0.96 & 58 \\
200.0 & 0.98 & 116 \\
400.0 & 0.99 & 231 \\
\hline
\end{tabular}
\end{table}
4 Some Illustrative Examples of the Steepest Descent Algorithm

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 \( f^* := f(x^*) = f(1, 1) = 10.0 \).

In Step (1.) of the steepest descent algorithm, we need to compute

\[
\begin{align*}
d^k &= \left( \begin{array}{c} d^k_1 \\ d^k_2 \end{array} \right) = -\nabla f(x^k_1, x^k_2) = \left( \begin{array}{c} -10x^k_1 - 4x^k_2 + 14 \\ -2x^k_2 - 4x^k_1 + 6 \end{array} \right).
\end{align*}
\]

Let us see how to implement Step (2.) of the steepest descent algorithm with an exact line-search. We need to solve \( \alpha^k = \arg \min_{\alpha \geq 0} h(\alpha) = f(x^k + \alpha d^k) \).

In this example we are be able to derive an analytic expression for \( \alpha^k \). Notice that

\[
\begin{align*}
h(\alpha) &= f(x^k + \alpha d^k) \\
&= 5(x^k_1 + \alpha d^k_1)^2 + (x^k_2 + \alpha d^k_2)^2 + 4(x^k_1 + \alpha d^k_1)(x^k_2 + \alpha d^k_2) - 14(x^k_1 + \alpha d^k_1) - 6(x^k_2 + \alpha d^k_2) + 20,
\end{align*}
\]

and this is a simple quadratic function of the scalar \( \alpha \). It is minimized at

\[
\alpha^k := \frac{(d^k_1)^2 + (d^k_2)^2}{2(5(d^k_1)^2 + (d^k_2)^2 + 4d^k_1 d^k_2)}.
\]

Table 2 shows the iterates of the steepest descent algorithm applied as above to minimize \( f(\cdot) \) initialized at \( x^0 = (x^0_1, x^0_2) = (0, 10) \). The linear convergence of the algorithm can be seen in the numbers in the table. Examining the last column, notice that the optimality gap gains one digit of accuracy in almost every one or two iterations, and achieves six-digit decimal accuracy by iteration 14. Examining the first two columns of the table, we see that the iterates converge to the optimal solution \( x^* = (1, 1) \). Furthermore, starting at iterations 6 and 8 for \( x_1 \) and \( x_2 \), the algorithm adds an additional digit of accuracy roughly every one or two iterations.
In this example, we can write:

\[ f(x) = f(x_1, x_2) = \frac{1}{2} x^T Q x + q^T x + 20 \]

where

\[ Q = \begin{pmatrix} 10 & 4 \\ 4 & 2 \end{pmatrix} \quad \text{and} \quad q = \begin{pmatrix} -14 \\ -6 \end{pmatrix}. \]

The eigenvalues of \( Q \) are easily computed as \( A = 3 + \sqrt{8} \) and \( a = 3 - \sqrt{8} \). Therefore the eigenvalue ratio is \( A/a = 33.9706 \), which is rather large, and the upper bound on the linear convergence rate constant \( \delta \) is:

\[ \delta = \left( 1 - \frac{2}{A/a + 1} \right)^2 = 8/9 = 0.88889. \]

In this case we can guarantee that the optimality gap is reduced by a factor of 10 for every 20 iterations. However, as Table 2 showed, the actual rate of convergence for this example is much faster, achieving an extra digit of accuracy every one or two iterations.

The iterations of the steepest descent algorithm can also be viewed graphically. Let us first recall some geometry for convex quadratic functions and ellipsoids. For a convex quadratic function \( f(x) = \frac{1}{2} x^T Q x + q^T x \), the contours of the function values of \( f(\cdot) \) will be shaped like ellipsoids. This is illustrated in Figure 1. The gradient vector \( \nabla f(x) \) at any point \( x \) will point perpendicular to the contour line passing through \( x \). Therefore the directions of the steps in the steepest-descent algorithm will be perpendicular to the contour of the current iterate. For the current example, Figure 2 plots the iterates from Table 2 along with the directions for each iterate.

