9. Minimization problems

We consider two classes of problems: the first is the unconstrained minimization problem (UCMP) and the second is nonlinear least squares (NLLS).

UCMP: Given \( f : \mathbb{R}^N \to \mathbb{R} \), find \( x \in \mathbb{R}^N \) which minimizes \( f(x) \).

NLLS: Given \( F = (f_1, \ldots, f_m)^T : \mathbb{R}^N \to \mathbb{R}^m \), with \( m \geq N \), find \( x \in \mathbb{R}^N \) which minimizes \( \phi(x) = (1/2) \sum_{k=1}^m [f_k(x)]^2 \).

Note that the second problem is a special case of the first, but with more structure.

9.1. Newton’s method and steepest descent. To solve UCMP, we can look for \( x^* \) at which

\[
\nabla f = (\partial f/\partial x_1, \ldots, \partial f/\partial x_N) = 0
\]

and the Hessian matrix \( H_f = (\partial^2 f/\partial x_i \partial x_j) \) is symmetric and positive definite. Thus the problem becomes one of solving a nonlinear systems of equations \( F(x) = 0 \), where \( F_i = \partial f/\partial x_i \).

If we apply Newton’s method, we get the iteration

\[
x^{n+1} = x^n - J_{\nabla f}(x^n)^{-1} \nabla f(x^n) = x^n - H_f(x^n)^{-1} \nabla f(x^n),
\]

since \( (J_F)_{ij} = \partial F_i/\partial x_j \) and hence

\[
(J_{\nabla f})_{ij} = \frac{\partial}{\partial x_j} \frac{\partial f}{\partial x_i} = \frac{\partial^2 f}{\partial x_i \partial x_j} = H_f.
\]

Note that \( H_f \) is symmetric, so for a minimum at \( x^* \), it will be sufficient to have \( H_f(x^*) \) to be positive definite.

In the special case of UCMP where \( f = (1/2)x^T A x - x^T b \), we considered methods of the form \( x^{n+1} = x^n + \alpha_n p^n \), where \( p^n \) is a search direction and \( \alpha_n \) is a scalar. In the method of steepest descent, we choose \( p^n = -\nabla f(x^n) \). With this choice and the special form of \( f \), we can solve explicitly for the best choice of \( \alpha_n \). In the more general case, we would consider methods of the form

\[
x^{n+1} = x^n - \alpha_n \nabla f(x^n),
\]

where \( \alpha_n \) is chosen to guarantee that \( f(x^{n+1}) < f(x^n) \). The advantage of this approach is that we do not need to compute the Hessian. However, the method converges slowly. To compromise, we could consider methods of the form

\[
x^{n+1} = x^n - \alpha_n B_n^{-1} \nabla f(x^n),
\]

where \( B_n \) is symmetric and positive definite and \( \alpha_n \) is chosen to insure that \( f(x^{n+1}) < f(x^n) \). For example, let \( B_n = (H_f(x^n) + \mu_n I) \), with \( \mu_n > 0 \) chosen so that \( B_n \) is positive definite. For large values of \( \mu_n \) this method behaves like steepest descent and for small values of \( \mu_n \), it behaves like Newton’s method. The difficult part is deciding how to choose the parameter \( \mu_n \).
9.2. Quasi-Newton methods. Analogous to the case of quasi-Newton methods for nonlinear equations, we now wish to avoid computation of the Hessian at each iteration. In this case, we want to generate a sequence of symmetric, positive definite matrices $B_n$ such that $B_n$ approximates $H_f(x^n)$, but can be computed easily from $B_{n-1}$. Since we are now solving the system $\nabla f(x) = 0$, the appropriate Taylor series expansion is:

$$\nabla f(x^n) = \nabla f(x^{n+1}) + H_f(x^{n+1})(x^n - x^{n+1}) + O(x^n - x^{n+1})^2.$$ 

Thus, we want the approximation $B_{n+1}$ to $H_f(x^{n+1})$ to satisfy the quasi-Newton equation

$$\nabla f(x^n) = \nabla f(x^{n+1}) + B_{n+1}(x^n - x^{n+1}).$$

To simplify notation, let

$$y^n = \nabla f(x^{n+1}) - \nabla f(x^n), \quad s^n = x^{n+1} - x^n,$$

so the quasi-Newton equation is $B_{n+1} s^n = y^n$. If we look for a symmetric, single rank (the maximum number of linearly independent rows is one) update satisfying the quasi-Newton equation, then provided $(y^n - B_n s^n)^T s^n \neq 0$, the only one is given by

$$B_{n+1} = B_n + \frac{(y^n - B_n s^n)(y^n - B_n s^n)^T}{(y^n - B_n s^n)^T s^n}.$$

It turns out that this method does not work well, so we look for a double rank update. It can be shown that the general symmetric rank 2 update is given by:

$$B_{n+1} = B_n + \frac{(y^n - B_n s^n)(c^n)^T + c^n(y^n - B_n s^n)^T}{(c^n)^T s^n} - \frac{(y^n - B_n s^n)^T s^n c^n (c^n)^T}{[(c^n)^T s^n]^2},$$

where $c$ is an arbitrary vector such that $(c^n)^T s^n \neq 0$.

