Optimization

Lecture notes for Combined Honours 3rd year

David Barber

Course Information: `www.ncrg.aston.ac.uk/~barberd/optstat.html`

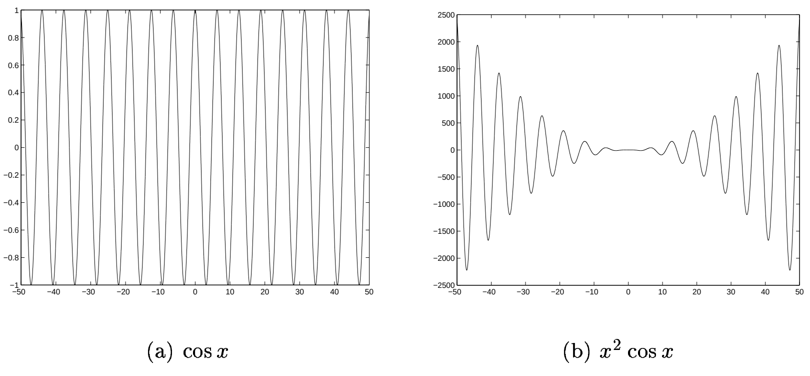

Figure 1: (a) The function $\cos x$. This has infinitely many local minima, each of which is also a global minimum. (b) The function $f(x) = x^2 \cos x$ also has infinitely many local minima, but no global minimum.

1 Introduction

The problem of optimization can be easily stated. You are given a function $f : \mathbb{R}^n \to \mathbb{R}$ and you want to find those $\mathbf{x} \in \mathbb{R}^n$ where f takes on a maximum of minimum value, and evaluate f at those points. As maximizing f is equivalent to minimizing $-f$, we shall just consider minimization.
The techniques that we use find *local* minima; there is little known about how to find global minima, although it is clear that stochastic methods do help. Note that in general, simply evaluating f' and finding a zero of this function is a bad idea: lots of critical points aren't minima, and how would we know?
For minimization in one dimension, we can choose between methods that require derivatives of f and those that just use function values. The former are more powerful in general, but not necessarily more effective: they will tend to converge faster, but may fail to converge where slower methods succeed. This is effectively because derivative based methods are constructing higher order interpolating polynomials to find the minimum.
In multi-dimensional search spaces, methods that use derivative information can be very useful to explore the geometry of the function. From a given point, there are infinitely many directions to search in: a single dot product $\nabla f.\mathbf{v}$ determines whether moving in the direction $\mathbf{v}$ is uphill or downhill. This dot product involves just $O(n)$ operations. In one dimension, there are just two directions to move in, so evaluating f is as good as evaluating f'.

2 One-dimensional minimization

One-dimensional optimization is important as it is a component of the multivariate minimization methods we will discuss later. These methods are iterative and rely on a *line-search* to find a minimum along a given search direction $\mathbf{v}$.
We shall first review some sufficient conditions for a point x^* to be a local minimum, and use these to obtain a rough estimate of the accuracy which can be attained when carrying out one-dimensional minimization numerically. We then turn to numerical optimization techniques. These can be divided into *dissection* methods (e.g. Golden section) and *fitted polynomial* methods (e.g. Quadratic interpolation and Brent's method).

2.1 Sufficient conditions for a minimum

For x^* to be a local minimum of the univariate function $f(x)$, we require that

$$f(x^*) \leq f(x)$$

for all x which are sufficiently close to x^*. More precisely, for x^* to be a local minimum, there must exist a positive constant Δ such that the above inequality holds for all x satisfying $|x - x^*| < \Delta$. Further, $\mathbf{x}^*$ is a global minimum if the inequality holds for all x, i.e. one can choose $\Delta = \infty$.

Example 1 The function $f(x) = \cos x$ has infinitely many local minima, each of them is also a global minimum. The function $f(x) = x^2 \cos x$ also has infinitely many local minima, but no global minimum, see fig(1)

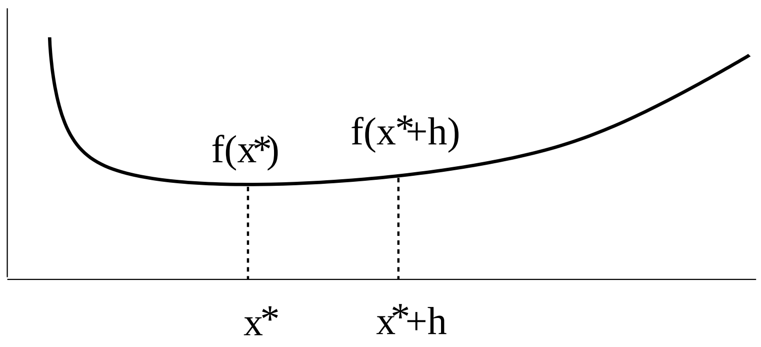

Figure 2: The precision to which we can find a minimum depends on how "flat" the function is around the miniumum. Even though we may be able to calculate function values to machine precision, we can typically only evaluate the minimum point x^* to about the square root of machine precision.

Now assume that $f(x)$ can be differentiated three times at some point x^*. Then we have the Taylor expansion:

$$f(x^* + h) = f(x^*) + f'(x^*)h + \frac{f''(x^*)}{2}h^2 + O(h^3) \ . \tag{1}$$

Choosing a small positive ϵ and setting $h = -f'(x^*)\epsilon$ we get $f(x^*+h) \approx f(x^*) - f'(x^*)^2\epsilon$ and thus $f(x^*+h) < f(x^*)$ unless $f'(x^*) = 0$. Consequently x^* can only be a local minimum if $f'(x^*) = 0$.
Let us assume that $f'(x^*) = 0$ and that further $f''(x^*)$ is greater than zero. Then for small values of h the Taylor expansion yields,

$$f(x^* + h) \approx f(x^*) + \frac{f''(x^*)}{2}h^2 \tag{2}$$

and thus $f(x^* + h) > f(x^*)$. So in this case x^* is a local minimum of f. In case that also $f''(x^*) = 0$, one has to take into account the behaviour of the higher order term $O(h^3)$.
To summarize, $f'(x^*) = 0, f''(x^*) > 0$ is a sufficient condition for x^* to be a local minimum.

2.2 Numerical Precision

Let x^* be a minimum satisfying $f'(x^*) = 0, f''(x^*) > 0$, and rewrite (equation (2)) as

$$f(x^* + h)/f(x^*) - 1 \approx \frac{f''(x^*)}{2f(x^*)}h^2 \ .$$

We consider a situation where we are carrying out some numerical search procedure for x^* and let us assume our current best approximation of x^* is $x^* + h$. Is there any point in trying to obtain a better approximation? Note that the numerical computations are not entirely accurate, and let ϵ be the relative machine precision. This means that two numbers a, b may numerically seem the same, if $|a/b - 1| < \epsilon$. So numerically we may not be able to see any difference between $f(x^* + h)$ and $f(x^*)$ once

$$\epsilon \approx \frac{f''(x^*)}{2|f(x^*)|}h^2,$$

and there is no point in trying to obtain a better approximation than $x^* + h$ to the location of the minimum. So we will have to stop the search procedure, once

$$|h| \approx \sqrt{\epsilon}\sqrt{2|f(x^*)/f''(x^*)|} \ .$$

The true value of x^* and thus of h is unknown, but typically the search algorithm is going to yield a sequence x_k of increasingly accurate approximations to x^*. So if the computations were perfectly accurate, we would have $\lim_{k\to\infty} x_k = x^*$. Due to numerical imprecision, however, it is sensible to stop the search, when

$$|x_{k+1} - x_k| \approx \sqrt{\epsilon}\sqrt{2|f(x^*)/f''(x^*)|} \ .$$

One might consider estimating the unknown ratio $f(x^*)/f''(x^*)$ using $f(x_k)/f''(x_k)$. Normally, however, one just assumes that $2|f(x^*)/f''(x^*)|$ is approximately 1 and stops when $|x_{k+1} - x_k| \leq \sqrt{\epsilon}$.

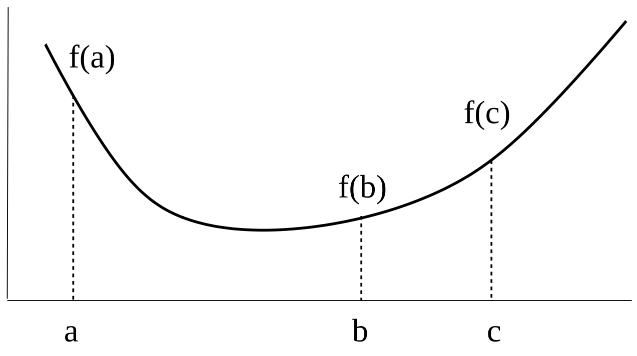

Figure 3: A bracket is a triple $\{a, b, c\}$ of real numbers, such that $a < b < c$ and $f(a) \geq f(b) \leq f(c)$. This means that there must exist a minimum somewhere in the interval $[a, c]$.

Example 2 Set $g(x) = \lambda f(x)$. Then x^* is also a local minimum of g and $g(x^*)/g''(x^*) = f(x^*)/f''(x^*)$. So this ratio is independent of the magnitude of the function.

While these considerations show, that the precision in finding the location x^* of the minimum will only be $\sqrt{\epsilon}$, the value of the function $f(x^*)$ is nevertheless estimated with a precision of ϵ.
For single precision computations on the Suns, $\sqrt{\epsilon}$ is about 3×10^{-4}, and for double precision it is about 10^{-8}.

2.3 Rate of convergence

In the sequel we shall consider algorithms which compute a sequence of improving approximation x_k to x^*. So, numerical considerations aside, $\lim_{k\to\infty} x_k = x^*$. One way of comparing different algorithms, is to consider the rate of convergence p of the sequence x_k. The rate of convergence p is defined to be the largest number for which it is possible to find a bound C_p so that the following inequality holds for all k:

$$\frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} \leq C_p$$

Example 3 The sequence $x_k = 2^{-k}$ has rate of convergence $p = 1$. Obviously $x^* = 0$ and

$$\frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} = \frac{2^{-k-1}}{2^{-kp}} = 2^{-k-1+kp} = 2^{-1}2^{k(p-1)}$$

So for $p = 1$ we can use $C_1 = 2^{-1}$ whereas $2^{k(p-1)}$ diverges for $p > 1$.

Example 4 The sequence $x_k = 2^{-(2^k)}$ has rate of convergence $p = 2$.

2.4 Brackets

A bracket for minimizing a function f, is a triple $\{a, b, c\}$ of real numbers, such that $a < b < c$ and

$$f(a) \geq f(b) \leq f(c)\ .$$

Since the value of f on the interior point b is smaller or equal to the value on the boundary points a and c, one will expect that there is a local minimum somewhere inside the interval $[a, c]$. The next proposition shows that that is indeed the case if f is continuous.

Example 5 The points $\{0, \pi, 2\pi\}$ are a bracket for minimizing $f(x) = \sin x$ and $x^* = 3\pi/2$ is a local minimum.

Proposition If $\{a, b, c\}$ is a bracket for minimizing the continuous function f, then f has a local minimum x^* with $a < x^* < c$.
Proof:. From Real Analysis we know that the continuous function f must attain a smallest value on the closed interval $[a, c]$; i.e. there is a point $\tilde{x}$ with

$$a \leq \tilde{x} \leq c \quad \text{and} \quad f(\tilde{x}) \leq f(x) \text{ for all } x \in [a, c]\ .$$

If $a < \tilde{x} < c$, we set $x^* = \tilde{x}$ and have found our local minimum. Otherwise $a = \tilde{x}$ or $c = \tilde{x}$. In both cases this means that $f(b) \leq f(\tilde{x})$ since $\{a, b, c\}$ is a bracket. However, since by construction $f(\tilde{x})$ is the smallest value we also have $f(b) \geq f(\tilde{x})$. Thus $f(b) = f(\tilde{x})$ and $x^* = b$ is a local minimum of f.
So having a bracket is important since it gives us an indication in which range we have to hunt for a local minimum. But just given a function f, how are we going to obtain a bracket in the first place? We just pick any initial point z and a step size $s > 0$. Let us assume that $f(z) \geq f(z+s)$. We then evaluate f on the sequence $x_i = z + is$, starting with $i = 0$, until we find the first i such that $f(x_i) \leq f(x_{i+1})$. If we find such an i the triple $\{x_0, x_i, x_{i+1}\}$ is a bracket. Of course it might be that there is no such i, that is $f(x_i) > f(x_{i+1})$ for all $i = 0, \ldots, \infty$. In this case we are at least finding points on which f attains smaller and smaller values. We have assumed $f(z) \geq f(z+s)$, and so still have to consider the case that $f(z) < f(z+s)$. In this case we can scan the sequence $x_i = x + (1-i)s$, starting with $i = 0$, until we find the first i with $f(x_i) < f(x_{i+1})$. If we are successful, $\{x_{i+1}, x_i, x_0\}$ is a bracket.
While this demonstrates the principles (and limitations) of methods for finding a bracket, using a fixed step size s is not always the best strategy. It can be useful to exponentially increase the step size by using a sequence like $x_i = z + 2^i s$.

A bracket $\{a, b, c\}$ for minimizing f provides a rough approximation for the location of a local minimum, it must lie somewhere between a and c. Once we have bracket, it is quite straightforward to obtain a better approximation, by picking a point $x \neq b$ in the interval (a, c) and comparing $f(x)$ to $f(b)$.
Assume we pick x such that $a < x < b$, then

$$\begin{aligned} \text{either } f(x) \geq f(b), &\quad \text{so } \{x, b, c\} \quad \text{is a new bracket,} \\ \text{or } f(x) < f(b), &\quad \text{so } \{a, x, b\} \quad \text{is a new bracket.} \end{aligned}$$

In the first case, it is obvious that $\{x, b, c\}$ is a bracket. For the second case, $f(x) < f(b)$, note that $f(a) \geq f(b)$, since $\{a, b, c\}$ is a bracket. So $f(a) > f(x)$ and $\{a, x, b\}$ is indeed a bracket.
In the case that we choose to pick x such that $b < x < c$ the rules are:

$$\begin{aligned} \text{Either } f(x) \geq f(b), &\quad \text{so } \{a, b, x\} \quad \text{is a new bracket,} \\ \text{or } f(x) < f(b), &\quad \text{so } \{b, x, c\} \quad \text{is a new bracket.} \end{aligned}$$

So, by choosing an interior point x in the current bracket and obtaining a new bracket as described above, we arrive at better and better approximations to the location of the minimum. In next two section, we shall consider clever strategies for picking x.