**4.2 Example 2: Slow Convergence and “Hem-Stitching”**

Consider the function

\[ f(x) = \frac{1}{2} x^T Q x + q^T x + 10 \]

where \( Q \) and \( q \) are given by:

\[ Q = \begin{pmatrix} 20 & 5 \\ 5 & 2 \end{pmatrix} \]
Table 2: Iterations of the steepest descent algorithm for the example in Subsection 4.1, using exact line-search.
Figure 1: The contours of a convex quadratic function are ellipsoids.

and

\[ q = \begin{pmatrix} -14 \\ -6 \end{pmatrix}. \]

For this example, we have \( x^* = (-4/3, 10/3) \) and \( f^* = f(x^*) = 14/15 = 0.933333 \).

We also have for this example that \( \frac{\Delta}{x} = 30.234 \), and so the linear convergence rate \( \delta \) of the steepest descent algorithm is bounded above by

\[ \delta = \left( 1 - \frac{2}{\sqrt{\Delta x} + 1} \right)^2 = 0.8760. \]

The steepest descent algorithm is guaranteed to reduce the optimality gap by a factor of 10 every 18 iterations, which is a rather slow rate of convergence. Table 3 shows the iterates of the steepest descent algorithm applied to minimize \( f(\cdot) \) initialized at \( x^0 = (40, -100) \).
Figure 2: Iterates and directions of the steepest descent algorithm for the example of Subsection 4.1.

Here we see the linear convergence is indeed slow, requiring over 50 iterations to reduce the optimality gap beyond $10^{-6}$. The values of the iterates of Table 3 are shown graphically in Figure 3. The zig-zag pattern of the iterates is referred to as “hem-stitching,” and typically is observed when $A/a$ is large, as it is in this example.

4.3 Example 3: Rapid Convergence

Consider the function

$$f(x) = \frac{1}{2}x^TQx + q^Tx + 10$$

where $Q$ and $q$ are given by:
Table 3: Iterations of the steepest descent algorithm for the example in Subsection 4.2, using exact line-search.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$x^k$</th>
<th>$x_k^1$</th>
<th>$x_k^2$</th>
<th>$f(x^k) - f^*$</th>
<th>$f(x^k) - f(x^*)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>40.000000</td>
<td>-100.000000</td>
<td>6049.066667</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>25.542693</td>
<td>-99.696700</td>
<td>3980.761795</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>26.277558</td>
<td>-64.668130</td>
<td>2619.654460</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>16.763512</td>
<td>-64.468535</td>
<td>1723.938744</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>17.247111</td>
<td>-41.416980</td>
<td>1134.487330</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10.986120</td>
<td>-41.285630</td>
<td>746.581922</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>11.304366</td>
<td>-26.115894</td>
<td>491.309644</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7.184142</td>
<td>-26.029455</td>
<td>323.320401</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7.393573</td>
<td>-16.046575</td>
<td>212.770262</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>4.882141</td>
<td>-15.989692</td>
<td>140.019573</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.460907</td>
<td>0.948466</td>
<td>2.132883</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>-0.059980</td>
<td>3.038991</td>
<td>0.032490</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-0.124280</td>
<td>3.297005</td>
<td>0.000495</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>-0.132216</td>
<td>3.328850</td>
<td>0.000000</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>-0.133195</td>
<td>3.332780</td>
<td>0.000000</td>
<td>0.658079</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>-0.133316</td>
<td>3.333265</td>
<td>0.000000</td>
<td>0.658025</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-0.133331</td>
<td>3.333325</td>
<td>0.000000</td>
<td>0.654566</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>-0.133333</td>
<td>3.333332</td>
<td>0.000000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Iterations of the steepest descent algorithm for the example in Subsection 4.2, using exact line-search.

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

and

$$q = \begin{pmatrix} -14 \\ -6 \end{pmatrix}.$$  

For this example, we have $x^* = (0.657627, 0.169491)$ and $f^* = f(x^*) = 4.888136$.

We also have for this example that $\frac{A}{a^2} = 1.8541$, and so the linear convergence rate $\delta$ of the steepest descent algorithm is bounded above by $\delta = \left(1 - \frac{2}{\frac{A}{a^2} + 1}\right)^2 = 0.0896$. The steepest descent algorithm is guaranteed to reduce the optimality gap by a factor of 10 every iteration, which is
Figure 3: Iterates and directions of the steepest descent algorithm for the example of Subsection 4.2.

a very rapid rate of convergence. Table 4 shows the iterates of the steepest descent algorithm applied to minimize $f(\cdot)$ initialized at $x^0 = (40, -100)$. Here we see the linear convergence is indeed rapid, requiring fewer than 10 iterations to reduce the optimality gap beyond $10^{-6}$. The values of the iterates of Table 4 are shown graphically in Figure 4.

