Newton’s method can be extended to multivariate functions in order to compute much better search directions than gradient descent. It attempts to find a point at which the function gradient is zero using a quadratic approximation of the function. Like in the univariate case, Newton’s method achieves quadratic order of convergence. But, its main drawback is that it requires computing and inverting the Hessian matrix, the matrix of second derivatives of $f$.

Quasi-Newton methods avoid this expense using a procedure analogous to the secant method, in which the Hessian matrix is approximated using differencing of gradient evaluations. Moreover some the expense of matrix inversion can be reduced using the Sherman-Morrison-Woodbury update formula. As a result, quasi-Newton methods are very popular for medium-scale (hundreds or thousands of variables) local optimization.

1 Hessian Matrix

The Hessian matrix of a scalar field $f$ at a point $x = (x_1, \ldots, x_n)$, denoted $\nabla^2 f(x)$ or $H(x)$ when we know which function we are talking about, is the matrix of second partial derivatives:

$$H(x) = \nabla^2 f(x) = \begin{bmatrix}
\frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\
\frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x)
\end{bmatrix}.$$  \hspace{1cm} (1)
The significance of this matrix is easiest to visualize in the 2D case:

\[
H(x) = \nabla^2 f(x) = \begin{bmatrix}
\frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) \\
\frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_2^2}(x)
\end{bmatrix}.
\] (2)

The diagonal elements tell you the second derivative of \(f\) as you move along each axis. The off-diagonal elements tell you how the partial derivative of \(f\) in the \(x_1\) direction changes as you move in the \(x_2\) direction. It may help to imagine the partial derivative of \(f\) in the \(x_1\) direction as a new scalar field \(g(x)\). Then, you take the partial derivative of \(g(x)\) in the \(x_2\) direction.

**Use of Hessian in second-order Taylor expansion.** The Hessian appears in the second-order Taylor expansion of a scalar field:

\[
f(x) = f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{1}{2}(x - x_0)^T \nabla^2 f(x_0)(x - x_0) + O(||x - x_0||^3).
\] (3)

Taking the first three terms, we see an expression of the form \(\frac{1}{2}x^TH(x_0)x + b^Tx + c\) for a vector \(b\) and scalar \(c\). This is known as a quadratic form because each element of \(x\) is raised to a maximum power of two. We see that it is the unique quadratic form that passes through \(x_0\) and has the same first and second partial derivatives as \(f\) at \(x_0\).

**Use of Hessian as a second derivative test.** Note that the second-order Taylor approximation at a critical point \(x_0\) (one where \(\nabla f(x_0) = 0\)) is 

\[
f(x) \approx \frac{1}{2}(x - x_0)^T H(x_0)(x - x_0) + f(x_0).
\]

So, to a second-order approximation the Hessian captures all the relevant information about the shape of \(f\) in a local neighborhood. Let us now look at the possible shapes captured by \(H\).

It turns out that multiple differentiation is invariant to changes in order. That is, \(\frac{\partial^2 f}{\partial x_2 \partial x_1} = \frac{\partial^2 f}{\partial x_1 \partial x_2}\). So, **Hessian matrices are symmetric**. Because symmetric matrices have a full set of real eigenvalues and an orthogonal basis of eigenvectors, we can perform the eigendecomposition of \(H = QAQ^T\) where \(Q\) is orthogonal and \(\Lambda\) is diagonal. This is useful to understand how \(H\) affects the shape. If we perform a change of variables \(y(x)\) by performing the following translation and rotation: \(y(x) = Q^T(x - x_0)\), then we see that the quadratic form is simply \(f(y) - f(x_0) = \frac{1}{2}y^T\Lambda y\). Since \(\Lambda\) is diagonal, we have a sum of axis-aligned parabolas: \(f(y) - f(x_0) = \frac{1}{2} (\lambda_1 y_1^2 + \cdots + \lambda_n y_n^2)\). We therefore need to look at the signs of the eigenvalues \(\lambda_1, \ldots, \lambda_n\) to determine the shape of the function.
Local minima. If all \( \lambda_i \) are positive and equal \( \lambda_1 = \cdots = \lambda_n \), then \( f(y) \) is an isotropic bowl shape pointing upward. If they are positive but different, then \( f(y) \) is skewed and stretched along the axes, but the bowl still points upward. In either case, \( x_0 \) is a local minimum. Because a matrix is positive definite if and only if all of its eigenvalues are positive, \( x_0 \) is a local minimum if the gradient is zero and \( H(x_0) \) is (strictly) positive definite.

