BFGS Algorithm

Objective: Minimize a scalar function $f : \mathbb{R}^n \to \mathbb{R}$.

Given: A continuously differentiable function f and its gradient ∇f.

Initialization:

1. Choose an initial guess x_0 for the minimum.
2. Set B_0 as an approximation to the inverse Hessian (often the identity matrix I is used).
3. Set $k = 0$.

Iterative Process:

1. **Compute the Search Direction**:
   Compute the search direction p_k by solving the linear system:
   $$B_k s_k = -\nabla f(x_k)$$
2. **Line Search**:
   Perform a line search to find the step size α_k that satisfies the Wolfe conditions:
   - Armijo rule: $f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^T p_k$, where $0 < c_1 < 1$ is a constant (typically $c_1 = 10^{-4}$).
   - Curvature condition: $\nabla f(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f(x_k)^T p_k$, where $c_1 < c_2 < 1$ (typically $c_2 = 0.9$).
3. **Update the Current Point**:
   Set:
   $$x_{k+1} = x_k + \alpha_k p_k$$
4. **Compute the Difference**:
   Calculate the differences in gradients and the step made:
   $$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$$
   $$s_k = x_{k+1} - x_k$$
5. **Update the Approximate Inverse Hessian**:
   Using the BFGS formula:
   $$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}$$
6. **Convergence Test**:
   If $\nabla f(x_{k+1})$ is sufficiently small or some other convergence criteria are met, then stop. Otherwise, increment k by 1 and go back to step 1.

Remarks:

- BFGS belongs to a category of optimization algorithms called quasi-Newton methods. They derive their name because they build up an approximation to the Hessian (or its inverse) rather than compute it directly.
- The BFGS update formula ensures that B_{k+1} remains positive definite, provided B_k is positive definite and the Wolfe conditions are satisfied.
- BFGS typically converges faster than the steepest descent method but may be slower than Newton's method (which requires the Hessian). However, BFGS doesn't require the explicit computation of the Hessian, making it more practical for functions where the Hessian is difficult or expensive to compute.