Optimization Methods

Introduction

Optimization is a central problem in many fields. It involves minimizing (or maximizing) an objective function over some domain. Here, we look at a selection of optimization methods that span both gradient-based and derivative-free approaches.

An overview of algorithms

All optimization algorithms have a common approach: start with an initial guess x_0 and generate a sequence of iterates $\{x_k\}$ that converge to the optimal solution x^*.

Line Search Methods

Here, we search along a direction p_k for a new iterate $(x_k + 1)$ that has a smaller function value than $f(x_k)$, i.e., find the length α to be walked from x_k in the direction p_k. The optimization procedure considered is:

$$\{\alpha^*, p_k^*\} = \arg\min_{\alpha, p_k} f(x_k + \alpha p_k),$$

leading to the update:

$$x_{k+1} = x_k + \alpha^* p_k^*.$$

Depending on the choice of the update direction p_k. These are further classified as follows

1. **Steepest descent method** – Here p_k is chosen to be the direction in which f decreases the most, i.e., $p_k = -\nabla f_k$.
2. **Newton method** – Here curvature information is used to infer the direction. In particular, if the Hessian, $\nabla^2 f_k$, is positive definite, then

$$p_k = -(\nabla^2 f_k)^{-1} \nabla f_k.$$

3. **Quasi-Newton methods** — These use *approximations*, B_k, of the Hessian that are cheaper to compute and are typically faster than steepest descent methods. Moreover, they allow incremental computations between iterations, i.e., B_{k+1} is computed from B_k.

1. Gradient Descent or Steepest Descent method

Update Rule

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

Proof

Consider a descent direction p_k and a small step length ϵ. Using Taylor's theorem:

$$f(x_k + \epsilon p_k) = f(x_k) + \epsilon p_k^\top \nabla f_k + \mathcal{O}(\epsilon^2)$$

If ϵ is small, the quadratic term becomes negligible and the term

$$p_k^\top \nabla f_k$$

can be considered as an inner product. Let θ_k be the angle between p_k and ∇f_k. Then:

$$p_k^\top \nabla f_k = \|p_k\| \|\nabla f_k\| \cos\theta_k$$

The maximum decrease in f occurs when $\theta_k = \pi$, i.e., when $p_k \propto -\nabla f_k$. This gives the steepest descent direction.

Estimating the Learning Rate α

Backtracking Line Search:
A very popular line search strategy involves working with the sufficient decrease condition and using an intelligent method of backtracking to choose an appropriate step length. The idea is to start with an initial value of α (e.g., $\alpha^{(0)} = 1$) and use the condition:

$$f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f_k^T p_k,$$

where $0 < c < 1$ is a small constant (e.g., $c = 10^{-4}$).
If this $\alpha^{(0)}$ is not suitable, we simply scale it by a factor $0 < \rho < 1$ until the condition is satisfied, i.e.,

$$\alpha^{(i+1)} = \rho \alpha^{(i)}.$$

When It Works Best

Best for smooth, convex problems, however this method is sensitive to learning rate α. For a very large value of α the iteration step overshoots and the function value diverges instead of converging. In case of a very small value of α the number of steps required for convergence will be very large. Therefore we must be very careful while choosing the value of α.

2. Newton's Method

Update Rule

$$x_{k+1} = x_k - H^{-1}(x_k)\nabla f(x_k)$$

where $H = \nabla^2 f(x_k)$

Proof

Newton's method uses second-order curvature information to accelerate convergence near a local minimum. Consider a second-order Taylor series approximation to $f(x_k + p)$:

$$f(x_k + p) \approx f_k + p^T \nabla f_k + \frac{1}{2} p^T \nabla^2 f_k p$$

We are currently at x_k and want to go to the next step $x_k + p$. We enforce a condition that this new step should be the minimizer of f. If this is a stationary point, then:

$$\nabla_p f(x_k + p) = 0$$

Using the second-order model and differentiating with respect to p, we get:

$$0 = \nabla f_k + \nabla^2 f_k\, p$$

Rearranging gives:

$$p = -(\nabla^2 f_k)^{-1} \nabla f_k$$

If we are minimising a quadratic function, then in one step we can reach the minimum.

When It Works Best

Best for Quadratic functions,with strong convex profile with well defined, invertible and positive definite Hessians. However in higher dimensions, hessian calculation becomes expensive $O(n^2)$, and matrix inversion which is $O(n^3)$ this is not the best-suited method.

3. BFGS (Quasi-Newton)

This method has the same update rule as Newtons but here instead of calculating the inverse of hessian we try to estimate it from the previous step.

0.0.1 Approximate Hessian Update

$$H_{k+1} = \left(I - \rho_k s_k y_k^\top\right) H_k \left(I - \rho_k y_k s_k^\top\right) + \rho_k s_k s_k^\top$$

0.0.2 Advantages

Does not require explicit Hessian; suitable for high-dimensional problems.

4. Nelder-Mead Simplex Method

Overview

Nelder-Mead is a derivative-free method that uses a simplex (a geometric figure with $n + 1$ vertices when dealing with n dimensional problem) and modifies it through a set of operations depending on the function value at the vertices of the simplex. These operations are named as:

- Reflection- Reflect the worst point across the centroid of the remaining points.
- Expansion- If the point achieved by reflection is fairly good, try going even further in that particular direction.
- Contraction- If the point achieved by reflection is not better than the worst point, move the worst point to 1/4th or 3/4th of the distance between the refelcted and original point and then select the best performing point.
- Shrink - when everything fails reduce the simplex size towards the best point.

Repeat this set of operations till a maximum number of iterations have been reached or till a tolerance level is breached.

When It Works Best

Effective for noisy, non-smooth objective functions. But often gets stuck in local minimas and saddle points.