1 Minima of a multidimensional function

Assume we have a quadratic form in a multidimensional space and we are interested in searching for the minimum. The quadratic form can be written as

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

The corresponding gradient is

$$\nabla f = \mathbf{A} \cdot \mathbf{x} - b$$

2 Steepest descent

The steepest descent algorithm is based on the idea of progressive minimisation along the direction of maximum slope (e.g. the gradient).

The algorithm is thus the following

- Calculate the force (the gradient) at $\mathbf{x}_0$
- Move along the direction of the force ($\mathbf{x}(\lambda) = \mathbf{x}_0 + \lambda\mathbf{F}$) until a minimum is found (at λ_{min})
- update $\mathbf{x}_0 = \mathbf{x}_0 + \lambda_{min}\mathbf{F}$
- iterate the process

Note that at each iteration, the gradient is perpendicular to the previous one, since by definition the starting point is a minimum for the direction of the previous gradient.

3 Conjugate Gradient

For quadratic forms with very elongated shapes the steepest-descent method is not very efficient, since it can continue exploring the same directions over and over. A more effective way is provided by the so-called conjugate gradient, which is based on the idea that ALL new search directions should be orthogonal to the previously explored ones.

The method is based on the following mathematical properties. Let's define two arbitrary initial vectors

$$\mathbf{g}_0 \qquad \mathbf{h}_0 = \mathbf{g}_0$$

and let's build the sequence of vectors

$$\mathbf{g}_{i+1} = \mathbf{g}_i - \lambda_i \mathbf{A} \cdot \mathbf{h}_i \qquad \mathbf{h}_{i+1} = \mathbf{g}_{i+1} - \gamma_i \mathbf{h}_i$$

where

$$\lambda_i \equiv \frac{\mathbf{g}_i \cdot \mathbf{g}_i}{\mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i} \qquad \gamma_i \equiv -\frac{\mathbf{g}_{i+1} \cdot \mathbf{A} \cdot \mathbf{h}_i}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i}$$

With this choice

$$\mathbf{g}_i \cdot \mathbf{g}_{i+1} = 0 \qquad \mathbf{h}_{i+1} \cdot \mathbf{A} \cdot \mathbf{h}_i = 0$$

Indeed,

$$\mathbf{g}_i \cdot \mathbf{g}_{i+1} = \mathbf{g}_i \cdot \mathbf{g}_i - \lambda_i \mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i$$

and

$$\mathbf{h}_{i+1} \cdot \mathbf{A} \cdot \mathbf{h}_i = \mathbf{g}_{i+1} \cdot \mathbf{A} \cdot \mathbf{h}_i + \gamma_i \mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i$$

Both quantities vanish when the definitions of λ_i and γ_i are substituted in the expressions.

It is possible to show (see the book Numerical Recipes) that the sequence of generated vectors satisfy (for $j \neq i$)

$$\mathbf{g}_i \cdot \mathbf{g}_j = 0 \qquad \mathbf{h}_j \cdot \mathbf{A} \cdot \mathbf{h}_i = 0 \tag{1}$$

The $\mathbf{g}_i$ are thus all orthogonal.

Using these definitions, we can prove two additional relations. Since $\mathbf{h}_i = \mathbf{g}_i + \gamma_{i-1}\mathbf{h}_{i-1}$ (from the definition)

$$\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i = (\mathbf{g}_i + \gamma_{i-1}\mathbf{h}_{i-1}) \cdot \mathbf{A} \cdot \mathbf{h}_i = \mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i + \gamma_{i-1}\mathbf{h}_{i-1} \cdot \mathbf{A} \cdot \mathbf{h}_i$$

and the second contribution vanishes (due to Eq. 1) . Hence $\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i = \mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i$ (this does not means that $\mathbf{g}_i = \mathbf{h}_i$, but only that the scalar product is the same !)

The second additional relation is

$$\mathbf{h}_i \cdot \mathbf{g}_i = (\mathbf{g}_i + \gamma_{i-1}\mathbf{h}_{i-1}) \cdot \mathbf{g}_i = \mathbf{g}_i \cdot \mathbf{g}_i + \gamma_{i-1}\mathbf{h}_{i-1} \cdot \mathbf{g}_i$$

The last term can be written (iterating the definition of $\mathbf{h_i}$)

$$\mathbf{h}_{i-1} \cdot \mathbf{g}_i = (\mathbf{g}_{i-1} + \gamma_{i-2}\mathbf{h}_{i-2}) \cdot \mathbf{g}_i = \gamma_{i-2}\mathbf{h}_{i-2} \cdot \mathbf{g}_i$$

where we have used the condition of orthogonality of the past $\mathbf{g_i}$. Iterating over and over this process by keeping substituting for h_{i-k} one ends with

$$\mathbf{h}_{i-1} \cdot \mathbf{g}_i = \prod_{k=0}^{i-2} \gamma_k \mathbf{h}_0 \cdot \mathbf{g}_i$$

and since $\mathbf{h}_0 = \mathbf{g}_0$ and $\mathbf{g}_0$ is perpendicular to $\mathbf{g}_i$ the term vanishes. Hence we have demonstrated that

$$\mathbf{h}_i \cdot \mathbf{g}_i = \mathbf{g}_i \cdot \mathbf{g}_i$$

Let's now go back to our minimisation problem. Let's define $\mathbf{g}_i$ as the gradient of the function we want to minimiza in a generic point $\mathbf{P_i}$ e.g.