2.5 Golden Section Search

One approach to choosing x, is to try and make the length of the new bracket a small as possible. Assume that we pick x in the interval (a, b), then the new bracket is going to be $\{x, b, c\}$, with length $c - x$, or $\{a, x, b\}$, with length $b - a$. Since we don't know the outcome before having done the hard work of evaluating $f(x)$, we want to hedge our bets and choose x so that lengths of the two possible new brackets are the same, i.e. $c - x = b - a$. In the case that x is in the interval (b, c), the same reasoning yields the equation $x - a = c - b$. Both equations for x have the same solution

$$x = a + c - b, \tag{3}$$

so it doesn't matter which of the two assumptions, $x \in (a, b)$ or $x \in (b, c)$, holds. The above prescription for choosing x, however, has the problem that it yields $x = b$, if b lies right in the middle between a and c. Of course we must evaluate f at a new point $x \neq b$ and for numerical reason we should also beware of choosing a value for x which is close to b.
The following reasoning yields a close to optimal and safe way of choosing x. Let λ be the ratio of the length of the smaller of the two intervals to the length of the entire bracket.i.e.

$$\lambda = \min\left\{\frac{b-a}{c-a}, \frac{c-b}{c-a}\right\} .$$

Note that brackets with a value of λ close zero are undesirable since this means that b is close to one of the endpoints. But values of λ close to $\frac{1}{2}$ are bad as well, because they yield x close to b on application of (3). We now calculate a safe intermediate value for λ, which reproduces itself when (3) is applied. Let us assume that the first interval is smaller, $b - a < c - b$ then using (3) will yield an $x \in (b, c)$. Thus the new bracket is going to be either $\{a, b, x\}$ or $\{b, x, c\}$. Let us assume the latter and further assume that $x - b < c - x$. Then the value of λ for the new bracket $\{b, x, c\}$ is

$$\lambda_{\text{new}} = \frac{x-b}{c-b} \, .$$

Using (3) and that $\lambda = (b-a)/c-a)$, $1-\lambda = (c-b)/(c-a)$ one calculates

$$\begin{aligned} \lambda_{\text{new}} &= \frac{a+c-b-b}{c-b} = \frac{c-b-(b-a)}{c-b} = 1 - \frac{b-a}{c-b} = 1 - \frac{b-a}{c-a}\frac{c-a}{c-b} \\ &= 1 - \lambda\frac{1}{1-\lambda} = \frac{1-2\lambda}{1-\lambda} \end{aligned} \tag{4}$$

We would like $\lambda_{\text{new}} = \lambda$ and this yields the quadratic equation $\lambda = (1-2\lambda)/(1-\lambda)$. One of its solutions is greater than $\frac{1}{2}$ and useless, the other one is:

$$\lambda = \frac{3-\sqrt{5}}{2} \simeq 0.38197 \, .$$

This value is the *golden mean*, a quantity that was studied by Pythagoras (who lived around 570–490 B.C.). It was supposed by the ancient Greeks to be the most aesthetic proportion in which to divide a line: echoes of its use can be found in Leonardo da Vinci (and other painter/geometers of the Italian Renaissance), Dali, and Mondrian, amongst many other artists.

The derivation assumed that the new bracket is $\{b, x, c\}$, but analysing the case $\{a, b, x\}$ yields the same result for λ. So, if $b - a < c - b$ in Golden Section Search the new point x is chosen as

$$x = b + \lambda(c-b) \, .$$

If $b - a \geq c - b$ the prescription is:

$$x = b + \lambda(a-b) \, .$$

In both cases the next point to be evaluated is that which is a fraction 0.38197 into the larger of the two intervals (measuring from the central point).

The rate of convergence of Golden Section Search is linear. (Algorithm can be found on pp. 400–402 of NR).

▶ **Exercise 1** *(10 marks) Analytically find the minimum of the function*

$$f(x) = 1 - x^2 e^{-x}.$$

Sketch (or plot) a graph of this function. Verify that for the triple $a = 1, b = 1.5, c = 3$, $f(b) < f(c)$ and $f(b) < f(a)$. Apply Golden section search to find the minimum; carry out enough iterations of the procedure to locate the minimum to within ± 0.1.

You may do this question either by hand or by programming in MATLAB.

2.6 Polynomial Interpolation and Brent's Algorithm

A second approach is to try and choose x so that is a good approximation to the true minimum x. This is done by fitting a polynomial to the curve and analytically calculating the local minimum of the polynomial. The simplest case is to use a parabola. Given a bracket $\{a, b, c\}$ for minimizing f Lagrange's formula

$$P(x) = \frac{(x-b)(x-c)}{(a-b)(a-c)}f(a) + \frac{(x-a)(x-c)}{(b-a)(b-c)}f(b) + \frac{(x-a)(x-b)}{(c-a)(c-b)}f(c)$$

yields a quadratic function with $P(a) = f(a)$, $P(b) = f(b)$ and $P(c) = f(c)$. P has a unique minimum x except in the singular case $f(a) = f(b) = f(c)$. Since $\{a, b, c\}$ is also a bracket for minimizing P, this value of x must lie inside the bracket. Solving $P'(x) = 0$ gives the formula for the new point x:

$$x = b - \frac{1}{2}\frac{(b-a)^2\,[f(b)-f(c)] - (b-c)^2\,[f(b)-f(a)]}{(b-a)\,[f(b)-f(c)] - (b-c)\,[f(b)-f(a)]}$$

Although for a smooth function f quadratic interpolation has asymptotic superlinear convergence ($p \simeq 1.324$), it can be very slow when outside of the asymptotic regime. For example, if b is already at or near the minimum of the interpolating parabola, then the new point will be close to b. This causes slow convergence if b is not close to the local minimum of f. By contrast, the linear convergence of the Golden section method is guaranteed from the outset. It can therefore be useful to use a *hybrid* algorithm which tries to combine the best features of both methods in order to minimize the number of function evaluations required. Brent's algorithm (NR 404–5) is an example of such a hybrid algorithm. Safeguards are included to prevent iterates from being too close.

3 Multivariate Calculus

3.1 Matrix Algebra

For an l by n matrix A and an n by m matrix B, the product AB is the l by m matrix C with the following components

$$C_{ik} = (AB)_{ij} = \sum_{j=1}^{n} A_{ij} B_{jk}\,; \qquad i = 1, \dots, l \quad k = 1, \dots, m\,.$$

Note that even if BA is defined as well, that is if $l = n$, except in very special cases BA is not equal to AB. One often thinks of a n dimensional vector $\mathbf{x}$ as being an n by 1 matrix.

Example 6

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_2 + a_{22}x_2 \end{pmatrix}$$

The transpose B^T of the n by m matrix B is the m by n matrix D with components

$$D_{kj} = (B^T)_{kj} = B_{jk}\,; \qquad k = 1, \dots, m \quad j = 1, \dots, n\,.$$

Example 7

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}^T = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

Transposing B twice, again yields B, $(B^T)^T = B$. Further $(AB)^T = B^T A^T$ since

$$((AB)^T)_{ki} = (AB)_{ik} = \sum_{j=1}^{n} A_{ij}B_{jk} = \sum_{j=1}^{n} (A^T)_{ji}(B^T)_{kj} = \sum_{j=1}^{n} (B^T)_{kj}(A^T)_{ji} = (B^T A^T)_{ki}\,.$$

If the shapes of the matrices A,B and C are such that it makes sense to calculate the product ABC, then also $(ABC)^T = C^T B^T A^T$.
A square matrix A, i.e. $l = n$, is symmetric if $A^T = A$.

Example 8

$$\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix} \text{ is symmetric; } \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \text{ is not.}$$

A real number a can be thought of as a symmetric 1 by 1 matrix, so $a^T = a$. The inner product of two n-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$ can be written as $\mathbf{x}^T\mathbf{y}$ and this is a real number. In particular

$$\mathbf{x}^T\mathbf{y} = (\mathbf{x}^T\mathbf{y})^T = \mathbf{y}^T(\mathbf{x}^T)^T = \mathbf{y}^T\mathbf{x}\,.$$

If A is a symmetric n by n matrix, the expression $\mathbf{x}^T A\mathbf{y}$ yields a kind of generalized inner product. In particular, since $\mathbf{x}^T A\mathbf{y}$ is a scalar, we again have

$$\mathbf{x}^T A\mathbf{y} = (\mathbf{x}^T A\mathbf{y})^T = \mathbf{y}^T A^T (\mathbf{x}^T)^T = \mathbf{y}^T A\mathbf{x}\,.$$

In the sequel we shall often deal with multivariate quadratic functions. We will write these in the form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$, where A is a symmetric n by n matrix, $\mathbf{x}$ and $\mathbf{b}$ are n-dimensional vectors, and c is a real constant. It is important to be able to convert between vector and coordinate notation:

Example 9

$$\frac{1}{2}\mathbf{x}^T \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \mathbf{x} - \begin{pmatrix} 4 \\ 5 \end{pmatrix}^T \mathbf{x} + 6 = \frac{1}{2}x_1^2 + \frac{1}{2}3x_2^2 + 2x_1x_2 - 4x_1 - 5x_2 + 6\,.$$

In the case $\mathbf{b} = \mathbf{0}$ and $c = 0$ the conversion is:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} = \frac{1}{2}\mathbf{x}^T \begin{pmatrix} \sum_{j=1}^n A_{1j}x_j \\ \vdots \\ \sum_{j=1}^n A_{nj}x_j \end{pmatrix} = \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n x_i A_{ij} x_j$$

Does $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x}$ have a global minimum? Assume we can find a vector $\mathbf{v}$, such that $f(\mathbf{v}) < 0$. For any scalar λ we have $f(\lambda\mathbf{v}) = \lambda^2 f(\mathbf{v})$, and by choosing large values for λ, we can make $f(\lambda\mathbf{v})$ as small as we like. So in this case f does not have a global minimum.
A symmetric matrix A, with the property that $\mathbf{x}^T A\mathbf{x} \geq 0$ for any vector $\mathbf{x}$ is called *nonnegative definite.* For such a matrix, the point $\mathbf{x} = \mathbf{0}$ is a global minimum of f.
A symmetric matrix A, with the property that $\mathbf{x}^T A\mathbf{x} > 0$ for any vector $\mathbf{x} \neq 0$ is called *positive definite.* For such a matrix, the point $\mathbf{x} = \mathbf{0}$ is the unique global minimum of f.

Example 10 The matrix

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 3 \end{pmatrix}$$

is positive definite. $\mathbf{x}^T A\mathbf{x} = x_1^2 + 2x_1x_2 + 3x_2^2 = (x_1 + x_2)^2 - x_2^2 + 3x_2^2 = (x_1 + x_2)^2 + 2x_2^2$. So $\mathbf{x}^T A\mathbf{x} \geq 0$; and $\mathbf{x}^T A\mathbf{x} = 0$ implies $x_1 + x_2 = 0$ and $x_2 = 0$, i.e. $\mathbf{x} = \mathbf{0}$.

Note that a positive definite matrix has full rank and is thus invertible: If A does not have full rank we can find a $\mathbf{v} \neq \mathbf{0}$ with $A\mathbf{v} = \mathbf{0}$. But then also $\mathbf{v}^T A\mathbf{v} = 0$, and A is not positive definite.

▶ **Exercise 2** *(5 marks) Show that $f(\mathbf{x}) = \frac{1}{2}(\sum_{i=1}^n x_i)^2$ can be written as $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x}$ for a suitable symmetric matrix A. Is A positive definite for some value of n? Does f have a local minimum or only global minima? What is the global minimum value and where is it?*

3.2 Partial derivatives

Consider a function of n variables, $f(x_1, x_2, \ldots, x_n)$ or $f(\mathbf{x})$. The partial derivative of f wrt x_1 at $\mathbf{x}^*$ is defined as the following limit (when it exists)

$$\left.\frac{\partial f}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}^*} = \lim_{h\to 0} \frac{f(x_1^* + h, x_2^*, \ldots, x_n^*) - f(\mathbf{x}^*)}{h}$$

The *gradient vector* of f will be denoted by ∇f or $\mathbf{g}$

$$\nabla f(\mathbf{x}) \equiv \mathbf{g}(\mathbf{x}) \equiv \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

Example 11 The gradient of $f(\mathbf{x}) = x_2 \cos(x_1 - x_2)$ is

$$\nabla f(\mathbf{x}) = \begin{pmatrix} -x_2 \sin(x_1 - x_2) \\ \cos(x_1 - x_2) + x_2 \sin(x_1 - x_2) \end{pmatrix}$$

3.3 Higher derivatives

The "first derivative" of a function of n variables is an n-vector; the "second-derivative" of an n-variable function is defined by the n^2 partial derivatives of the n first partial derivatives wrt the n variables

$$\frac{\partial}{\partial x_i}\left(\frac{\partial f}{\partial x_j}\right) \qquad i = 1, \ldots, n; \;\; j = 1, \ldots, n$$

which is usually written

$$\frac{\partial^2 f}{\partial x_i \partial x_j}, \; i \neq j \qquad \frac{\partial^2 f}{\partial {x_i}^2}, \; i = j$$

If the partial derivatives $\partial f/\partial x_i$, $\partial f/\partial x_j$ and $\partial^2 f/\partial x_i \partial x_j$ are continuous, then $\partial^2 f/\partial x_i \partial x_j$ exists and

$$\partial^2 f/\partial x_i \partial x_j = \partial^2 f/\partial x_j \partial x_i .$$

These n^2 second partial derivatives are usually represented by a square, symmetric matrix called the *Hessian* matrix of $f(\mathbf{x})$.

$$H_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial {x_1}^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 f}{\partial {x_n}^2} \end{pmatrix}$$

