Smoothed Analysis: An Overview

1 Smoothed Analysis

Smoothed analysis is a framework introduced by Spielman and Teng to understand the behavior of algorithms in practice. It provides a more refined analysis than the traditional worst-case and average-case analyses by considering the performance of algorithms under slight random perturbations of the input data.

1.1 Basic Ideas

In the worst-case analysis, the performance of an algorithm is evaluated based on the most difficult inputs, which can sometimes lead to overly pessimistic assessments. On the other hand, average-case analysis assumes a probability distribution over all possible inputs, which might not always reflect real-world scenarios.

Smoothed analysis bridges the gap between these two extremes. It considers the performance of an algorithm on a given input, but also takes into account small random perturbations of this input. This approach helps in understanding why certain algorithms, like the simplex method for linear programming, perform well in practice despite having poor worst-case time complexities.

1.2 Mathematical Description

Consider an algorithm A and an input x. In smoothed analysis, we perturb x slightly to obtain a new input x'. The performance of A is then evaluated on x'.

Mathematically, let Φ be a performance measure (e.g., running time), and let x be an input. We add a small random noise ξ to x to get a perturbed input $x' = x + \xi$. The smoothed performance $\Phi_{\text{smoothed}}(x)$ is defined as the expected value of $\Phi(x')$ over the random noise ξ:

$$\Phi_{\text{smoothed}}(x) = \mathbb{E}_{\xi}[\Phi(x + \xi)].$$

1.3 Application to the Simplex Algorithm

Smoothed analysis has been particularly insightful in explaining the practical efficiency of the simplex algorithm for linear programming. Despite its exponential worst-case running time, the simplex algorithm is efficient in practice.

1.3.1 Condition Numbers and Geometry

The performance of the simplex algorithm is closely related to the geometry of the input linear program. Condition numbers of the constraint matrices provide insights into the "flatness" of the corners of the feasible polytope. Smoothed analysis helps in understanding the algorithm's behavior under perturbations of the input data, leading to variations in the condition numbers and the geometry of the polytope.

1.3.2 Shadow Vertex Method

The shadow vertex method is a technique that extends the simplicity of the two-dimensional simplex method to higher dimensions. It considers the "shadow" of the feasible polytope, projected onto a plane defined by two objective functions. The method involves walking along the vertices of this shadow polygon, akin to the two-dimensional simplex method, to find the optimal solution.

2 Kaczmarz Algorithm

2.1 Basic Ideas

The Kaczmarz algorithm is an iterative method used for solving systems of linear equations. It is particularly effective for large, sparse systems. The algorithm converges to the solution by successively projecting a guess onto the hyperplanes defined by each equation in the system.

2.2 Mathematical Description

Consider a system of linear equations represented as $Ax = b$, where

- A is a matrix with m rows and n columns,
- x is the unknown vector of length n,
- b is the known vector of length m.

The Kaczmarz algorithm updates the solution estimate $x^{(k)}$ at each iteration k using the formula:

$$x^{(k+1)} = x^{(k)} + \frac{b_i - \langle a_i, x^{(k)} \rangle}{\|a_i\|^2} a_i$$

where

- a_i is the i-th row of A,
- b_i is the i-th element of b,
- $\langle a_i, x^{(k)} \rangle$ is the dot product of a_i and $x^{(k)}$.

2.3 Example

Consider the system of equations

$$\begin{aligned} 2x + 3y &= 5, \\ 4x - 5y &= 6. \end{aligned}$$

We initialize our solution estimate as $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. The Kaczmarz algorithm will then iteratively update the estimates of x and y by projecting the current estimates onto the lines defined by each equation until convergence.

3 CG Algorithm

The Conjugate Gradient (CG) algorithm is an iterative method for solving systems of linear equations, particularly those that are symmetric and positive definite. It is known for its efficiency and accuracy, making it a popular choice in various applications, including numerical simulations and optimization problems.

3.1 Basic Ideas

The CG algorithm aims to solve a system of linear equations of the form $Ax = b$, where:

- A is a symmetric, positive definite matrix.
- x is the vector of unknowns.
- b is the known vector.

The algorithm iteratively refines an initial guess of the solution, constructing a sequence of vectors that converge to the exact solution.

3.2 Mathematical Description

The steps of the CG algorithm are as follows:

1. Compute the residual $r_0 = b - Ax_0$.
2. Set the initial search direction $p_0 = r_0$.
3. For $k = 0, 1, 2, \ldots$ until convergence:

(a) Compute the step size $\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$.

(b) Update the solution $x_{k+1} = x_k + \alpha_k p_k$.

(c) Update the residual $r_{k+1} = r_k - \alpha_k A p_k$.

(d) If r_{k+1} is sufficiently small, stop.

(e) Compute the coefficient $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$.

(f) Update the search direction $p_{k+1} = r_{k+1} + \beta_k p_k$.

4 GMRES Algorithm

The Generalized Minimal Residual (GMRES) algorithm is an iterative method for solving non-symmetric linear systems of equations. It is particularly useful for large and sparse systems and does not require the matrix to be symmetric or positive definite.

4.1 Basic Ideas

GMRES solves a system of linear equations of the form $Ax = b$, where:

- A is a square matrix.
- x is the vector of unknowns.
- b is the known vector.

The algorithm minimizes the residual norm over a Krylov subspace generated by the matrix and the initial residual.

4.2 Mathematical Description

The GMRES method involves the following steps:

1. Compute the initial residual $r_0 = b - Ax_0$ and normalize it to get the first basis vector of the Krylov subspace.
2. For each iteration, apply Arnoldi's method to construct an orthogonal basis of the Krylov subspace.
3. Solve the least squares problem to minimize the residual norm in the Krylov subspace.
4. Update the approximation of the solution.