# 5 Quasi-Newton Methods and BFGS algorithm

We solved our optimization problem first using gradient descent and then using Newton method. We highlighted the major differences between these 2 models. Summarizing, the Newton method:

* Does not need a learning rate parameter
* Converges much faster than gradient descent
* Is sensitive to initial conditions, more than gradient descent, especially if our objective function is non-convex.
* Is very computationally expensive, with a computational time of O(, due to the fact that we have to compute the Hessian and its inverse

We can obtain a method which is a sort of hybrid between gradient descent and Newton’s method, where we can have faster convergence than gradient descent, but lower operational cost per iteration than Newton’s method. This class of optimization methods is called *quasi-Newton methods.*Recall that in Newton’s method, we have to make the following update at each iteration:

where is a positive definite Hessian. If instead of the Hessian we use an approximation , we can have a much faster algorithm comparing to Newton’s method. This happens because the positive definite matrix B is updated iteration to iteration using information computed from previous steps, so we compute fewer new quantities at each iteration.  
  
A common feature to all the quasi-Newton methods is that the Hessian approximation B must satisfy the *quasi-Newton condition* (or *secant equation*):

which is obtained from the first order Taylor expansion of about . This condition essentially states that the product of the approximation of the inverse Hessian () and the change in the variable space () should approximate the difference in gradients (). In other words, it expresses a relationship between changes in the variable space and changes in the gradient.   
  
We can understand quite easily this condition in one dimension case, where we replace the second derivative with its finite difference approximation, but things could complicate when dealing with n-dimensional secant condition.

The problem now is that our equation only has n components, while B is in general a symmetric n × n matrix with n(n+1)/2 components. We are dealing with an undetermined system (while in the one-dimension case we had a square system).  
  
The quasi-Newton method solves this problem by imposing further constraints on B to solve for it. These additional constraints depend on the specific quasi-Newton method used. In this case, we focus on the BFGS method, which is one of the most popular quasi-Newton methods.

## BFGS optimization

The name of the method comes from the names of its creators: Broyden, Fletcher, Goldfarb, and Shanno, who each came up with the algorithm independently in 1970.  
  
To determine a scheme for B in n>1 dimensions, we will need additional constraints. Two of them are positive-definiteness and symmetry of B(these properties should be valid in each update).   
  
A third property we want is for to be sufficiently close to at each update . We make use of the matrix norm to characterize this property. But we recall from the Newton method that we need the Hessian’s inverse (and not the Hessian itself), so we can compute it directly:

,

where and   
  
Instead of requiring the full Hessian matrix at the point to be computed as ,,,, the approximate Hessian at stage *k* is updated by the addition of two symmetric rank-one matrices:

In order to maintain the symmetry and positive definiteness of , the update form can be chosen as:

We impose the secant condition and we choose and . We then have:

Substituting and into the previous equation we have the BFGS update:

# Bibliography

Nocedal, J., Wright, S., Numerical Optimization (Chapter 6). Springer, 2nd edition, 2006

Goldfarb, D.F., “A family of variable-metric methods derived by variational means”, Math. Comp. 24, 23-26 (1970)

Shanno D., “Conditioning of quasi-Newton methods for function minimization”, Math. Comp. 24, 23-26 (1970)