Example 12 The Hessian of $f(\mathbf{x}) = x_2 \cos(x_1 - x_2)$ is

$$H_f(\mathbf{x}) = \begin{pmatrix} -x_2 \cos(x_1 - x_2) & -\sin(x_1 - x_2) + x_2 \cos(x_1 - x_2) \\ -\sin(x_1 - x_2) + x_2 \cos(x_1 - x_2) & 2\sin(x_1 - x_2) - x_2 \cos(x_1 - x_2) \end{pmatrix}$$

3.4 Chain Rule

Consider $f(x_1, \ldots, x_n)$. Now let each x_j be parameterized by $u_1, \ldots, u_n$, i.e. $x_j = x_j(u_1, \ldots, u_m)$. What is $\partial f/\partial u_\alpha$?

$$\begin{aligned} \delta f &= f(x_1 + \delta x_1, \ldots, x_n + \delta x_n) - f(x_1, \ldots, x_n) \\ &= \sum_{j=1}^{n} \frac{\partial f}{\partial x_j} \delta x_j + \text{higher order terms} \\ \delta x_j &= \sum_{\alpha=1}^{m} \frac{\partial x_j}{\partial u_\alpha} \delta u_\alpha + \text{higher order terms} \end{aligned}$$

So

$$\delta f = \sum_{j=1}^{n} \sum_{\alpha=1}^{m} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial u_\alpha} \delta u_\alpha + \text{higher order terms}$$

Therefore

$$\frac{\partial f}{\partial u_\alpha} = \sum_{j=1}^{n} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial u_\alpha}$$

or in vector notation

$$\frac{\partial}{\partial u_\alpha} f(\mathbf{x}(\mathbf{u})) = \nabla f^T(\mathbf{x}(\mathbf{u})) \frac{\partial \mathbf{x}(\mathbf{u})}{\partial u_\alpha}$$

3.5 Directional derivatives

Assume f is differentiable. We want to define the directional derivative $(D_\mathbf{v} f)(\mathbf{x}^*)$ of f in a direction $\mathbf{v}$ at a point $\mathbf{x}^*$. Let $\mathbf{x} = \mathbf{x}^* + h\mathbf{v}$, i.e. $x_j = x_j^* + hv_j$. Then

$$(D_\mathbf{v} f)(\mathbf{x}^*) = \frac{d}{dh} f(\mathbf{x}^* + h\mathbf{v}) \bigg|_{h=0}$$

Example 13 The directional derivative of $f(\mathbf{x}) = x_2 \cos(x_1 - x_2)$ in the direction $\mathbf{v} = (1,1)^T$:

$$f(\mathbf{x}^* + h\mathbf{v}) = (x_2^* + h) \cos(x_1^* + h - (x_2^* + h)) = f(\mathbf{x}^*) + h \cos(x_1^* - x_2^*) .$$

So $(D_\mathbf{v} f)(\mathbf{x}^*) = \cos(x_1^* - x_2^*)$.

By the chain rule we obtain

$$\frac{d}{dh} f(\mathbf{x}^* + h\mathbf{v}) = \sum_j v_j \frac{\partial f}{\partial x_j} \bigg|_{\mathbf{x}=\mathbf{x}^*+h\mathbf{v}}$$

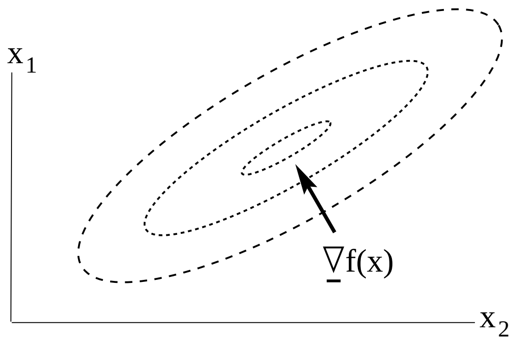

Figure 4: Interpreting the gradient. The ellipses are contours of constant function value, $f = const$. At any point $\mathbf{x}$, the gradient vector $\nabla f(\mathbf{x})$ points along the direction of maximal increase of the function.

$$
\begin{aligned}
(D_{\mathbf{v}}f)(\mathbf{x}^*) &= \sum_j v_j \left.\frac{\partial f}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}^*} \\
&= \nabla f^T \mathbf{v}
\end{aligned}
$$

Example 14 The directional derivative of $f(\mathbf{x}) = x_2 \cos(x_1 - x_2)$ in the direction $\mathbf{v} = (1,1)^T$:

$$
(D_{\mathbf{v}}f)(\mathbf{x}^*) = \begin{pmatrix} -x_2^* \sin(x_1^* - x_2^*) \\ \cos(x_1^* - x_2^*) + x_2^* \sin(x_1^* - x_2^*) \end{pmatrix}^T \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \cos(x_1^* - x_2^*).
$$

Example 15 The directional derivative of the linear function $f(\mathbf{x}) = \mathbf{b}^T\mathbf{x}$. We have $f(\mathbf{x}^* + h\mathbf{v}) = f(\mathbf{x}) + h\mathbf{b}^T\mathbf{v}$, so $(D_{\mathbf{v}}f)(\mathbf{x}^*) = \mathbf{b}^T\mathbf{v}$. Comparing to the general formula $(D_{\mathbf{v}}f)(\mathbf{x}^*) = \nabla f^T\mathbf{v}$ we obtain

$$
\nabla f(\mathbf{x}^*) = \mathbf{b}\,.
$$

▶ **Exercise 3** *(10 marks) Rosenbrock's function is given by*

$$
f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2
$$

Find expressions for all of the first and second derivatives of this function of 2 variables. Verify that $\mathbf{x}^* = (1,1)^T$ *satisfies* $\mathbf{g}^* = \mathbf{0}$ *and that* H^* *is positive definite.*
Show that $H(\mathbf{x})$ *is singular iff* $\mathbf{x}$ *satisfies the condition* $(x_2 - x_1^2) = 0.005$.
Calculate the directional derivative of the function in the direction $\mathbf{v} = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})^T$ *evaluated at the point* $\mathbf{x}' = (-1,0)^T$.

▶ **Exercise 4** *(5 marks) Calculate the directional derivative of* $f(\mathbf{x}) = x_1^2(x_1+x_2)^2(x_1+x_2+x_3)^2$ *at a point* $\mathbf{x}$ *in the direction* $\mathbf{p} = (1,-1,0)^T$.

3.6 Interpreting the Gradient $\nabla f(\mathbf{x})$

The gradient points along the direction in which the function increases most rapidly. Why?

Consider a direction $\hat{\mathbf{p}}$ (a unit legnth vector). Then a displacement, δ units along this direction changes the function value to

$$
f(\mathbf{x} + \delta\hat{\mathbf{p}}) \approx f(\mathbf{x}) + \delta\nabla f(\mathbf{x}) \cdot \hat{\mathbf{p}}
$$

The direction $\hat{\mathbf{p}}$ for which the function has the largest change is that which maximises the overlap

$$
\nabla f(\mathbf{x}) \cdot \hat{\mathbf{p}} = |\nabla f(\mathbf{x})||\hat{\mathbf{p}}| \cos\theta = |\nabla f(\mathbf{x})| \cos\theta
$$

where θ is the angle between $\hat{\mathbf{p}}$ and $\nabla f(\mathbf{x})$. The overlap is maximised when $\theta = 0$, giving $\hat{\mathbf{p}} = \nabla f(\mathbf{x})/|\nabla f(\mathbf{x})|$. Hence, the direction along which the function changes the most rapidly is along $\nabla f(\mathbf{x})$.

3.7 Taylor's theorem

Recall Taylor's theorem for univariate functions

$$F(x+h) = F(x) + hF'(x) + \frac{h^2}{2}F''(x) + \ldots + \frac{h^{r-1}}{(r-1)!}F^{(r-1)}(x) + O(h^r)\ ,$$

assuming $F(x) \in C^r$.
Set $F(h) = f(\mathbf{x}^* + h\mathbf{v})$ for some vector $\mathbf{v}$. Then

$$f(\mathbf{x}^* + h\mathbf{v}) = F(h) = F(0) + hF'(0) + \frac{h^2}{2}F''(0) + O(h^3)\ .$$

Of course $F(0) = f(\mathbf{x}^*)$ and

$$F'(h) = \frac{d}{dh}f(\mathbf{x}^* + h\mathbf{v}) = \sum_j v_j \frac{\partial f}{\partial x_j}\bigg|_{\mathbf{x}=\mathbf{x}^*+h\mathbf{v}}\ .$$

So

$$F'(0) = \frac{d}{dh}f(\mathbf{x}^* + h\mathbf{v})\bigg|_{h=0} = (D_{\mathbf{v}}f)(\mathbf{x}^*) = \nabla f^T(\mathbf{x}^*)\mathbf{v}\ .$$

Further

$$\begin{aligned} F''(h) &= \frac{d}{dh}\sum_j v_j \frac{\partial f}{\partial x_j}\bigg|_{\mathbf{x}=\mathbf{x}^*+h\mathbf{v}} \\ &= \sum_j v_j \frac{d}{dh}\frac{\partial}{\partial x_j}f\bigg|_{\mathbf{x}=\mathbf{x}^*+h\mathbf{v}} \\ &= \sum_j v_j \sum_i v_i \frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j}f\bigg|_{\mathbf{x}=\mathbf{x}^*+h\mathbf{v}}\ . \end{aligned}$$

Thus

$$F''(0) = \sum_{ij} v_i v_j \frac{\partial^2 f}{\partial x_i \partial x_j}\bigg|_{\mathbf{x}=\mathbf{x}^*} = v^T H_f(\mathbf{x}^*)v$$

Inserting yields:

$$f(\mathbf{x}^* + h\mathbf{v}) = f(\mathbf{x}^*) + h\nabla f^T\mathbf{v} + \frac{h^2}{2}\mathbf{v}^T H_f \mathbf{v} + O(h^3)$$

Example 16 For a quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$, with symmetric A, one finds:

$$\begin{aligned} f(\mathbf{x}+h\mathbf{v}) &= \frac{1}{2}(\mathbf{x}+h\mathbf{v})^T A(\mathbf{x}+h\mathbf{v}) - \mathbf{b}^T(\mathbf{x}+h\mathbf{v}) + c \\ &= \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \frac{1}{2}h\mathbf{v}^T A\mathbf{x} + \frac{1}{2}h\mathbf{x}^T A\mathbf{v} + \frac{1}{2}h^2\mathbf{v}^T A\mathbf{v} - \mathbf{b}^T\mathbf{x} - h\mathbf{b}^T\mathbf{v} + c \\ &= \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c + h(\frac{1}{2}\mathbf{v}^T A\mathbf{x} + \frac{1}{2}\mathbf{x}^T A\mathbf{v} - \mathbf{b}^T\mathbf{v}) + \frac{1}{2}h^2\mathbf{v}^T A\mathbf{v} \\ &= f(\mathbf{x}) + h(\mathbf{x}^T A\mathbf{v} - \mathbf{b}^T\mathbf{v}) + \frac{h^2}{2}\mathbf{v}^T A\mathbf{v} \\ &= f(\mathbf{x}) + h(A\mathbf{x} - \mathbf{b})^T\mathbf{v} + \frac{h^2}{2}\mathbf{v}^T A\mathbf{v} \end{aligned}$$

Comparing to Taylor's formula above now yields that for a quadratic function:

$$\nabla f(\mathbf{x}) = A\mathbf{x} - \mathbf{b} \text{ and } H_f(\mathbf{x}) = A\,.$$

3.8 Critical points

When all first-order partial derivatives at a point are zero (i.e. $\nabla f = \mathbf{0}$) then the point is said to be a stationary or critical point. Can be a minimum, maximum or saddle point.

• Necessary first-order condition for a minimum.
There is a minimum of f at $\mathbf{x}^*$ if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ sufficiently close to $\mathbf{x}^*$. Let $\mathbf{x} = \mathbf{x}^* + h\mathbf{v}$ for small h and some direction $\mathbf{v}$. Then by a Taylor expansion, for small h,

$$f(\mathbf{x}^* + h\mathbf{v}) = f(\mathbf{x}^*) + h\nabla f^T \mathbf{v} + O(h^2)$$

and thus for a minimum

$$h\nabla f^T \mathbf{v} + O(h^2) \geq 0$$

Choosing $\mathbf{v}$ to be $-\nabla f$ the condition becomes

$$-h\nabla f^T \nabla f + O(h^2) \geq 0$$

and is violated for small positive h unless $|\nabla f|^2 = \nabla f^T \nabla f = 0$. So $\mathbf{x}^*$ can only be a local minimum if $|\nabla f(\mathbf{x}^*)| = 0$, i.e if $\nabla f(\mathbf{x}^*) = \mathbf{0}$.

• Necessary second-order condition for a minimum.
At a stationary point $\nabla f = \mathbf{0}$. Hence the Taylor expansion is given by

$$f(\mathbf{x}^* + h\mathbf{v}) = f(\mathbf{x}^*) + h^2 \mathbf{v}^T H_f \mathbf{v} + O(h^3)$$

Thus the minimum condition requires that $\mathbf{v}^T H_f \mathbf{v} \geq 0$, i.e. *the Hessian is non-negative definite.*
Sufficient conditions for a minimum at $\mathbf{x}^*$ are (i) $\nabla f(\mathbf{x}^*) = 0$ and (ii) $H_f(\mathbf{x}^*)$ is positive definite.

Example 17 $x_1^4 + x_2^2$ has a minimum at $\mathbf{x}^* = (0,0)^T$, but H is not positive definite.

Example 18 For a quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$, with symmetric A the necessary condition $\nabla f(\mathbf{x}^*) = \mathbf{0}$ reads:

$$A\mathbf{x}^* - \mathbf{b} = 0$$

If A is invertible this equation has the unique solution $\mathbf{x}^* = A^{-1}\mathbf{b}$. If A is positive definite, $\mathbf{x}^*$ is a minimum.

4 Multivariate Minimization 1: Gradient Descent

Almost all of the search techniques that we consider are iterative, i.e. we proceed towards the minimum $\mathbf{x}^*$ by a sequence of steps. On the kth step we take a step of length α_k in the direction $\mathbf{p}_k$,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{5}$$

The length of the step can either be chosen using prior knowledge, or by carrying out a line search in the direction $\mathbf{p}_k$. It is the way that $\mathbf{p}_k$ is chosen that tends to distinguish the different methods of multivariate optimization that we will discuss.
We shall assume that we can analytically evaluate the gradient of f and will often use the shorthand notation

$$\mathbf{g}_k = \nabla f(\mathbf{x}_k) \, .$$

Typically we will want to choose $\mathbf{p}_k$ using only gradient information; for large problems it can be very expensive to compute the Hessian, and this can also require a large amount of storage.