If we choose $c^n = s^n$, we get the Powell symmetric Broyden update. Since we would also like to have the property that $B_n$ positive definite implies that $B_{n+1}$ is positive definite, a better choice is $c^n = y^n$ (called the Davidon-Fletcher-Powell method).

Finally, as in the case of nonlinear equations, instead of updating $B_n$ and then having to solve a linear system of equations at each step, we can update the inverse directly. If we let $H_n = B_n^{-1}$, we then get the iteration $x^{n+1} = x^n - \alpha_n H_n \nabla f(x^n)$, where

$$H_{n+1} = H_n + \frac{(s^n - H_n y^n)(s^n)^T + s^n(s^n - H_n y^n)^T}{(s^n)^T y^n} - \frac{(s^n - H_n y^n)^T y^n s^n (s^n)^T}{[(s^n)^T y^n]^2}.$$

This method in which the inverse is updated directly is known as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method.

9.3. Nonlinear least squares. Finally, we consider the nonlinear least squares problem of minimizing $\phi(x) = (1/2) \sum_{k=1}^m [f_k(x)]^2$. Let $F = [f_1, \ldots, f_m]^T$. Then $J_F$, the Jacobian matrix of partial derivatives of $F$ is given by $[J_F(x)]_{kj} = (\partial f_k/\partial x_j)$.

Now

$$\frac{\partial \phi}{\partial x_j} = \sum_{k=1}^m f_k(x) \frac{\partial f_k}{\partial x_j} = \{[J_F(x)]^T F(x)\}_{j}, \quad (H_\phi)_{ij} = \frac{\partial^2 \phi}{\partial x_i \partial x_j} = \sum_{k=1}^m \left[ f_k(x) \frac{\partial^2 f_k}{\partial x_i \partial x_j} + \frac{\partial f_k}{\partial x_i} \frac{\partial f_k}{\partial x_j} \right].$$
Hence, $\nabla \phi(x) = J_F(x)^T F(x)$ and

$$H_\phi(x) = J_F(x)^T J_F + \sum_{k=1}^{m} f_k(x) H_{f_k}(x).$$

Newton’s method for the system $\nabla \phi(x) = J_F(x)^T F(x) = 0$, is then given by

$$x^{n+1} = x^n - [H_\phi(x^n)]^{-1} \nabla \phi(x^n).$$

Since this is fairly complicated to compute, we seek a simpler method. One approach is to linearize $f_k(x)$ about $x^n$ and minimize the resulting quadratic functional instead. Using Taylor series, we approximate

$$f_k(x) \approx f_k(x^n) + [\nabla f_k(x^n)]^T (x - x^n),$$

since we expect the remainder to be small if $x - x^n$ is small. Inserting this approximation,

$$\phi(x) \approx (1/2) \sum_{k=1}^{m} [f_k(x^n)]^2 + \sum_{k=1}^{m} f_k(x^n) [\nabla f_k(x^n)]^T (x - x^n) + (1/2) \sum_{k=1}^{m} ([\nabla f_k(x^n)]^T (x - x^n))^2.$$

Then

$$\nabla \phi(x) \approx \sum_{k=1}^{m} f_k(x^n) \nabla f_k(x^n) + \sum_{k=1}^{m} [\nabla f_k(x^n)]^T (x - x^n).$$

Since we want to find $x$ such that $\nabla \phi(x) = 0$, we choose $x^{n+1}$ to satisfy

$$\sum_{k=1}^{m} \nabla f_k(x^n) [\nabla f_k(x^n)]^T (x^{n+1} - x^n) = - \sum_{k=1}^{m} f_k(x^n) \nabla f_k(x^n).$$

This may be written in the form

$$J_F^T(x^n) J_F(x^n) (x^{n+1} - x^n) = -J_F^T(x^n) F(x^n).$$

Hence, this approximation amounts to dropping the term $\sum_{k=1}^{m} f_k(x) H_{f_k}(x)$ in Newton’s method. We would expect this to be small if the minimum of $\phi$ is near zero. This method is known as the Gauss-Newton method.

In practice, a modified version of Gauss-Newton, known as the Levenberg-Marquardt method, is used. This method is given by the iteration

$$[\alpha_n I + J_F^T(x^n) J_F(x^n)](x^{n+1} - x^n) = -J_F^T(x^n) F(x^n),$$

where $\alpha_n \geq 0$ is an appropriately chosen scalar. The idea is that for $\alpha_n$ large, the method behaves like steepest descent (to ensure that the function $\phi(x)$ is being decreased), while for $\alpha_n$ small, it behaves more like Newton’s method, which will converge faster. In general, we would increase $\alpha_n$ if we take a step that increases $\phi(x)$, and decrease $\alpha_n$ if $\phi(x)$ is decreasing.