4.4 Example 4: A non-quadratic function

Let

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

In this example, $x^* = (0.5, 2.5, 0.2, 0.8)$ and $f^* = f(x^*) = 1.609438$. Table 5 shows the iterates of the steepest descent algorithm applied to minimize $f(\cdot)$ initialized at $x^0 = (1, 1, 1, 1)$, using the bisection line-search method, which is an inexact line-search.

4.5 Example 5: An analytic example

Suppose that

$$f(x) = \frac{1}{2} x^T Q x + q^T x$$
Table 4: Iterations of the steepest descent algorithm for the example in Subsection 4.3, using exact line-search.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$x^k$</th>
<th>$f(x^k) - f^*$</th>
<th>$\frac{f(x^k)-f(x^<em>)}{f(x^</em>)} - f(x^*)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>$x_1^k$</td>
<td>$x_2^k$</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>40.000000</td>
<td>-100.000000</td>
<td>76045.111864</td>
</tr>
<tr>
<td>1</td>
<td>19.867118</td>
<td>-1.025060</td>
<td>3586.727191</td>
</tr>
<tr>
<td>2</td>
<td>2.513241</td>
<td>-4.555081</td>
<td>169.170794</td>
</tr>
<tr>
<td>3</td>
<td>1.563658</td>
<td>0.113150</td>
<td>7.970972</td>
</tr>
<tr>
<td>4</td>
<td>0.745149</td>
<td>-0.053347</td>
<td>0.376339</td>
</tr>
<tr>
<td>5</td>
<td>0.700361</td>
<td>0.166834</td>
<td>0.017750</td>
</tr>
<tr>
<td>6</td>
<td>0.661755</td>
<td>0.158881</td>
<td>0.000837</td>
</tr>
<tr>
<td>7</td>
<td>0.659643</td>
<td>0.169366</td>
<td>0.000039</td>
</tr>
<tr>
<td>8</td>
<td>0.657822</td>
<td>0.168996</td>
<td>0.000001</td>
</tr>
<tr>
<td>9</td>
<td>0.657722</td>
<td>0.169486</td>
<td>0.000000</td>
</tr>
<tr>
<td>10</td>
<td>0.657636</td>
<td>0.169468</td>
<td>0.000000</td>
</tr>
<tr>
<td>11</td>
<td>0.657632</td>
<td>0.169491</td>
<td>0.000000</td>
</tr>
<tr>
<td>12</td>
<td>0.657628</td>
<td>0.169490</td>
<td>0.000000</td>
</tr>
<tr>
<td>13</td>
<td>0.657627</td>
<td>0.169492</td>
<td>0.000000</td>
</tr>
<tr>
<td>14</td>
<td>0.657627</td>
<td>0.169491</td>
<td>0.000000</td>
</tr>
</tbody>
</table>