4.1 Exact Line Search Condition

At the k-th step, we chose α_k to minimize $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k)$. So setting $F(\lambda) = f(\mathbf{x}_k + \lambda \mathbf{p}_k)$, at this step we solve the one-dimensional minimization problem for $F(\lambda)$. Thus our choice of $\alpha_k = \lambda^*$ will satisfy $F'(\alpha_k) = 0$.

Now

$$\begin{aligned}F'(\alpha_k) &= \frac{d}{dh}F(\alpha_k+h)_{|h=0}\\ &= \frac{d}{dh}f(\mathbf{x}_k+\alpha_k\mathbf{p}_k+h\mathbf{p}_k)_{|h=0}\\ &= \frac{d}{dh}f(\mathbf{x}_{k+1}+h\mathbf{p}_k)_{|h=0}\\ &= (D_{\mathbf{p}_k}f)(\mathbf{x}_{k+1})\\ &= \nabla f^T(\mathbf{x}_{k+1})\mathbf{p}_k\ .\end{aligned}$$

So $F'(\alpha_k)=0$ means the the directional derivative in the search direction must vanish at the new point and this gives the Exact Line Search Condition:

$$0=\mathbf{g}_{k+1}^T\mathbf{p}_k\ . \tag{6}$$

For a quadratic function $f(\mathbf{x})=\frac{1}{2}\mathbf{x}^TA\mathbf{x}-\mathbf{b}^T\mathbf{x}+c$, with symmetric A, we can use the condition to analytically calculate α_k. Since $\nabla f(\mathbf{x}_{k+1})=A\mathbf{x}_k+\alpha_kA\mathbf{p}_k-\mathbf{b}=\nabla f(\mathbf{x}_k)+\alpha_kA\mathbf{p}_k$ we find

$$\alpha_k=-\frac{\mathbf{p}_k^T\mathbf{g}_k}{\mathbf{p}_k^TA\mathbf{p}_k}\ .$$

4.2 Gradient descent

Perhaps the simplest choice for $\mathbf{p}_k$ is to set it equal to $-\mathbf{g}_k$. If we find that $\mathbf{g}_k=\mathbf{0}$ we can stop. Otherwise we can guarantee that $f(\mathbf{x}_{k+1})<f(\mathbf{x}_k)$, as long as α_k is positive and small enough. To see this, expand f around $\mathbf{x}_k$ using Taylor's theorem:

$$f(\mathbf{x}_k+\alpha_k\mathbf{p}_k)\approx f(\mathbf{x}_k)+\alpha_k\mathbf{g}_k^T\mathbf{p}_k\ .$$

With $\mathbf{p}_k=-\mathbf{g}_k$ and for small positive α_k, we see a guaranteed reduction:

$$f(\mathbf{x}_k+\alpha_k\mathbf{p}_k)\approx f(\mathbf{x}_k)-\alpha_k||\mathbf{g}_k||^2\ .$$

The algorithm for gradient descent can be written

0cm

```
input $\mathbf{x}_0$, $gtol$
$k := 0$
while $||\mathbf{g}_k|| > gtol$
  $\mathbf{p}_k := -\mathbf{g}_k$
  compute $\alpha_k$ to minimize $f(\mathbf{x}_k+\alpha_k\mathbf{p}_k)$
  $\mathbf{x}_{k+1} := \mathbf{x}_k+\alpha_k\mathbf{p}_k$
  $k := k+1$
end
```

In general the *compute* α_k step is carried out using a one dimensional minimization method, e.g. Golden Section. For a quadratic function, α_k can be computed with the formula in the preceding section. For a quadratic function the rate of convergence will only be linear, unless we happen to start off with $\mathbf{x}_1$ lying on one of the principal axes, in which case we will find the minimum in one step.
This slow rate of convergence is unacceptable; for a quadratic function it should be possible to obtain *quadratic termination*, which is that the minimum is found in a finite number of steps. After all, we could explicitly calculate the minimum using $\mathbf{x}^*=A^{-1}\mathbf{b}$.
Let us consider the following 2-dimensional example to understand the drawbacks of gradient descent. Assume f is the negative (since we are minimizing) life expectancy of a 30-year old who weighs x_1 pounds and has height of x_2 feet, and we use gradient descent to find the optimal value starting from some initial point $\mathbf{x}_1$. Now instead of measuring in pounds and feet, we might just as well use kilos and meters. Let ϕ be the function that converts from metric to imperial

$$\phi(\hat{x}_1,\hat{x}_2)=(\lambda_1\hat{x}_1,\lambda_2\hat{x}_2)\ ,$$

then $\hat{f}(\hat{\mathbf{x}})=f(\phi(\hat{\mathbf{x}}))$ gives the negative life expectancy if we use metric units in $\hat{\mathbf{x}}$. If in metric units our initial choice for weight and height $\hat{\mathbf{x}}_1$ is the same as in the imperial case, $\phi(\hat{\mathbf{x}}_1)=\mathbf{x}_1$, then one would hope that also

$$\phi(\hat{\mathbf{x}}_2)=\mathbf{x}_2$$

when using gradient descent.
The following calculation shows that unfortunately this is not the case. For the i-th coordinate of $\mathbf{x}_2$ we have:

$$\mathbf{x}_{2i} = \mathbf{x}_{1i} + \alpha_1 \left.\frac{\partial f(\mathbf{x})}{\partial x_i}\right|_{\mathbf{x}=\mathbf{x}_1} .$$

and for $\hat{\mathbf{x}}_2$:

$$\begin{aligned}\hat{\mathbf{x}}_{2i} &= \hat{\mathbf{x}}_{1i} + \hat{\alpha}_1 \left.\frac{\partial \hat{f}(\hat{\mathbf{x}})}{\partial \hat{x}_i}\right|_{\hat{\mathbf{x}}=\hat{\mathbf{x}}_1}\\ &= \hat{\mathbf{x}}_{1i} + \hat{\alpha}_1 \left.\frac{\partial f(\phi(\hat{\mathbf{x}}))}{\partial \hat{x}_i}\right|_{\hat{\mathbf{x}}=\hat{\mathbf{x}}_1}\\ &= \hat{\mathbf{x}}_{1i} + \hat{\alpha}_1 \lambda_i \left.\frac{\partial f(\mathbf{x})}{\partial x_i}\right|_{\mathbf{x}=\phi(\hat{\mathbf{x}}_1)} .\end{aligned}$$

So, using $\phi(\hat{\mathbf{x}}_1) = \mathbf{x}_1$ and $\phi(\hat{\mathbf{x}})_i = \lambda_i \hat{\mathbf{x}}_i$, we find:

$$\phi(\hat{\mathbf{x}}_2)_i = \mathbf{x}_{1i} + \hat{\alpha}_1 \lambda_i^2 \left.\frac{\partial f(\mathbf{x})}{\partial x_i}\right|_{\mathbf{x}=\mathbf{x}_1} .$$

Consequently, to achieve $\phi(\hat{\mathbf{x}}_2) = \mathbf{x}_2$, would require $\alpha_1 = \hat{\alpha}_1 \lambda_i^2$ for $i = 1, 2$. But, since the conversion factors are different (and $\lambda_1^2 \neq \lambda_2^2$), this is impossible (unless $\alpha_1 = 0$).
The above discussion shows that the performance of gradient descent heavily depends on the the scaling of the input variables. In this context, it is worthwhile mentioning that in the literature gradient descent is sometimes called steepest descent. This is somewhat misleading since it seems to indicate that the negative gradient gives an exceptionally good direction for minimizing a function.
The name steepest descent derives from the fact, that among all directions, i.e. all vectors of a given small length, the direction of the gradient is the one in which the function changes most rapidly. This is indeed true, if one measures length using the conventional Euclidean norm $\sqrt{\mathbf{x}^T\mathbf{x}}$. However, the unit of measurement in the above example then is $\sqrt{\mathrm{lbs}^2 + \mathrm{ft}^2}$ in the imperial and $\sqrt{\mathrm{kg}^2 + \mathrm{m}^2}$ in the metric case. So one has different notions of what a length is, and no straightforward way to choose between them.
A definition of what length, or more generally the distance between two points is, is called a *metric*. Many of the more advanced optimization method can be considered as finding a metric which is tuned to the function being optimized and as performing 'steepest descent' w.r.t. to this metric. Except for the case of quadratic functions, the metric used depends on the current point, so these are *variable metric methods.*

5 Multivariate Minimization 2: Quadratic functions

The goal of this section is to derive efficient algorithms for minimizing multivariate quadratic functions. We shall begin by summarizing some properties of quadratic functions, and as byproduct obtain an efficient method for checking whether a symmetric matrix is positive definite.

5.1 Eigenvalues and Eigenvectors

Consider the quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$ and for the moment assume that A is not just symmetric but diagonal, i.e. $A_{ij} = 0$ for $i \neq j$. In this case it is easy to decide whether A is positive definite: Since $\mathbf{x}^T A\mathbf{x} = \sum_{i=1}^n A_{ii}x_i^2$, a diagonal matrix is positive definite if and only if $A_{ii} > 0$. Also $f(\mathbf{x})$ has the simple form

$$\begin{aligned}f(\mathbf{x}) &= \frac{1}{2}\sum_{i=1}^n A_{ii}x_i^2 - \sum_{i=1}^n b_i x_i + c\\ &= c + \sum_{i=1}^n (\frac{1}{2}A_{ii}x_i^2 - b_i x_i)\end{aligned}$$

and if A is positive definite:

$$\min_{\mathbf{x}} f(\mathbf{x}) = c + \sum_{i=1}^n \min_{x_i}(\frac{1}{2}A_{ii}x_i^2 - b_i x_i)$$

. For diagonal A we can decompose the multidimensional problem into n independent 1-dimensional problems and obtain the solution $x_i^* = b_i/A_{ii}$.
Note that A is diagonal if and only if $A\mathbf{e}_i = A_{ii}\mathbf{e}_i$, $i = 1, \ldots, n$ where $\mathbf{e}_i$ is the unit vector in the i-th direction. That is $\mathbf{e}_1 = (1, 0, \ldots, 0)^T$, $\mathbf{e}_2 = (0, 1, 0, \ldots, 0)^T$ etc.
If A is symmetric an important result from Linear Algebra tells us that we can find n linearly independent eigenvectors $\mathbf{v}_i$ with real eigenvalues λ_i:

$$A\mathbf{v}_i = \lambda_i \mathbf{v}_i \,.$$

Example 19 Eigenvectors of $A = \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix}$ are $(1,1)^T$, with eigenvalue $1+a$, and $(1,-1)^T$, with eigenvalue $1-a$. For $a = 0$ the two eigenvalues are the same and any nonzero vector is an eigenvector.

The special property of the diagonal case is that the eigenvectors coincide with the axis of the coordinate system. This suggests that for symmetric A we could simplify the minimization problem by a change of coordinates. This becomes specially simple if the eigenvectors v_i are orthogonal, and an important property of symmetric matrices is that we can always find such eigenvectors. To see this, rewrite the expression $\lambda_i \mathbf{v}_j^T \mathbf{v}_i$ in the following way:

$$\lambda_i \mathbf{v}_j^T \mathbf{v}_i = \mathbf{v}_j^T \lambda_i \mathbf{v}_i = \mathbf{v}_j^T A \mathbf{v}_i = \mathbf{v}_i^T A \mathbf{v}_j = \mathbf{v}_i^T \lambda_j \mathbf{v}_j = \lambda_j \mathbf{v}_i^T \mathbf{v}_j = \lambda_j \mathbf{v}_j^T \mathbf{v}_i$$

Comparing the first and last expression shows that $\mathbf{v}_j^T \mathbf{v}_i = 0$, if $\lambda_i \neq \lambda_j$. So, for a symmetric matrix, eigenvectors which have different eigenvalues are always orthogonal. Now assume that we encounter the special case $\lambda_i = \lambda_j = \lambda$. Since

$$A(s\mathbf{v}_i + t\mathbf{v}_j) = s\lambda\mathbf{v}_i + t\lambda\mathbf{v}_j = \lambda(s\mathbf{v}_i + t\mathbf{v}_j) \,,$$

any linear combination of $\mathbf{v}_i$ and $\mathbf{v}_j$ is an eigenvector as well. Thus we can the replace $\mathbf{v}_j$ by the eigenvector

$$\mathbf{v}_j' = \mathbf{v}_j - t\mathbf{v}_i; \quad t = \frac{\mathbf{v}_i^T \mathbf{v}_j}{\mathbf{v}_i^T \mathbf{v}_i}$$

and now $\mathbf{v}_i^T \mathbf{v}_j' = 0$. So if two eigenvalues are the same, we can nevertheless find orthogonal eigenvectors, and it is straightforward to extend the argument to the case that more than two eigenvalues coincide. In summary, a symmetric matrix always has n orthogonal eigenvectors, and since $\mathbf{v}_i/|\mathbf{v}_i|$ is an eigenvector as well, it even has n orthonormal eigenvectors.
From now on we assume that the $\mathbf{v}_i$ are such an orthonormal basis of eigenvectors and use the eigenvectors to construct the square matrix

$$R = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n) \,.$$

For the components of the product $R^T R$ we find $(R^T R)_{ij} = \mathbf{v}_i^T \mathbf{v}_j$, so all the off-diagonal elements of $R^T R$ are zero and the diagonal elements equal 1, i.e. $R^T R$ is the identity matrix. So $R^{-1} = R^T$, that is R is orthogonal.
Similarly ,

$$R^T A R = R^T (A\mathbf{v}_1, A\mathbf{v}_2, \ldots, A\mathbf{v}_n) = \begin{pmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_n^T \end{pmatrix} (\lambda_1 \mathbf{v}_1, \lambda_2 \mathbf{v}_2, \ldots, \lambda_N \mathbf{v}_n) \,,$$

so $(R^T A R)_{ij} = \lambda_j \mathbf{v}_i^T \mathbf{v}_j$ and

$$D = R^T A R$$

is diagonal. The diagonal elements of D are just the eigenvalues of A, $D_{ii} = \lambda_i$.
Now assume that D is not positive definite, that is we can find an $\mathbf{x} \neq \mathbf{0}$ such that $\mathbf{x}^T D \mathbf{x} \leq 0$. Then $\hat{\mathbf{x}} = R\mathbf{x}$ is nonzero as well since R is invertible. Further $\hat{\mathbf{x}}^T A \hat{\mathbf{x}} = \mathbf{x}^T D \mathbf{x} \leq 0$ and A is not positive definite. But, since $R^{-1} = R^T$, $RDR^T = A$. So one can turn the argument around and also show that if A is not positive definite, neither is D. Hence A is positive definite iff D is, that is iff all the eigenvalues of A are greater than zero.
Returning to minimization of

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x} + c$$

let us define the function

$$\hat{f}(\hat{\mathbf{x}}) = \frac{1}{2}\hat{\mathbf{x}}^T D\hat{\mathbf{x}} - (R^T\mathbf{b})^T\hat{\mathbf{x}} + c\,. \tag{7}$$

Since $\hat{f}(R^{-1}\mathbf{x}) = f(\mathbf{x})$, if $\hat{\mathbf{x}}^*$ is a minimum of $\hat{f}$, we find that $x^* = R\hat{\mathbf{x}}^*$ is a minimum of f. But minimizing $\hat{f}$ is easy since D is diagonal.