If \( H(x_0) \) is positive semi-definite with some zero eigenvalues, then the shape of \( f(y) \) is an upward facing “channel” and \( x_0 \) may not be a local minimum, or it may not be a unique minimum.

Local maxima. If all \( \lambda_i \) are negative, the bowl points downward, and we have a local maximum. Hence, \( x_0 \) is a local maximum if the gradient is zero and \( H(x_0) \) is (strictly) negative definite. If \( H(x_0) \) is negative semi-definite with some zero eigenvalues, then the shape of \( f(y) \) is a downward-facing “ridge” and \( x_0 \) may not be a local maximum, or it may not be a unique maximum.

Saddle points. If some \( \lambda_i \) are negative and some are positive, then \( x_0 \) is a saddle point where some directions cause an increase in \( f \) and others decrease it. If this is the case, then \( H(x_0) \) is an indefinite matrix.

2 Multivariate Newton’s method

The multivariate Newton’s method for minimization is very similar to Newton’s method for root finding. Recall that when Newton’s method is used for minimization of the function \( f \) you try to find the root of the function \( g = f' \). Expanding the Newton iteration \( x_{t+1} = x_t - g(x_t)/g'(x_t) \) in terms of \( f \) we have

\[
x_{t+1} = x_t - f'(x_t)/f''(x_t).
\]

(4)

For functions of many variables the iteration is very similar, except that the gradient and Hessian take the place of \( f' \) and \( f'' \), respectively. Also, we must take heed of the important differences between matrix algebra and scalar algebra.
2.1 Derivation from Taylor Expansion

Take the second-order Taylor expansion around the point $x_t$:

$$f(x) \approx f(x_t) + \nabla f(x_t) \cdot (x - x_t) + \frac{1}{2}(x - x_t)^T H(x_t)(x - x_t).$$  \hspace{1cm} (5)

We would like to find a point at which the quadratic form on the right hand side is minimized. Supposing that $H(x_t)$ is positive definite, the unique minimum is obtained at $x = x_t - H(x_t)^{-1}\nabla f(x_t)$. To see this, the gradient of a quadratic form $q(x) = \frac{1}{2}x^T Hx + b^T x + c$ is

$$\nabla q(x) = Hx + b$$  \hspace{1cm} (6)

which attains the value of 0 at $x = -H^{-1}b$ as long as $H$ is invertible.

In practice, the step size is usually combined with a line search to prevent divergence. So, Newton’s method can be summed up as:

$$x_{t+1} = x_t - \alpha_t H(x_t)^{-1}\nabla f(x_t).$$  \hspace{1cm} (7)

Aside: Derivatives of matrix equations. It is not immediately clear why the gradient of $q(x)$ led to such a compact expression. The most foolproof way of performing such gradients is to perform elementwise partial derivatives (using chain rules, multiplication rules, etc. as necessary) and assemble them into a gradient vector. Here are some useful identities:

- If $f(x) = b^T x = x^T b$ then $\nabla f(x) = b$.
- If $f(x) = x^T Ax$ then $\nabla f(x) = A^T x + Ax$.
- If $f(x) = g(x)^T h(x)$ with $g, h : \mathbb{R}^n \mapsto \mathbb{R}^m$, then

$$\nabla f(x) = \begin{bmatrix} h(x)^T \partial g(x)/\partial x_1 \\ \vdots \\ h(x)^T \partial g(x)/\partial x_n \end{bmatrix} + \begin{bmatrix} g(x)^T \partial h(x)/\partial x_1 \\ \vdots \\ g(x)^T \partial h(x)/\partial x_n \end{bmatrix}$$  \hspace{1cm} (8)

So, in the derivation of the expression $\nabla q(x) = Hx + b$ we applied identities 1 and 2, and used the fact that the Hessian is symmetric.
2.2 Practical considerations

To compute the step direction \( d_t = -H(x_t)^{-1} \nabla f(x_t) \) it is more expensive to invert \( H \) entirely than it is to solve the linear equation \( H(x_t)d_t = -\nabla f(x_t) \). For medium sized problems, Cholesky factorization is practical, while for very large problems, conjugate gradient techniques can be used to compute the step approximately. Another advantage to Cholesky factorization is that you can test whether \( H \) is positive definite. If not, then the search can switch to an alternative direction, e.g., the negated gradient.

Another problem is that the computation of the inverse becomes unstable when \( H(x_t) \) is non-invertible, or is very close to being non-invertible. As a result it is often preferrable to add an adjustment of the form \( H(x_t) + \mu I_n \) where \( \mu \) is some parameter to ensure that the resulting matrix is positive definite. If \( \mu \) is large enough, then the contribution of the Hessian becomes diminished, and the search direction approaches the gradient.

3 Quasi-Newton Methods

Although Newton’s method converges quadratically near the root, it requires inverting a Hessian, which is \( O(n^3) \) with standard techniques. As a result each iteration can be quite expensive for large \( n \). It also requires evaluating a possibly complex and unwieldy Hessian expression involving second partial derivatives. If second derivatives are not available, then approximate differ-entiation techniques can be used. However, these can be computationally expensive because they require \( O(n^2) \) function evaluations.

Quasi-Newton methods overcome many of these limitations. They re-semble the secant method in that the Hessian is approximated using finite differences of the function gradient. We will also introduce a useful trick to avoid computing the Hessian inverse. They achieve sub-quadratic but super-linear rates of convergence, and as a result they are employed more often in practice than Newton’s method.

3.1 Derivation

Consider the secant approximation of the second derivative of the univariate function \( f(x) \):

\[
f''(x_t) \approx \frac{f'(x_t) - f'(x_{t-1})}{x_t - x_{t-1}},
\]

(9)
or in other words,
\[ f''(x_t)(x_t - x_{t-1}) \approx f'(x_t) - f'(x_{t-1}). \]  
(10)

A generalization of (10) to a multivariate function is as follows:
\[ \nabla^2 f(x_t)(x_t - x_{t-1}) \approx \nabla f(x_t) - \nabla f(x_{t-1}) \]  
(11)

The idea is to find a matrix \( H_t \approx \nabla^2 (x_t) \) that makes this an equality. But there are \( n^2 \) entries in \( H_t \) and only \( n \) constraints, this is an underdetermined system. The question is now which matrix to choose from?

Quasi-Newton methods begin with some estimate of the Hessian \( H_0 \) and incrementally improve it over time: \( H_1, H_2, \ldots \). Each \( H_t \) is derived from \( H_{t-1} \) such that equality in (11) is fulfilled, but also such that \( H_t \) differs from \( H_{t-1} \) as little as possible (in some way, to be defined in a moment). We will also maintain that \( H_0 \) and all subsequent \( H_t \) are positive definite.