$$\mathbf{g}_i = -\nabla f(\mathbf{P_i}) = -\mathbf{A} \cdot \mathbf{P_i} + b$$

and let's define $\mathbf{g}_{i+1}$ as the gradient evaluated at the minimum along the (for the time being arbitrary) direction $\mathbf{h}_i$. Then

$$\mathbf{g}_{i+1} = -\nabla f(\mathbf{P_i} + \lambda_{min}\mathbf{h}_i) = -\mathbf{A}\cdot(\mathbf{P_i} + \lambda_{min}\mathbf{h}_i) + b = \mathbf{g}_i - \lambda_{min}\mathbf{A}\cdot\mathbf{h}_i$$

In $\mathbf{P_i} + \lambda_{min}\mathbf{h}_i$ the gradient is perpendicular to the direction of the previous "search" direction and so

$$\mathbf{h}_i \cdot \mathbf{g}_{i+1} = 0$$

and hence

$$\mathbf{h}_i \cdot \mathbf{g}_i - \lambda_{min}\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i = 0$$

from which

$$\lambda_{min} = \frac{\mathbf{h}_i \cdot \mathbf{g}_i}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i}$$

But, according to what we have demonstrated before

$$\lambda_{min} = \frac{\mathbf{h}_i \cdot \mathbf{g}_i}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i} = \frac{\mathbf{g}_i \cdot \mathbf{g}_i}{\mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i} = \lambda_i$$

and hence λ_{min} coincides with the definition of λ_i and thus it is possible to build the sequence of vectors previously discussed simply identifying $\mathbf{g}_i$ with the gradient of the function we want to minimise.

This identity provides a method for generating the directions to progressively search for the local minimum, avoiding resampling the previously explored directions. What have we gained by doing all this? We have been able to define the vectors $\mathbf{g_i}$ without knowing the Hessian matrix $\mathbf{A}$, avoiding even the storage necessary to store such a matrix. A sequence of directions $\mathbf{h_i}$ is constructed, using only line minimizations, evaluations of the gradient vector, and an auxiliary vector to store the latest in the sequence of $\mathbf{g_i}$'s.

We also note that

$$\gamma_i \equiv -\frac{\mathbf{g}_{i+1} \cdot \mathbf{A} \cdot \mathbf{h}_i}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i}$$

making use of the relation $\mathbf{g}_{i+1} = \mathbf{g}_i - \lambda_i \mathbf{A} \cdot \mathbf{h}_i$ can be written as

$$\gamma_i = -\frac{\mathbf{g}_{i+1} \cdot (\mathbf{g}_i - \mathbf{g}_{i+1})}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i} \frac{1}{\lambda_i} = -\frac{\mathbf{g}_{i+1} \cdot (\mathbf{g}_i - \mathbf{g}_{i+1})}{\mathbf{h}_i \cdot \mathbf{A} \cdot \mathbf{h}_i} \frac{\mathbf{g}_i \cdot \mathbf{A} \cdot \mathbf{h}_i}{\mathbf{g}_i \cdot \mathbf{g}_i} = \frac{\mathbf{g}_{i+1} \cdot (\mathbf{g}_{i+1} - \mathbf{g}_i)}{\mathbf{g}_i \cdot \mathbf{g}_i}$$

and since $\mathbf{g}_i$ is orthogonal to $\mathbf{g}_{i+1}$ and equivalent alternative way for writing γ_i is

$$\gamma_i = \frac{\mathbf{g}_{i+1} \cdot \mathbf{g}_{i+1}}{\mathbf{g}_i \cdot \mathbf{g}_i}$$

In summary, evaluating the gradient one find $\mathbf{g}_{i+1}$. From this one find γ_i and from γ_i one finds $\mathbf{h}_{i+1}$ which is the new search direction.