▶ **Exercise 5** *(5 marks) Show that if* $\mathbf{v}$ *is an eigenvector of the square matrix* A*, and* γ *is a real number different from zero, then* $\gamma\mathbf{v}$ *is also an eigenvector of* A*.*

▶ **Exercise 6** *(5 marks) Let* $f(\mathbf{x}) = \mathbf{x}^T A\mathbf{x}$*, where* A *is square but not necessarily symmetric. Show that* $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T(A + A^T)\mathbf{x}$ *and show that* $A + A^T$ *is symmetric.*

▶ **Exercise 7** *(5 marks) Given the matrix* $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ *find vectors* $\mathbf{x}$ *and* $\mathbf{y}$ *such that* $\mathbf{x}^T A\mathbf{y} \neq \mathbf{y}^T A\mathbf{x}$*. Would it still be possible to find such vectors if* A *were symmetric ?*

▶ **Exercise 8** *(10 marks) This is a MATLAB exercise.*
Consider the quadratic form

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}$$

with

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} 3 \\ 5 \end{pmatrix}$$

Implement the gradient descent optimization method for this function starting from the point $(-1, 2.5)$*, and make a plot showing the path taken by the gradient descent algorithm. [Hint: Since* f *is quadratic the result of the line search in the direction of the gradient can be calculated analytically and you do not have to implement a line search algorithm.] This plot should show that successive directions taken are orthogonal. Decide on an appropriate convergence (stopping) criterion. Hand in all of the code that you write. Also hand in a listing of the successive* $\mathbf{x}$ *positions found by the algorithm.*
Use the method with a starting location $\mathbf{x}^* + \mathbf{v}$*, where* $\mathbf{x}^*$ *is the minimum and* $\mathbf{v}$ *is an eigenvector of* A*.*
To add lines to the plot you can use something like the following:

```
hold on
plot( [0 1],[0 1] )
plot( [1 2],[1 0] )
hold off
```

Figures can be printed to a file by using the command `print -deps <filename>` *from the matlab command line. This will produce an encapsulated postscript file (with the suffix .eps) which can be easily printed.*

5.2 Minimising Quadratic functions using Line Search

Without loss of generality, consider

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c \tag{8}$$

where A is positive definite and symmetric. (If A is not symmetric, consider instead the symmetrised matrix $(A + A^T)/2$, which gives the same function f).
Although we know where the minimum of this function is, just using linear algebra, we wish to use this function as a toy model for more complex functions which however locally look approximately quadratic.
One approach to finding minima is to search along a particular direction $\mathbf{p}$, and find a minimum along this direction. We can then search for a deeper minima by looking in different directions. That is, we can search along a line $\mathbf{x}^* = \mathbf{x}^0 + \lambda\mathbf{p}$ such that the function attains a minimum. That is, the directional derivative is zero along this line,

$$0 = D_{\mathbf{p}}f(\mathbf{x}^*) = \nabla f(\mathbf{x}^*)\cdot\mathbf{p} \tag{9}$$

Differentiating equation (8), this gives

$$0 = (A\mathbf{x}^* - \mathbf{b}) \cdot \mathbf{p}$$
$$0 = (A\mathbf{x}0 + \lambda A\mathbf{p} - \mathbf{b}) \cdot \mathbf{p}$$

This has solution,

$$\lambda = \frac{(\mathbf{b} - A\mathbf{x}^0) \cdot \mathbf{p}}{\mathbf{p}^T A\mathbf{p}} \equiv \frac{-\nabla f(\mathbf{x}^0) \cdot \mathbf{p}}{\mathbf{p}^T A\mathbf{p}}$$

Ok, so now we've found the minimum along the line through $\mathbf{x}^0$ with direction $\mathbf{p}$. But how should we choose the line search direction $\mathbf{p}$? It would seem sensible to choose successive line search directions $\mathbf{p}$ according to $\mathbf{p}^{new} = -\nabla f(\mathbf{x}^*)$, so that each time we minimise the function along the line of steepest descent. However, this is far from the optimal choice in the case of minimising quadratic functions. A much better set of search directions are those defined by the vectors conjugate to A.

5.3 Conjugate vectors

So one way of solving the minimization problem would be to compute all of the eigenvectors of A. However, this is quite involved and it turns out that a simpler procedure is available. The starting point is the observation that the matrix R of the eigenvectors hast two properties: $R^{-1} = R^T$, and $R^T AR$ is diagonal. But we only needed the second property to define the transformation (7) which diagonalizes f.
Consequently we want to construct a matrix P with the property that $P^T AP$ is diagonal. Let $P = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_k)$, and note that we start with an n by k matrix, $k \leq n$. The reason for this is that we are aiming at an incremental procedure, where columns are successively added to P. Since $(P^T AP)_{ij} = \mathbf{p}_i^T A\mathbf{p}_j$ the matrix $P^T AP$ will be diagonal if $\mathbf{p}_i^T A\mathbf{p}_j = 0$ for $i \neq j$. It will be useful to require that $P^T AP$ has positive diagonal elements, i.e. $\mathbf{p}_i^T A\mathbf{p}_i > 0$.
We thus arrive at the following definition: The vectors $\mathbf{p}_i$, $i = 1, \dots, k$ are called *conjugate* to the matrix A, iff for $i, j = 1, \dots, k$ and $i \neq j$:

$$\mathbf{p}_i^T A\mathbf{p}_j = 0 \quad \text{and} \quad \mathbf{p}_i^T A\mathbf{p}_i > 0 \ .$$

The two conditions guarantee that conjugate vectors are linearly independent: Assume that

$$\mathbf{0} = \sum_{j=1}^{k} \alpha_j \mathbf{p}_j = \sum_{j=1}^{i-1} \alpha_j \mathbf{p}_j + \alpha_i \mathbf{p}_i + \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_j$$

Now multiplying from the left with $\mathbf{p}_i^T A$ yields $0 = \alpha_i \mathbf{p}_i^T A\mathbf{p}_i$. So α_i is zero since we know that $\mathbf{p}_i^T A\mathbf{p}_i > 0$. As we can make this argument for any $i = 1, \dots, k$, all of the α_i must be zero.
That conjugate vectors are linearly independent, means that the matrix $P = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_k)$ has full rank. In particular, if $k = n$, the matrix P will be invertible. An important consequence is: If the symmetric n by n matrix A has n conjugate directions, it is positive definite. To see this, let $\mathbf{x}$ be any nonzero vector. Then $\hat{\mathbf{x}} = P^{-1}\mathbf{x}$ is not zero, and since $P\hat{\mathbf{x}} = \mathbf{x}$:

$$\mathbf{x}^T A\mathbf{x} = \hat{\mathbf{x}}^T P^T AP\hat{\mathbf{x}} \ .$$

But since P is made of conjugate vectors, $P^T AP$ is diagonal and all the diagonal elements are positive. So $P^T AP$ is positive definite, $\hat{\mathbf{x}}^T P^T AP\hat{\mathbf{x}}$ is greater than zero, and thus $\mathbf{x}^T A\mathbf{x} > 0$.

▶ **Exercise 9** *(5 marks) For a symmetric n by n matrix A let $\mathbf{p}_1, \dots, \mathbf{p}_n$ be linearly independent vectors such that*

$$\mathbf{p}_i^T A\mathbf{p}_j = 0 \quad \textit{if} \quad i \neq j \ .$$

Show that A is positive definite if $\mathbf{p}_i^T A\mathbf{p}_i > 0$ for $i = 1, \dots, n$.

We now turn to constructing conjugate vectors using a procedure which is analoguous to Gram-Schmidt orthogonalization. Assume we already have k conjugate vectors $\mathbf{p}_1, \dots, \mathbf{p}_k$ and let $\mathbf{v}$ be a vector which is linearly independent of $\mathbf{p}_1, \dots, \mathbf{p}_k$. We then set

$$\mathbf{p}_{k+1} = \mathbf{v} - \sum_{j=1}^{k} \frac{\mathbf{p}_j^T A\mathbf{v}}{\mathbf{p}_j^T A\mathbf{p}_j} \mathbf{p}_j \tag{13}$$

and claim that now the vectors $\mathbf{p}_1, \dots, \mathbf{p}_{k+1}$ are conjugate if A is positive definite: Since $\mathbf{v}$ is linearly independent of $\mathbf{p}_1, \dots, \mathbf{p}_k$ the new vector $\mathbf{p}_{k+1}$ cannot be zero, and thus for positive definite A, $\mathbf{p}_{k+1}^T A \mathbf{p}_{k+1} > 0$. We now only need to show that $\mathbf{p}_i^T A \mathbf{p}_{k+1} = 0$. Indeed

$$\begin{aligned}
\mathbf{p}_i^T A \mathbf{p}_{k+1} &= \mathbf{p}_i^T A \mathbf{v} - \sum_{j=1}^{k} \frac{\mathbf{p}_j^T A \mathbf{v}}{\mathbf{p}_j^T A \mathbf{p}_j} \mathbf{p}_i^T A \mathbf{p}_j \\
&= \mathbf{p}_i^T A \mathbf{v} - \frac{\mathbf{p}_i^T A \mathbf{v}}{\mathbf{p}_i^T A \mathbf{p}_i} \mathbf{p}_i^T A \mathbf{p}_i \\
&= 0 .
\end{aligned}$$

An important property of the Gram-Schmidt procedure is that the vectors $\mathbf{p}_1, \dots, \mathbf{p}_{k+1}$ span the same subspace as the vectors $\mathbf{v}$ and $\mathbf{p}_1, \dots, \mathbf{p}_k$: Any linear combination of $\mathbf{p}_1, \dots, \mathbf{p}_{k+1}$ can be rewritten as a linear combination of $\mathbf{v}$ and $\mathbf{p}_1, \dots, \mathbf{p}_k$ just by replacing $\mathbf{p}_{k+1}$ with the RHS of equation (13). So the space spanned by $\mathbf{p}_1, \dots, \mathbf{p}_{k+1}$ is contained in the one spanned by $\mathbf{v}$ and $\mathbf{p}_1, \dots, \mathbf{p}_k$. But solving (13) for $\mathbf{v}$ allows us to turn the argument around and so the two subspaces are the same.
Using the Gram-Schmidt procedure, we can construct n conjugate vectors for a positive definite matrix in the following way. We start with n linearly independent vectors $\mathbf{u}_1, \dots, \mathbf{u}_n$, e.g. we might chose $\mathbf{u}_i = \mathbf{e}_i$, the unit vector in the i-th direction. We then set $\mathbf{p}_1 = \mathbf{u}_1$ and use (13) to compute $\mathbf{p}_2$ from $\mathbf{p}_1$ and $\mathbf{v} = \mathbf{u}_2$. Next we set $\mathbf{v} = \mathbf{u}_3$ and compute $\mathbf{p}_3$ from $\mathbf{p}_1, \mathbf{p}_2$ and $\mathbf{v}$. Continuing in this manner we obtain n conjugate vectors. Note that at each stage of the procedure the vectors $\mathbf{u}_1, \dots, \mathbf{u}_k$ span the same subspace as the vectors $\mathbf{p}_1, \dots, \mathbf{p}_k$.
What is going to happen if A is not positive definite? If we could find n conjugate vectors, A would be positive definite, and so at some point k the Gram-Schmidt procedure must break down. This will happen if $\mathbf{p}_k^T A \mathbf{p}_k \le 0$. So by trying out the Gram-Schmidt procedure, we can in fact find out whether a matrix is positive definite.

Example 20 For the matrix $A = \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix}$ we apply the Gram-Schmidt procedure to $\mathbf{u}_1 = (1,0)^T$ and $\mathbf{u}_2 = (0,1)^T$. So $\mathbf{p}_1 = (1,0)^T$ and $\mathbf{p}_1^T A \mathbf{p}_1 = 1$ is positive. Now

$$\mathbf{p}_2 = \mathbf{v} - \frac{\mathbf{p}_1^T A \mathbf{v}}{\mathbf{p}_1^T A \mathbf{p}_1} \mathbf{p}_1$$

for $\mathbf{v} = \mathbf{u}_2 = (0,1)^T$. So

$$\begin{aligned}
\mathbf{p}_2 &= \begin{pmatrix} 0 \\ 1 \end{pmatrix} - \left[(1,0) \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right] \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 1 \end{pmatrix} - a \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} -a \\ 1 \end{pmatrix} \qquad (14)
\end{aligned}$$

Now $\mathbf{p}_2^T A \mathbf{p}_2 = 1 - a^2$, so A is positive definite if $-1 < a < 1$.