**DFP Method.** The Davidson-Fletcher-Powell (DFP) update uses the following formula:
\[
H_t = (I_n - \frac{1}{q_l^T \delta x_t} q_l \delta x_t^T) H_{t-1} (I_n - \frac{1}{q_l^T \delta x_t} \delta x_t q_l^T) + \frac{q_l q_l^T}{q_l^T \delta x_t} \]  
(12)

where \( q_t = \nabla f(x_t) - \nabla f(x_{t-1}) \) and \( \delta x_t = x_t - x_{t-1} \). Verify to yourself that (11) is fulfilled. It turns out with a rather lengthy derivation that this matrix is the closest symmetric positive definite matrix to \( H_{t-1} \) that satisfies (11) in the sense of the Frobenius norm, which is the square root of the sum of squared element-wise differences.

4 **Sheneman-Morrison-Woodbury Formula**

The SMW (or Woodbury) formula gives an analytical expression for the inverse of \( B = A + uv^T \) where \( A \) is a square matrix and \( u \) and \( v \) are vectors:
\[
B^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1 + u^TA^{-1}v} \]  
(13)

(you may verify that this is true via inspection).

This formula is significant because if we have already stored \( A^{-1} \), then we can compute \( B^{-1} \) without an \( O(n^3) \) matrix inversion. In fact this update only takes \( O(n^2) \) time. This formula is quite useful in many applications.
Efficient DFP update. If we let $B_t = H_t^{-1}$, it turns out that through repeatedly application of the Woodbury formula we can obtain an expression of $B_t$ in terms of $B_{t-1}$:

\[
B_t = B_{t-1} - \frac{B_{t-1}q_t q_t^T B_{t-1}}{q_t^T B_{t-1} q_t} + \frac{\Delta x_t \Delta x_t^T}{q_t^T \Delta x_t} \tag{14}
\]

Verify to yourself that (11) is satisfied; that is,

\[
\Delta x_t = B_t q_t. \tag{15}
\]

This expression is significant because by storing approximations $B_0, B_1, B_2, \ldots$ to the Hessian inverse rather than the Hessian, the search direction for each Newton iteration $-B_t \nabla f(x_t)$ can be computed in $O(n^2)$ time.

BFGS Method. The Broyden, Fletcher, Goldfarb, and Shanno method is similar to the DFP update except that we seek a matrix that satisfies (11) but preserves similarity in terms of the inverses of $H_t$ and $H_{t-1}$. In other words, we want to find a symmetric positive definite $B_t$ that is as close as possible to $B_{t-1}$ in terms of the Frobenius norm, but preserves the constraint $\Delta x_t = B_t q_t$.

\[
B_t = (I_n - \frac{1}{q_t^T \delta x_t} \delta x_t q_t^T) B_{t-1} (I_n - \frac{1}{q_t^T \delta x_t} q_t \delta x_t^T) + \frac{\Delta x_t \Delta x_t^T}{q_t^T \delta x_t} \tag{16}
\]

Note that this formula simply switches $H$ with $B$, and swaps $q_t$ and $\delta x_t$ in (12).

Decades of experimentation has demonstrated that the BFGS method is somewhat more effective than the DFP method and is often preferred in practice.

4.1 Practical concerns

Initialization. There are generally two ways of initializing the Hessian $H_0$ (and its inverse $B_0$). The first is simply to start with an identity matrix. This causes the iteration to take the same first step as gradient descent, and then alter the search direction later as better Hessian estimates are obtained. The second method computes the true Hessian, either through finite differences or analytically, and inverts it. The second method produces better search
directions at the early stages of the iteration, but is more computationally expensive.

**Degeneracy.** In each of the above formulas there is a major possibility for the estimated Hessian matrices to diverge significantly from the true Hessian. It is also possible for the denominators $q_t^T \delta x_t$ to become negative or close to zero, or for the estimated Hessian to lose positive definiteness. The latter problem can be detected by testing the sign of $\nabla f(x_t)^T B_t \nabla f(x_t)$; if it is negative, then the Hessian estimate is surely no longer positive definite. If such faults are detected, it is useful to reset the Hessian estimate to the identity matrix or the true Hessian.

5 Exercises

1.