Finally, we comment on the fact that since $\mathbf{g}_i \cdot \mathbf{g}_{i+1} = 0$, then an equivalent expression for γ_i is provided by

$$\gamma_i = \frac{(\mathbf{g}_{i+1} - \mathbf{g}_i) \cdot \mathbf{g}_{i+1}}{\mathbf{g}_i \cdot \mathbf{g}_i}$$

The two expressions for γ_i correspond to the Polak and Ribiere and to the Fletcher-Reeves version of the conjugate gradient algorithm. These two expressions are equal for exact quadratic forms, since $\mathbf{g}_i \cdot \mathbf{g}_{i+1} = 0$. In the real world, however, the function to be minimised is not exactly a quadratic form. Arriving at the supposed minimum of the quadratic form, one may still need to proceed for another set of iterations. There is some evidence that the Polak-Ribiere formula accomplishes the transition to further iterations more gracefully.

3.1 How do we really do it...

Implementation of the minimisation procedure requires a subroutine to find the minimum along one direction. The standard algorithm is explained in the figure. It requires as input the coordinates of three points bracketing the minimum.

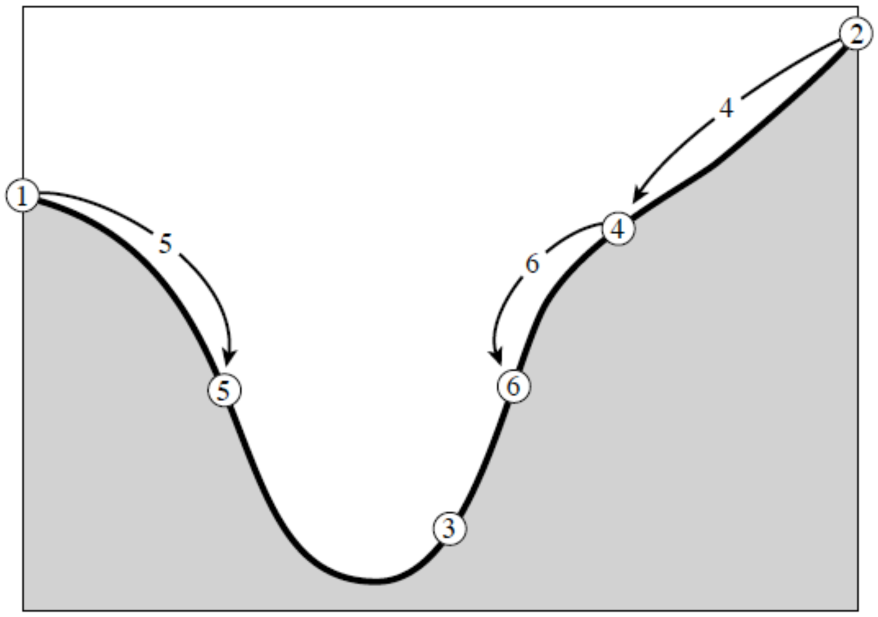

Figure 10.1.1. Successive bracketing of a minimum. The minimum is originally bracketed by points 1,3,2. The function is evaluated at 4, which replaces 2; then at 5, which replaces 1; then at 6, which replaces 4. The rule at each stage is to keep a center point that is lower than the two outside points. After the steps shown, the minimum is bracketed by points 5,3,6.

or the inverse parabolic interpolation

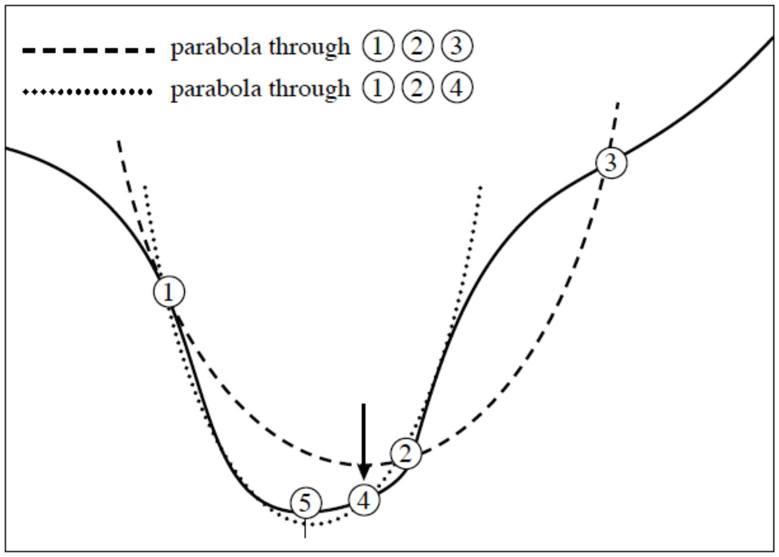

Figure 10.2.1. Convergence to a minimum by inverse parabolic interpolation. A parabola (dashed line) is drawn through the three original points 1,2,3 on the given function (solid line). The function is evaluated at the parabola's minimum, 4, which replaces point 3. A new parabola (dotted line) is drawn through points 1,4,2. The minimum of this parabola is at 5, which is close to the minimum of the function.

From the starting point $\mathbf{x_0}$, moving along the force directions one first search for these three points bracketing the minimum and then call the subroutine that find the minimum within a certain tolerance.