▶ **Exercise 10** *(10 marks) Let $\{\mathbf{u}_i\}_{i=1}^n$ be a set of linearly independent vectors and for $k = 0, \dots, n-1$ set*

$$\mathbf{p}_{k+1} = \mathbf{u}_{k+1} - \sum_{j=1}^{k} \frac{\mathbf{u}_{k+1}^T A \mathbf{p}_j}{\mathbf{p}_j^T A \mathbf{p}_j} \mathbf{p}_j .$$

Then $\{\mathbf{p}_i\}_{i=1}^n$ are A-conjugate, and $\mathbf{p}_1, \dots, \mathbf{p}_k$ span the same subspace as $\mathbf{u}_1, \dots, \mathbf{u}_k$ if A is symmetric and positive definite.
Try out this Gram-Schmidt procedure on the following matrices.

$$\begin{pmatrix} 2 & -3 & 0 \\ -3 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 2 & -2 & 0 \\ -2 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Explain whether or not the Gram-Schmidt procedure determines if the matrix is postive definite.

▶ **Exercise 11** *(5 marks) For an n by n Matrix A and vectors*

$$\mathbf{e}_i = (\delta_{1i}, \delta_{2i}, \ldots, \delta_{ni})^T = (0, \ldots, 0, \underset{\uparrow \ i\text{-th position}}{1}, 0, \ldots, 0)^T$$

show that $\mathbf{e}_i^T A \mathbf{e}_j = A_{ij}$, *for* $i, j = 1, \ldots, n$. *In particular, A is diagonal if* $\mathbf{e}_i^T A \mathbf{e}_j = 0$ *for* $i \neq j$.

▶ **Exercise 12** *(20 marks)*
(a) State the conditions for a matrix A to be positive definite.

State the conditions for a set of vectors $\{\mathbf{p}_i, i = 1, \ldots, n\}$ *to be conjugate to the* $n \times n$ *matrix A.*
State the conditions for a set of vectors $\{\mathbf{u}_i, i = 1, \ldots, n\}$ *to be linearly independent.*
(b) You are given a set of linearly independent vectors $\{\mathbf{u}_i, i = 1, \ldots, n\}$ *and a postive definite matrix A. We wish to construct a set of vectors* $\{\mathbf{p}_i, i = 1, \ldots, n\}$ *conjugate to A.*
Without loss of generality, we may assume that $\mathbf{p}_1 = \mathbf{u}_1$. *Assuming that*

$$\mathbf{p}_2 - \mathbf{u}_1 + \lambda \mathbf{p}_1$$

find an appropriate value for λ *such that* $\{\mathbf{p}_1, \mathbf{p}_2\}$ *for a set conjugate to A.*
Assuming that, in general, we may write

$$\mathbf{p}_{k+1} = \mathbf{u}_{k+1} + \sum_{i=1}^{k} \lambda_i \mathbf{p}_i$$

find an appropriate set of values $\{\lambda_i, i = 1, \ldots, k\}$ *such that* $\{\mathbf{p}_1, \ldots \mathbf{p}_{k+1}\}$ *form a set conjugate to A. You may assume that* $\{\mathbf{p}_1, \ldots \mathbf{p}_k\}$ *form a set conjugate to A.*
(c) Show that if there exists a conjugate set $\{\mathbf{p}_1, \ldots \mathbf{p}_n\}$ *for the* $n \times n$ *matrix A, then A is positive definite. Hint: You may assume that any vector can be expressed as a linear combination of all the conjugate vectors.*

5.4 The conjugate vectors algorithm

Let us assume that when minimizing $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x} + c$ we first construct n vectors $\mathbf{p}_1, \ldots, \mathbf{p}_n$ conjugate to A which we use as our search directions. So

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \ . \tag{15}$$

At each step we chose α_k by an exact line search, thus

$$\alpha_k = -\frac{\mathbf{p}_k^T \mathbf{g}_k}{\mathbf{p}_k^T A \mathbf{p}_k} \ . \tag{16}$$

We call this procedure the conjugate vectors algorithm, and the algorithm has a nice geometrical interpretation:
Theorem (Luenberger) The Expanding Subspace Theorem.
Let $\{\mathbf{p}_i\}_{i=1}^n$ be a sequence of vectors in $\mathbb{R}^n$ conjugate to the (positive definite) matrix A and $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x} + c$. Then for any $\mathbf{x}_1$ the sequence $\{\mathbf{x}_k\}$ generated according to (15) and (16) has the property that directional derivative of f in the direction $\mathbf{p}_i$ vanishes at the point $\mathbf{x}_{k+1}$ if $i \leq k$; i.e. $D_{\mathbf{p}_i} f(\mathbf{x}_{k+1}) = 0$. (That is, not only is the directional derivative zero at the new point along the direction $\mathbf{p}_k$, it is zero along all the previous search directions $\mathbf{p}_1, \ldots, \mathbf{p}_k$).
Proof: For $i \leq k$, we can write $\mathbf{x}_{k+1}$ as:

$$\mathbf{x}_{k+1} = \mathbf{x}_{i+1} + \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_j \ .$$

Since $\nabla f(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$ we have

$$\nabla f(\mathbf{x}_{k+1}) = A\mathbf{x}_{k+1} - \mathbf{b} = A(\mathbf{x}_{i+1} + \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_j) - \mathbf{b} = A\mathbf{x}_{i+1} - \mathbf{b} + A \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_j = \nabla f(\mathbf{x}_{i+1}) + \sum_{j=i+1}^{k} \alpha_j A \mathbf{p}_j \ .$$

So

$$\begin{aligned}
D_{\mathbf{p}_i} f(\mathbf{x}_{k+1}) &= \nabla f^T(\mathbf{x}_{k+1})\mathbf{p}_i = \mathbf{p}_i^T \nabla f(\mathbf{x}_{k+1}) \\
&= \mathbf{p}_i^T \nabla f(\mathbf{x}_{i+1}) + \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_i^T A \mathbf{p}_j \\
&= (D_{\mathbf{p}_i} f)(\mathbf{x}_{i+1}) + \sum_{j=i+1}^{k} \alpha_j \mathbf{p}_i^T A \mathbf{p}_j
\end{aligned}$$

Now $(D_{\mathbf{x}_{i+1}} f)(\mathbf{p}_i) = 0$ since the point $\mathbf{x}_{i+1}$ was obtained by an exact line search in the direction $\mathbf{p}_i$. But all of the terms in the sum over j also vanish since $j > i$ and $\mathbf{p}_i^T A \mathbf{p}_j = 0$ by conjugacy. So $(D_{\mathbf{p}_i f})(\mathbf{x}_{k+1}) = 0$.
The subspace theorem shows, that because we use conjugate vectors, optimizing in the direction $\mathbf{p}_k$, does not spoil the optimality w.r.t. to the previous search directions. In particular after having carried out n steps of the algorithm we have $(D_{\mathbf{x}_{n+1}} f)(\mathbf{p}_i) = \nabla f^T(\mathbf{x}_{n+1})\mathbf{p}_i = 0$, for $i = 1, \dots, n$. The n equations can be written in a more compact form as:

$$\nabla f^T(\mathbf{x}_{n+1})(\mathbf{p}_1, \mathbf{p}_2, \dots \mathbf{p}_n) = \mathbf{0}\ .$$

But the square matrix $P = (\mathbf{p}_1, \mathbf{p}_2, \dots \mathbf{p}_n)$ is invertible since the $\mathbf{p}_i$ are conjugate, so we in fact have $\nabla f(\mathbf{x}_{n+1}) = \mathbf{0}$: The point $\mathbf{x}_{n+1}$ is the minimum $\mathbf{x}^*$ of the quadratic function f. So in contrast to gradient descent, for a quadratic function the conjugate vectors algorithm converges in a finite number of steps.

5.5 The conjugate gradients algorithm

The conjugate gradients algorithm is a special case of the conjugate vectors algorithm, in which the Gram-Schmidt procedure becomes very simple. We do not use a predetermined set of conjugate vectors but construct these "on-the-fly".
After k-steps of the conjugate vectors algorithm we need to construct a vector $\mathbf{p}_{k+1}$ which is conjugate to $\mathbf{p}_1, \dots, \mathbf{p}_k$. This could be done by applying the Gram-Schmidt procedure to any vector $\mathbf{v}$ which is linearly independent of the vectors $\mathbf{p}_1, \dots, \mathbf{p}_k$. In the conjugate gradients algorithm one makes the special choice

$$\mathbf{v} = -\nabla f(\mathbf{x}_{k+1})\,.$$

By the subspace theorem the gradient at the new point $\mathbf{x}_{k+1}$ is orthogonal to $\mathbf{p}_i$, $i = 1, \dots, k$. So $\nabla f(\mathbf{x}_{k+1})$ is linearly independent of $\mathbf{p}_1, \dots, \mathbf{p}_k$ and a valid choice for $\mathbf{v}$, unless $\nabla f(\mathbf{x}_{k+1}) = \mathbf{0}$. In the latter case $\mathbf{x}_{k+1}$ is our minimum and we are done, and from now on we assume that $\nabla f(\mathbf{x}_{k+1}) \neq \mathbf{0}$. Using the notation $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, the equation for the new search direction given by the Gram-Schmidt procedure is:

$$\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \sum_{i=1}^{k} \frac{\mathbf{p}_i^T A \mathbf{g}_{k+1}}{\mathbf{p}_i^T A \mathbf{p}_i} \mathbf{p}_i\,. \tag{17}$$

Since $\mathbf{g}_{k+1}$ is orthogonal to $\mathbf{p}_i$, $i = 1, .., k$, by the subspace theorem we have $\mathbf{p}_{k+1}^T \mathbf{g}_{k+1} = -\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}$. So α_{k+1} can be written as

$$\alpha_{k+1} = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{p}_{k+1}^T A \mathbf{p}_{k+1}}\ , \tag{18}$$

and in particular $\alpha_{k+1} \neq 0$.
We now want to show that because we have been using the conjugate gradients algorithm at the previous steps as well, in equation (17) all terms but the last in the sum over i vanish. We shall assume that $k > 0$ since in the first step ($k = 0$) we just set $\mathbf{p}_1 = -\mathbf{g}_1$.
First note that

$$\mathbf{g}_{i+1} - \mathbf{g}_i = A\mathbf{x}_{i+1} - \mathbf{b} - (A\mathbf{x}_i - \mathbf{b}) = A(\mathbf{x}_{i+1} - \mathbf{x}_i) = \alpha_i A \mathbf{p}_i$$

and since $\alpha_i \neq 0$:

$$A\mathbf{p}_i = (\mathbf{g}_{i+1} - \mathbf{g}_i)/\alpha_i.$$

So in equation (17):

$$\mathbf{p}_i^T A \mathbf{g}_{k+1} = \mathbf{g}_{k+1}^T A \mathbf{p}_i = \mathbf{g}_{k+1}^T(\mathbf{g}_{i+1} - \mathbf{g}_i)/\alpha_i = (\mathbf{g}_{k+1}^T \mathbf{g}_{i+1} - \mathbf{g}_{k+1}^T \mathbf{g}_i)/\alpha_i$$

Since the $\mathbf{p}_i$ where obtained by applying the Gram-Schmidt procedure to the gradients $\mathbf{g}_i$, the subspace theorem $\mathbf{g}_{k+1}^T\mathbf{p}_i = 0$, implies, also $\mathbf{g}_{k+1}^T\mathbf{g}_i = 0$ for $i = 1, .., k$. This shows that

$$\mathbf{p}_i^T A\mathbf{g}_{k+1} = (\mathbf{g}_{k+1}^T\mathbf{g}_{i+1} - \mathbf{g}_{k+1}^T\mathbf{g}_i)/\alpha_i = \begin{cases} 0 & \text{if } 1 \le i < k \\ \mathbf{g}_{k+1}^T\mathbf{g}_{k+1}/\alpha_k & \text{if } i = k \end{cases}$$

Hence equation (17) simplifies to

$$\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \frac{\mathbf{g}_{k+1}^T\mathbf{g}_{k+1}/\alpha_k}{\mathbf{p}_k^T A\mathbf{p}_k}\mathbf{p}_k \,.$$

This can be brought into an even simpler form by applying equation (18) to α_k:

$$\begin{aligned} \mathbf{p}_{k+1} &= -\mathbf{g}_{k+1} + \frac{\mathbf{g}_{k+1}^T\mathbf{g}_{k+1}}{\mathbf{p}_k^T A\mathbf{p}_k}\frac{\mathbf{p}_k^T A\mathbf{p}_k}{\mathbf{g}_k^T\mathbf{g}_k}\mathbf{p}_k \\ &= -\mathbf{g}_{k+1} + \frac{\mathbf{g}_{k+1}^T\mathbf{g}_{k+1}}{\mathbf{g}_k^T\mathbf{g}_k}\mathbf{p}_k \,. \end{aligned}$$

We shall write this in the form

$$\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \beta_k\mathbf{p}_k \quad \text{where } \beta_k = \frac{\mathbf{g}_{k+1}^T\mathbf{g}_{k+1}}{\mathbf{g}_k^T\mathbf{g}_k} \,. \tag{19}$$

So using (18) and (19), the conjugate gradients algorithm is:

```
0cm
k = 1
input $\mathbf{x}_1$
$\mathbf{p}_1 = -\mathbf{g}_1$
while $\mathbf{g}_k \ne \mathbf{0}$
  $\alpha_k := -\mathbf{g}_k^T\mathbf{p}_k/(\mathbf{p}_k^T A\mathbf{p}_k)$
  $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k\mathbf{p}_k$
  $\beta_k := \mathbf{g}_{k+1}^T\mathbf{g}_{k+1}/(\mathbf{g}_k^T\mathbf{g}_k)$
  $\mathbf{p}_{k+1} := -\mathbf{g}_{k+1} + \beta_k\mathbf{p}_k$
  k = k +1
end
```

We have shown that the simple formula (19) yields conjugate vectors and that the following theorem holds:
Theorem (Fletcher) For $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c$, A a positive definite n by n matrix, the above algorithm terminates after $k \le n$ iteration and $\mathbf{x}_{k+1}$ is the minimum of f. The search directions $\mathbf{p}_1, \dots, \mathbf{p}_k$ are conjugate and the gradients $\mathbf{g}_1, \dots, \mathbf{g}_k$ are orthogonal.
In a practical implementation of the algorithm the test for completion, $\mathbf{g}_k \ne \mathbf{0}$, would have to be replaced by something like $\mathbf{g}_k^T\mathbf{g}_k >$ tol.
The formula (19) for β_k is due to Fletcher and Reeves. Since the gradients are orthogonal, β_k can also be written as

$$\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T\mathbf{g}_k} \,,$$

this is the Polak-Ribiere formula. The choice between the two expression for β_k can be of some importance if f is not quadratic.