Then

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

and so

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

and

$$f(x^*) = -1.0.$$  

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

$$\delta = \left( 1 - \frac{2}{A + 1} \right)^2 \leq 0.556.$$
<table>
<thead>
<tr>
<th>Iteration $k$</th>
<th>$x^k$</th>
<th>$f(x^k) - f^*$</th>
<th>$(f(x^k) - f(x^<em>) - f(x^</em>)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x^k_1$</td>
<td>$x^k_2$</td>
<td>$x^k_3$</td>
</tr>
<tr>
<td>0</td>
<td>1.000000</td>
<td>1.000000</td>
<td>1.000000</td>
</tr>
<tr>
<td>1</td>
<td>0.802973</td>
<td>1.118216</td>
<td>0.211893</td>
</tr>
<tr>
<td>2</td>
<td>0.634501</td>
<td>1.710704</td>
<td>0.332224</td>
</tr>
<tr>
<td>3</td>
<td>0.616707</td>
<td>1.735137</td>
<td>0.205759</td>
</tr>
<tr>
<td>4</td>
<td>0.545466</td>
<td>1.972851</td>
<td>0.267358</td>
</tr>
<tr>
<td>5</td>
<td>0.543856</td>
<td>1.986648</td>
<td>0.204195</td>
</tr>
<tr>
<td>6</td>
<td>0.553037</td>
<td>2.121519</td>
<td>0.241186</td>
</tr>
<tr>
<td>7</td>
<td>0.547600</td>
<td>2.129461</td>
<td>0.202091</td>
</tr>
<tr>
<td>8</td>
<td>0.526273</td>
<td>2.205845</td>
<td>0.229776</td>
</tr>
<tr>
<td>9</td>
<td>0.525822</td>
<td>2.212935</td>
<td>0.202342</td>
</tr>
<tr>
<td>10</td>
<td>0.511244</td>
<td>2.406747</td>
<td>0.200739</td>
</tr>
<tr>
<td>11</td>
<td>0.503892</td>
<td>2.468138</td>
<td>0.200271</td>
</tr>
<tr>
<td>12</td>
<td>0.501351</td>
<td>2.489042</td>
<td>0.200095</td>
</tr>
<tr>
<td>13</td>
<td>0.500467</td>
<td>2.496222</td>
<td>0.200033</td>
</tr>
<tr>
<td>14</td>
<td>0.500161</td>
<td>2.498966</td>
<td>0.200011</td>
</tr>
<tr>
<td>15</td>
<td>0.500056</td>
<td>2.499550</td>
<td>0.200004</td>
</tr>
<tr>
<td>16</td>
<td>0.500019</td>
<td>2.499845</td>
<td>0.200001</td>
</tr>
<tr>
<td>17</td>
<td>0.500007</td>
<td>2.499946</td>
<td>0.200000</td>
</tr>
<tr>
<td>18</td>
<td>0.500002</td>
<td>2.499981</td>
<td>0.200000</td>
</tr>
<tr>
<td>19</td>
<td>0.500001</td>
<td>2.499993</td>
<td>0.200000</td>
</tr>
<tr>
<td>20</td>
<td>0.500000</td>
<td>2.499998</td>
<td>0.200000</td>
</tr>
<tr>
<td>21</td>
<td>0.500000</td>
<td>2.499998</td>
<td>0.200000</td>
</tr>
<tr>
<td>22</td>
<td>0.500000</td>
<td>2.499998</td>
<td>0.200000</td>
</tr>
</tbody>
</table>

Table 5: Iterations of the steepest descent algorithm for the example in Subsection 4.4, using the bisection line-search method, which is an inexact line-search.
Figure 4: Iterates and directions of the steepest descent algorithm for the example of Subsection 4.3.

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

$x^1 = (-0.4, 0.4)$, \hspace{1cm} $x^2 = (0, 0.8)$

and the even numbered iterates satisfy

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

and so

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

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

5 Remarks

Note that the proof of convergence of the steepest descent algorithm is quite basic; indeed 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 $f(x)$. However, it also relies on the method using an exact line-search. It is possible to prove convergence for a variety of inexact line-search methods, but the proofs then become more involved.
The analysis of the rate of convergence of the steepest descent method is originally due to Kantorovich, and is quite elegant. Kantorovich made fundamental contributions not only to optimization, but also to economics, and was a recipient of the Nobel Prize in Economic Science in 1975 (along with Tjalling Koopmans) for their optimization contributions to economic theory.

The bound on the rate of convergence of the steepest descent algorithm derived herein is conservative in that it is the \textit{worst-case} rate of convergence. However, this rate is quite often attained in practice, which indicates that such worst-case behavior winds up being typical behavior in the case of the steepest descent algorithm.

The ratio $A/a$ of the largest to the smallest eigenvalue of a matrix is called the \textit{condition number} of the matrix. The condition number is a fundamental concept in the theory of matrices and computation involving matrices.

The proof of linear convergence presented herein assumes that $f(\cdot)$ is a convex quadratic function. It turns out that most functions behave as near-quadratic convex functions in a neighborhood of a strictly local optimal solution. The analysis of the non-quadratic case gets very involved; fortunately, the key intuition is obtained by analyzing the quadratic case as we have done herein.

