1. Basic knowledge beforehand
2. Gradient (of a scalar over a vector)

Note: is the direction that **increases** *f* the fastest!

1. Hessian matrix

Note: we can usually express a quadratic function in the term of:

In the above expression, *A* (a N by N symmetric matrix) would be the hessian matrix (that also explains why quadratic programming is faster than fmincon: no need to calculate Hessian again!).

Note: If *f* is a vector instead of scalar, the Hessian matrix becomes a 3D matrix.

1. Jacobian matrix

A Jacobian matrix is the gradient of a vector (M by 1) over another vector (N by 1).

In fact the gradient in (1) is just a special case for Jacobian.

For following discussions we define the optimization target as .

1. Gradient descent

Gradient descent (steepest descent) method tries to move the variable vector in the **opposite** direction of the Jacobian matrix, such that the value of *f* keeps decreasing in each iteration:

Here *α­k* is also called “learning rate”. Some algorithms will adjust this rate to make the process fast in the beginning and avoid wiggling in the end.

1. Newton’s method
2. Levenberg-Marquadt
3. Conjugate gradients