▶ **Exercise 13** *(10 marks) For the quadratic objective function*

$$f(\mathbf{x}) = 2x_1^2 + x_2^2/2 + 5x_3^2/2 + x_1x_2 - 2x_1x_3 - 8x_1 - 3x_2 + 7x_3$$

the method of conjugate gradients, starting at $\mathbf{x}_1^T = (0, 0, 0)$ *proceeds via* $\mathbf{x}_2^T = (1.248, 0.468, -1.092)$ *and* $\mathbf{x}_3^T = (1.491, 1.084, -0.726)$. *Verify that* $\mathbf{g}_3^T\mathbf{p}_2 = \mathbf{g}_3^T\mathbf{p}_1 = \mathbf{g}_2^T\mathbf{p}_1 = 0$. *Complete the minimization by finding* $\mathbf{x}_4$, *and verify that this is indeed the minimum* $\mathbf{x}^*$.

▶ **Exercise 14** *(10 marks) For the quadratic function*

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + c \,,$$

where A is positive definite, consider the change of variables $\mathbf{x} = \phi(\hat{\mathbf{x}})$ where for some some non-singular matrix S and constant vector $\mathbf{d}$

$$\phi(\hat{\mathbf{x}}) = S\hat{\mathbf{x}} + \mathbf{d} \quad \textit{or equivalently} \quad \phi^{-1}(\mathbf{x}) = S^{-1}(\mathbf{x} - \mathbf{d}) .$$

Setting $\hat{f}(\hat{\mathbf{x}}) = f(\phi(\hat{\mathbf{x}}))$ we have that $\hat{f}(\hat{\mathbf{x}}) = f(\mathbf{x})$ if $\mathbf{x} = \phi(\hat{\mathbf{x}})$. Show that

$$\nabla \hat{f}(\hat{\mathbf{x}}) = S^T \nabla f(\mathbf{x}) ,$$

if $\mathbf{x} = \phi(\hat{\mathbf{x}})$.
Let $\mathbf{x}_1, \mathbf{x}_2, \ldots$ be a sequence of points obtained by gradient descent in f, and $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots$ a sequence of points obtained by gradient descent in $\hat{f}$. So

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \quad \textit{and} \quad \hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k - \hat{\alpha}_k \hat{\mathbf{g}}_k$$

where

$$\mathbf{g}_k = \nabla f(\mathbf{x}) \quad \textit{and} \quad \hat{\mathbf{g}}_k = \hat{f}(\hat{\mathbf{x}}_k)$$

and α_k is obtained by an exact line search in f, while $\hat{\alpha}_k$ is obtained by an exact line search in $\hat{f}$.
Show that gradient descent is invariant to the change of coordinates ϕ if S is an orthogonal matrix. That is if S is orthogonal and $\mathbf{x}_1 = \phi(\hat{\mathbf{x}}_1)$ then for all k we have $\mathbf{x}_k = \phi(\hat{\mathbf{x}}_k)$.
In fact the invariance property is not limited to quadratic functions. Show that for any differentiable function f

$$\nabla \hat{f}(\hat{\mathbf{x}}) = S^T \nabla f(\mathbf{x})$$

if $\mathbf{x} = \phi(\hat{\mathbf{x}})$. For S orthogonal, give an argument as to why the line searches to find α_k and $\hat{\alpha}_k$ should give the same answer in both coordinate systems, and hence show that gradient descent is invariant to the transformation ϕ for all orthogonal S.

5.6 Quasi-Newton methods

We have seen that for a quadratic function $f(\mathbf{x}) = 1/2\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}$, the minimum occurs at $\mathbf{x}^* = A^{-1}\mathbf{b}$. Using the local information $\mathbf{g}(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$, we can rewrite this as

$$\mathbf{x}^* = \mathbf{x} - A^{-1}\mathbf{g}(\mathbf{x}) \tag{20}$$

This can also be seen as Newton's root-finding method applied to ∇f. Thus if we are prepared to do a $n \times n$ matrix inversion we can find the optimum $\mathbf{x}^*$ in one step. However, for large-scale problems (when the dimension of $\mathbf{x}$ may be $O(100)$) this inversion is computationally demanding, especially if the matrix is singular or nearly so.
An alternative is to set up the iteration

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k S_k \mathbf{g}_k. \tag{21}$$

This is a very general form; if $S_k = A^{-1}$ then we have Newton's method, while if $S_k = I$ we have steepest descent. In general it would seem to be a good idea to choose S_k to be an approximation to the inverse Hessian. Also note that it is important that S_k be positive definite so that for small α_k we obtain a descent method.
The idea behind most quasi-Newton methods is to try to construct an approximate inverse Hessian H_k using information gathered as the descent progresses, and to set $S_k = H_k$. As we have seen, for a quadratic optimization problem we have the relationship

$$\mathbf{g}_{k+1} - \mathbf{g}_k = A(\mathbf{x}_{k+1} - \mathbf{x}_k) \tag{22}$$

Defining

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k \quad \text{and} \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \tag{23}$$

we see that equation 22 becomes

$$\mathbf{y}_k = A\mathbf{s}_k \tag{24}$$

It is reasonable to demand that

$$H_{k+1}\mathbf{y}_i = \mathbf{s}_i \qquad 1 \le i \le k \tag{25}$$

After n linearly independent steps we would then have $H_{n+1} = A^{-1}$. For $k < n$ there are an infinity of solutions for H_{k+1} satisfying equation 25. We shall focus on one of the most popular, the Broyden-Fletcher-Goldfarb-Shanno (or BFGS) update. This is given by

$$H_{k+1} = H_k + \left(1 + \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k}\right) \frac{\mathbf{s}_k \mathbf{s}_k^T}{\mathbf{s}_k^T \mathbf{y}_k} - \frac{\mathbf{s}_k \mathbf{y}_k^T H_k + H_k \mathbf{y}_k \mathbf{s}_k^T}{\mathbf{s}_k^T \mathbf{y}_k} \tag{26}$$

Note that this is a rank-2 correction to H_k constructed from the vectors $\mathbf{s}_k$ and $H_k\mathbf{y}_k$.

The quasi-Newton algorithm is

```
0cm
input $\mathbf{x}_1$
set $H_1 = I$
for $k = 1, \dots n$
  $\mathbf{p}_k = -H_k \mathbf{g}_k$
  compute $\alpha_k$ to minimize $f(\mathbf{x}_k + \alpha \mathbf{p}_k)$
  $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, and update $H_{k+1}$
  k = k +1
end
```

There are a number of properties of the BFGS update

A. All the H's are symmetric. This is proved immediately as $H_1 = I$ is symmetric, and each update is symmetric.

B. All the H's are positive definite. This proof is quite involved and is omitted.

C. The direction vectors $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_k$ produced by the algorithm obey

$$\mathbf{p}_i^T A \mathbf{p}_j = 0 \qquad 1 \le i < j \le k \tag{27}$$

$$H_{k+1} A \mathbf{p}_i = \mathbf{p}_i \qquad 1 \le i \le k \tag{28}$$

Equation 28 is called the hereditary property. In our notation $\mathbf{s}_k = \alpha_k \mathbf{p}_k$, and as the α's are non-zero, equation 27 can also be written as

$$\mathbf{s}_i^T A \mathbf{s}_j = 0 \qquad 1 \le i < j \le k \tag{29}$$

Before proving that these properties hold, we first note that

$$H_{k+1}\mathbf{y}_k = \mathbf{s}_k \tag{30}$$

for all $k \ge 1$. We show this using the definition of H_{k+1}, thus

$$H_{k+1}\mathbf{y}_k = H_k \mathbf{y}_k + \left(1 + \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k}\right) \frac{\mathbf{s}_k (\mathbf{s}_k^T \mathbf{y}_k)}{\mathbf{s}_k^T \mathbf{y}_k} - \frac{\mathbf{s}_k (\mathbf{y}_k^T H_k \mathbf{y}_k) + H_k \mathbf{y}_k \mathbf{s}_k^T \mathbf{y}_k}{\mathbf{s}_k^T \mathbf{y}_k} \tag{31}$$

$$= H_k \mathbf{y}_k + \left(1 + \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k}\right) \mathbf{s}_k - \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{s}_k^T \mathbf{y}_k} \mathbf{s}_k - H_k \mathbf{y}_k \tag{32}$$

$$= \mathbf{s}_k \tag{33}$$

The proof is by induction, with the base case taken as $k = 2$. As a special case of equation 30, we have that $H_2 \mathbf{y}_1 = \mathbf{s}_1$. Also

$$\mathbf{p}_2^T A \mathbf{p}_1 = -\mathbf{g}_2^T H_2 A \mathbf{p}_1 \tag{34}$$

$$= -\mathbf{g}_2 \mathbf{p}_1 \tag{35}$$

$$= 0 \qquad \text{by the exact line search condition} \tag{36}$$

We now assume that equations 27, 28 and 29 hold up to k, and prove that they hold for $k+1$. We have already proved that $H_{k+1}\mathbf{y}_k = \mathbf{s}_k$. Now consider (for some $i < k$)

$$\begin{aligned} H_{k+1}\mathbf{y}_i &= H_k\mathbf{y}_i + \left(1 + \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k}\right)\frac{\mathbf{s}_k\mathbf{s}_k^T\mathbf{y}_i}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{\mathbf{s}_k\mathbf{y}_k^T H_k\mathbf{y}_i + H_k\mathbf{y}_k\mathbf{s}_k^T\mathbf{y}_i}{\mathbf{s}_k^T\mathbf{y}_k} && (37)\\ &= \mathbf{s}_i + \left(1 + \frac{\mathbf{y}_k^T H_k \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k}\right)\frac{\mathbf{s}_k\mathbf{s}_k^T A\mathbf{s}_i}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{\mathbf{s}_k\mathbf{y}_s^T A\mathbf{s}_i + H_k\mathbf{y}_k\mathbf{s}_k^T A\mathbf{s}_i}{\mathbf{s}_k^T\mathbf{y}_k} && (38)\\ &= \mathbf{s}_i && (39) \end{aligned}$$

using $\mathbf{y}_i = A\mathbf{s}_i$, $\mathbf{y}_k^T = \mathbf{s}_k^T A$ and $H_k\mathbf{y}_i = \mathbf{s}_i$.
Now consider the conjugacy relationship. For k we have

$$\begin{aligned} \mathbf{p}_{k+1}^T A\mathbf{p}_k &= -\mathbf{g}_{k+1} H_{k+1} A\mathbf{p}_k && (40)\\ &= -\mathbf{g}_{k+1}\mathbf{p}_k && (41)\\ &= 0 \quad \text{by the exact line search condition} && (42) \end{aligned}$$

and for $i < k$

$$\begin{aligned} \mathbf{p}_{k+1}^T A\mathbf{p}_i &= -\mathbf{g}_{k+1} H_{k+1} A\mathbf{p}_i && (43)\\ &= -\mathbf{g}_{k+1}\mathbf{p}_i && (44)\\ &= 0 && (45) \end{aligned}$$

This last equality is due to the expanding subspace theorem. QED.
Since the $\mathbf{p}_k$'s are A-conjugate and since we successively minimize f in these directions, we see that the BFGS algorithm is a conjugate direction method; with the choice of $H_1 = I$ it is in fact the conjugate gradient method.

The BFGS update is only one out of a large range of updates that be considered. An earlier on was the Davidon-Fletcher-Powell (or DFP) update; however, this has some disadvantages in practice and so the BFGS update is preferred.
BFGS tends to be better than CG methods for small problems. Prefer to use CG for large problems, with more than 1000 variables. Note that the storage requirements for Quasi Newton methods scale quadratically with the number of variables, whilst only linearly for Conjugate Gradient methods.

6 Multivariate Minimization 3: General Functions

6.1 Use of conjugate gradient and quasi-Newton methods

The basic strategy for the minimization of general (i.e. non-quadratic) functions is to apply the same algorithms as in the quadratic case, except that exact line-minimizations (the calculation of the α_k's) have to be replaced with inexact line-searches, for example using Brent's algorithm. Of course it is much more difficult to perform analysis in this general case.
An issue that arises is the question of "restarting", meaning re-initializing the search direction to the negative gradient after n iterations. This can be applied to both conjugate gradient (CG) and quasi-Newton (QN) methods; in the latter case the approximate inverse Hessian would be reinitialized to the identity matrix. For example, if the algorithm progresses from a non-quadratic region into the neighbourhood of the solution where $f(\mathbf{x})$ is closely approximated by a quadratic, a reset method can be expected to converge rapidly, while a non-reset one may not do so.
For the conjugate gradient method, the exact formula used to compute β_k may be important in this respect. For example, two possible formulae are

$$\beta_k = \frac{\mathbf{g}_{k+1}^T\mathbf{g}_{k+1}}{\mathbf{g}_k^T\mathbf{g}_k} \qquad \text{Fletcher-Reeves} \qquad (46)$$

$$\beta_k = \frac{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T\mathbf{g}_{k+1}}{\mathbf{g}_k^T\mathbf{g}_k} \qquad \text{Polak-Ribiere.} \qquad (47)$$

These formulae are equivalent for quadratic functions, as can be seen using equation 3.30. However, they can have quite different effects in the non-quadratic case. If the algorithm is not making progress then we

can expect $\mathbf{g}_{k+1} \simeq \mathbf{g}_k$, and hence that β_k computed by the Polak-Ribiere formula will be near 0, and hence that $\mathbf{p}_{k+1} \simeq -\mathbf{g}_{k+1}$. On the other hand, under the same circumstances the Fletcher-Reeves update will set $\beta_k \simeq 1$.
For general functions it can be shown (Fletcher, §3.2) that the BFGS quasi-Newton method:

(i) preserves positive definite matrices H_k, and hence that the descent property holds;

(ii) requires $O(n^2)$ multiplications per iteration;

(iii) has superlinear order of convergence;

(iv) has global convergence for strictly convex functions (with exact line searches)

A comparison of the conjugate gradient and quasi-Newton algorithms shows that CG methods require only $O(n)$ storage, as opposed to $O(n^2)$ for QN methods. Typically this means that QN methods are preferred for problems of relatively small size, while CG methods are more suitable for very large problems (say $O(1000)$ variables). QN methods appear to be more tolerant of moderate precision line-searches compared to CG methods, and Fletcher's conclusion is that CG methods are less efficient and less robust than QN methods on problems to which they can both be applied.