\section{Proof of the Kantorovich Inequality}

Here we prove Proposition 3.1, which is called the \textbf{Kantorovich Inequality}. Let $A$ and $a$ be the largest and the smallest eigenvalues of $Q$, respectively. Then Proposition 3.1 asserts that for any $d \neq 0$, it holds that

$$
\beta := \frac{d^T Q d \cdot d^T Q^{-1} d}{d^T d \cdot d^T d} \leq \frac{(A + a)^2}{4 A a}.
$$

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

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

19
and,

\[
D = \begin{pmatrix}
a_1 & 0 & \cdots & 0 \\
0 & a_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & a_n
\end{pmatrix}.
\]

Then

\[
\beta = \frac{d^T Q d \cdot d^T Q^{-1} d}{d^T d \cdot d^T d} = \frac{(d^T R D R^T d)(d^T R D^{-1} R^T d)}{(d^T R R^T d)(d^T R R^T d)} = \frac{f^T D f f^T D^{-1} f}{f^T f f^T f}
\]

where \(f = R^T d\). Let \(\lambda_i = \frac{f_i^2}{f^T f}\). Then \(\lambda_i \geq 0\) and \(\sum_{i=1}^{n} \lambda_i = 1\), and

\[
\beta = \left( \sum_{i=1}^{n} \lambda_i a_i \right) \left( \sum_{i=1}^{n} \lambda_i \left( \frac{1}{a_i} \right) \right).
\]

Let us examine this last expression, which has a left part and a right part. The left part is a convex combination of the \(a_i\) values. Let \(\bar{a} = \sum_{i=1}^{n} \lambda_i a_i\). For a given value of \(\bar{a}\), the value of the right part that yields the largest value of \(\beta\) occurs by composing \(\bar{a}\) using only the values of \(a_1 = A\) and \(a_n = A\) in the convex combination, that is, by having \(\lambda_1 + \lambda_n = 1\) and \(\lambda_2 = \cdots = \lambda_{n-1} = 0\). This is illustrated in Figure 5. Therefore:

\[
\beta \leq (\lambda_1 a + \lambda_n A) \left( \lambda_1 \frac{1}{A} + \lambda_n \frac{1}{A} \right)
\]

\[
= \frac{(\lambda_1 a + \lambda_n A)(\lambda_1 A + \lambda_n a)}{Aa}
\]

\[
\leq \frac{(\frac{1}{2} A + \frac{1}{2} a)(\frac{1}{2} a + \frac{1}{2} A)}{Aa}
\]

\[
= \frac{(A + a)^2}{4Aa},
\]

20
where the last inequality follows since \( \lambda_1 = \lambda_n = \frac{1}{2} \) maximizes the preceding expression over all values of \( \lambda_1, \lambda_n \) that satisfy \( \lambda_1 + \lambda_n = 1 \).

Figure 5: Illustration for the proof of the Kantorovich Inequality.

7 Steepest Descent Exercises

1. 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. Suppose that we seek to minimize

\[
 f(x_1, x_2) = 5x_1^2 + 5x_2^2 - x_1x_2 - 11x_1 + 11x_2 + 11.
\]
(a) Find a point satisfying the first-order necessary conditions for a solution.

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

(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. 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:

- $x^0 = (0, 0)$
- $x^0 = (-0.4, 0)$
- $x^0 = (10, 0)$
- $x^0 = (11, 0)$

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. Suppose we seek to minimize

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

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:
• $x^0 = (0, 0, 0)$
• $x^0 = (15.09, 7.66, -6.56)$
• $x^0 = (11.77, 6.42, -4.28)$
• $x^0 = (4.46, 2.25, 1.85)$

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. 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 $\theta$ 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 $|h'(\bar{\alpha})| \leq \varepsilon = 10^{-6}$. Run your algorithm for $\theta = 10$ and for $\theta = 100$, using the following starting points:

• $x^0 = (8, 90)$
• $x^0 = (1, 40)$
• $x^0 = (15, 68.69)$
• $x^0 = (10, 20)$

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 $\alpha$ for which the following constraints are satisfied:

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

The largest value of $\alpha$ satisfying these conditions can easily be computed by applying appropriate logic.
6. Suppose that \( f(x) \) is a strictly convex twice-continuously differentiable function whose Hessian \( H(x) \) is nonsingular for all \( x \), and consider Newton’s method with a line-search. Given \( \bar{x} \), we compute the Newton direction \( d = -[H(\bar{x})]^{-1}\nabla f(\bar{x}) \) and the next iterate \( \tilde{x} \) is chosen to satisfy:

\[
\tilde{x} := \arg \min_{\alpha} f(\bar{x} + \alpha d).
\]

Suppose that Newton’s method with a line-search is started at \( x^0 \), and that \( \{ x : f(x) \leq f(x^0) \} \) is a closed and bounded set. Prove that the iterates of this method converge to the unique global minimum of \( f(x) \). (\textbf{Hint:} look at the proof of convergence of steepest descent and modify this proof as appropriate.)