▶ **Exercise 15** *(10 marks)*
Try to find vectors conjugate to the matrix

$$A = \begin{pmatrix} 1 & a & 0 \\ a & 1 & a \\ 0 & a & 1 \end{pmatrix}$$

by applying the Gram-Schmidt procedure to the vectors $\mathbf{e}_1 = (1,0,0)^T$, $\mathbf{e}_2 = (0,1,0)^T$ *and* $\mathbf{e}_3 = (0,0,1)^T$. *For which values of a is A positive definite?*

7 Optimization with constraints

The methods treated so far have been concerned with minimizing a function f either locally or globally in a Euclidean space $\mathbb{R}^n$. However, many practical optimization problems place constraints on the search space. These can be of two types:

- Equality (type A). $c(\mathbf{x}) = 0$. These reduce the dimension of the search space.
- Inequality (type B). $c(\mathbf{x}) \geq 0$ or $c(\mathbf{x}) > 0$. These reduce the 'volume' of the search space, but do not affect the dimension.

Unconstrained optimization techniques cannot be applied directly to such problems (although stochastic methods can be used if the random steps can be constrained to the right region). When f and all the constraints are linear, specialised linear programming methods are available that reliably converge to the global optimum. However, these are complicated to describe and do not generalize, so will not be touched on further here.

7.1 Transformation Methods

For some class A constraints, it is possible to solve for one variable x_i in terms of the other coordinates of $\mathbf{x}$. If this can be done, then x_i can be eliminated from f, and the whole problem reduced to one of lower dimension with fewer constraints. This is always advantageous if it can be carried out.
Some class B constraints can be removed by simple variable substitutions so that they are automatically satisfied. The choice of substitution may depend on how easy it is to calculate relevant derivatives after the substitution has been made. For example, $x_i \geq 0$ can be removed by substituting $x_i = y_i^2$ or $x_i = \exp y_i$, while $a < x_i < b$ can be removed by substituting

$$x_i = \frac{a+b}{2} + \frac{a-b}{2} \tanh y_i.$$

Another useful substitution in statistics is the 'softmax' function. Minimizing $f(\mathbf{x})$ subject to the constraint

$$\sum_{i=1}^{n} x_i = 1, \text{ where } 0 \leq x_i \leq 1 \quad \text{for } i = 1, \ldots, n$$

can be transformed to minimizing f with

$$x_i = \frac{\exp y_i}{\sum_{j=1}^{n} \exp y_j}$$

where the y_i's are now unconstrained.
Note that these substitutions can complicate the problem by introducing extra local minima and critical points which, although predictable, could be found by the optimization routine. For example, consider the problem of minimizing $(x-1)^2$ subject to $x > 0$. This is minimized with $x = 1$. If we set $x = y^2$, then we obtain an (unconstrained) minimization problem with function $(y^2-1)^2$. There are now two global optima at $y = \pm 1$ corresponding to $x = 1$, and a new critical point at $y = 0$.

7.2 Lagrange multipliers

7.2.1 Single Constraint

Consider first the problem of minimising $f(\mathbf{x})$ subject to a single constraint $c(\mathbf{x}) = 0$.
Imagine that we have already identified an $\mathbf{x}$ that satisfies the constraint, that is $c(\mathbf{x}) = 0$. How can we tell if this $\mathbf{x}$ minimises the function f? We are only allowed to search for lower function values around this $\mathbf{x}$ in directions which are consistent with the constraint. That is, $c(\mathbf{x} + \boldsymbol{\delta}) = 0$.

$$c(\mathbf{x} + \boldsymbol{\delta}) \approx c(\mathbf{x}) + \boldsymbol{\delta} \cdot \nabla c(\mathbf{x})$$

Hence, in order that the constraint remains satisfies, we can only search in a direction such that $\boldsymbol{\delta} \cdot \nabla c(\mathbf{x}) = \mathbf{0}$, that is in directions $\boldsymbol{\delta}$ that are orthogonal to $\nabla c(\mathbf{x})$. So, lets explore the change in f along a direction $\mathbf{n}$ where $\mathbf{n} \cdot \nabla c(\mathbf{x}) = 0$,

$$f(\mathbf{x} + \epsilon \mathbf{n}) \approx f(\mathbf{x}) + \epsilon \nabla f(\mathbf{x}) \cdot \mathbf{n}.$$

Since we are looking for a point $\mathbf{x}$ that minimises the function f, we require $\mathbf{x}$ to be a stationary point, $\nabla f(\mathbf{x}) \cdot \mathbf{n} = 0$. That is, $\nabla f(\mathbf{x})$ must lie parallel to $\nabla c(\mathbf{x})$, so that

$$\nabla f(\mathbf{x}) = \lambda \nabla c(\mathbf{x})$$

for some $\lambda \in \mathbb{R}$. To solve the optimisation problem therefore, we look for a point $\mathbf{x}$ such that $\nabla f(\mathbf{x}) = \lambda \nabla c(\mathbf{x})$, for some λ, and for which $c(\mathbf{x}) = 0$. An alternative formulation of this dual requirement is to look for $\mathbf{x}$ and λ that jointly minimise the *Lagrangian*

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda c(\mathbf{x})$$

Differentiating with respect to $\mathbf{x}$, we get the requirement $\nabla f(\mathbf{x}) = \lambda \nabla c(\mathbf{x})$, and differentiaing with respect to λ, we get that $c(\mathbf{x}) = 0$.

7.2.2 Multiple Constraints

Consider the problem of optimizing $f(\mathbf{x})$ subject to the constraints $c_i(\mathbf{x}) = 0$, $i = 1, \ldots, r < n$, where n is the dimensionality of the space. Denote by S the $n - r$ dimensional subspace of $\mathbf{x}$ which obeys the constraints. Assume that $\mathbf{x}^*$ is such an optimum. As in the unconstrained case, we consider perturbations $\mathbf{v}$ to $\mathbf{x}^*$, but now such that $\mathbf{v}$ lies in S

$$c_i(\mathbf{x}^* + h\mathbf{v}) = c_i(\mathbf{x}^*) + \mathbf{v}^T \nabla c_i(\mathbf{x}^*) + O(h^2) \tag{48}$$

Let $\mathbf{a}_i^* = \nabla c_i(\mathbf{x}*)$. Thus for the the perturbation to stay within S, we require that $\mathbf{v}^T \mathbf{a}_i^* = 0$ for all $i = 1, \ldots r$. Let A^* be the matrix whose columns are $\mathbf{a}_1^*, \mathbf{a}_2^*, \ldots \mathbf{a}_r^*$. Then this condition can be rewritten as $A^* \mathbf{v} = \mathbf{0}$.
We also require for a local optimum that $\mathbf{v}^T \nabla f = 0$ for all $\mathbf{v}$ in S. We see that ∇f must be orthogonal to $\mathbf{v}$, and that $\mathbf{v}$ must be orthogonal to the $\mathbf{a}_i^*$'s. Thus ∇f must be a linear combination of the $\mathbf{a}_i^*$'s, i.e.

$$\nabla f = \sum_{i=1}^{r} \lambda_i^* \mathbf{a}_i^* = A^{*T} \boldsymbol{\lambda}^* \tag{49}$$

Geometrically this says that the gradient vector is normal to the tangent plane to S at $\mathbf{x}^*$.

[If this argument is too heuristic, a more formal proof is now given by contradiction. Let $\nabla f = \sum_i \lambda_i^* \mathbf{a}_i^* + \mathbf{d}$, where $\mathbf{d} \neq \mathbf{0}$ is orthogonal to the $\mathbf{a}_i^*$'s, i.e. $A^{*T}\mathbf{d} = \mathbf{0}$. In this case taking $\mathbf{v} = -\mathbf{d}$ would violate $\mathbf{v}^T\nabla f = 0$, and hence we would not be at a stationary point. Hence $\mathbf{d} = \mathbf{0}$.]

These conditions give rise to the method of Lagrange multipliers for optimization problems with equality constraints. The method requires finding $\mathbf{x}*$ and $\boldsymbol{\lambda}^*$ which solve the equations

$$\begin{aligned} \nabla f &= \sum_i \mathbf{a}_i(\mathbf{x})\lambda_i && (50) \\ c_i(\mathbf{x}) &= 0 \qquad \text{for } i = 1, \dots r && (51) \end{aligned}$$

There are $n + r$ equations and $n + r$ unknowns, so the system is well-determined. However, the system is nonlinear (in $\mathbf{x}$) in general, and so may not be esay to solve. We can restate these conditions by introducing the *Lagrangian function* $\mathcal{L} : \mathbb{R}^{n+r} \to \mathbb{R}$,

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \sum_i \lambda_i c_i(\mathbf{x}). \tag{52}$$

The partial derivatives of $\mathcal{L}$ with respect to $\mathbf{x}$ and $\boldsymbol{\lambda}$ reproduce equations 50 and 51. Hence a necessary condition for a local minimizer is that $\mathbf{x}^*$, $\boldsymbol{\lambda}^*$ is a stationary point of the Lagrangian function. Note that this stationary point is not a minimum but a saddle point, as $\mathcal{L}$ depends linearly on $\boldsymbol{\lambda}$.
We have given first-order necessary and sufficient conditions for a local optimum. To show that this optimum is a local minimum, we would need to consider second-order conditions, analogous to the positive definiteness of the Hessian in the unconstrained case; this can be done, but will not be considered in detail here.

Example: Consider the minimization of $f(\mathbf{x}) = x_1^2 + x_2^2$ subject to the constraint $c(\mathbf{x}) = x_2 - x_1 + 3 = 0$.

▶ **Exercise 16** *(5 marks) Find the minimum of the function*

$$Q = \frac{1}{2}(x_1^2 + x_2^2 + x_3^2)$$

subject to the constraints

$$x_1 - x_2 = 1 \qquad x_2 - x_3 = 2.$$

Solve this constrained optimization problem
(a) using Lagrange multipliers
(b) by direct elimination.

▶ **Exercise 17** *(5 marks) Let A be a symmetric n by n matrix and for $\mathbf{x} \in \mathbb{R}^n$, consider minimizing $\mathbf{x}^T A\mathbf{x}$ subject to the contraint $\mathbf{x}^T\mathbf{x} = 1$. Show that the solution $\mathbf{x}^*$ of this constrained optimization problem is an eigenvector of A.*

▶ **Exercise 18** *(20 marks) Consider the problem of fitting a straight line through a set of datapoints $\mathbf{x}^i, i = 1, \dots, m$.*
Assuming that the data has zero mean, $\sum_{i=1}^m \mathbf{x}^i = \mathbf{0}$, a reasonable criterion for finding a suitable direction $\mathbf{e}$ is to find $\mathbf{e}$ that maximises

$$\frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}^i \cdot \mathbf{e})^2$$

subject to $\mathbf{e} \cdot \mathbf{e} = 1$.
Find a suitable matrix A such that the above can be written as:
Find $\mathbf{e}$ that maximises

$$\mathbf{e}^T A\mathbf{e}$$

subject to $\mathbf{e} \cdot \mathbf{e} = 1$.
(b) Form a suitable Lagrangian function $\mathcal{L}(\mathbf{e}, \lambda)$ for the above optimisation problem.
Show that a solution $\mathbf{e}^*$ to this optimisation problem is an eigenvector of A. How many startionary points will there be for an $n \times n$ matrix A of full rank?
(c) Given $A = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$, find a vector $\mathbf{e}$ that maximises

$$\mathbf{e}^T A\mathbf{e}$$

subject to $\mathbf{e}^T\mathbf{e} = 1$. Hint: Be careful to ensure that your solution $\mathbf{e}^*$ maximises $\mathbf{e}^T A\mathbf{e}$ subject to the constraint.

7.3 Computational approaches to constrained optimization

There are a wide variety of computational approaches to the non-linear constrained optimization problem, going under names such as penalty and barrier function methods, the reduced-gradient method, the augmented Lagrangian method and the projected Lagrangian method. There are also many possible choices depending on whether the constraints are equality or inequality constraints, and whether they are linear or non-linear. We will consider the penalty and barrier function methods, which are not as effective as more sophisticated techniques, but offer a good place to start and are relatively straightforward to implement.
Suppose that we are minimizing a function $f(\mathbf{x})$ subject to constraints $c_i(\mathbf{x}) = 0$ for $i = 1, \ldots, r$. We can construct a quadratic penalty function

$$P_Q(\mathbf{x}, \rho) = f(\mathbf{x}) + \frac{\rho}{2}\mathbf{c}(\mathbf{x})^T\mathbf{c}(\mathbf{x}), \tag{53}$$

where $\mathbf{c}(\mathbf{x})$ is the vector of constraints $(c_1(\mathbf{x}), c_2(\mathbf{x}), \ldots, c_r(\mathbf{x}))^T$ and ρ is the *penalty parameter.*
Note that the penalty term is continuously differentiable. Under mild technical conditions, if $\mathbf{x}^*(\rho)$ denotes an unconstrained minimum of P_Q then it can be shown that

$$\lim_{\rho \to \infty} \mathbf{x}^*(\rho) = \mathbf{x}^*$$

where $\mathbf{x}^*$ is the constrained minimum of f. However, we cannot just choose ρ to be large initially. If the number of constraints is less than n, then the condition number of the Hessian at $\mathbf{x}^*(\rho)$ becomes larger with increasing ρ. Instead, it is more effective in practice to carry out several unconstrained minimizations for gradually increasing ρ, with each one starting at the point where the previous one finished.
The effect of the penalty term is to create a local minimum near to $\mathbf{x}^*$ for sufficiently large ρ. Simple examples show that the penalty function may be unbounded below for any value of ρ. An unconstrained minimization algorithm may move out side the feasible region. Thus it may be necessary to modify the algorithm to detect and, if possible, recover from unboundedness. In addition, the starting point for search can be very important.