Optimization Algorithms Overview

Method	Details
Simplex Method	**Description**: Solves LPs by moving along the edges of the feasible region to find the optimal vertex. **Convergence**: Exponential in the worst case but generally polynomial time for practical problems. **When to Use**: Suitable for small to medium-sized LPs, especially when a vertex solution is desired.
Gradient Descent (GD)	**Description**: Iteratively updates variables in the direction of the negative gradient. **Convergence Rates**: Convex and L-smooth: $O(1/k)$. μ-strongly convex and L-smooth: $O((1-\mu/L)^k)$. γ-PL condition and L-smooth: $O((1-\gamma/L)^k)$. **When to Use**: For smooth and (strongly) convex problems; careful step size selection is crucial.
Newton's Method	**Description**: Utilizes second-order derivative (Hessian) to find stationary points. **Convergence**: μ-strongly convex, L-smooth, Lipschitz continuous Hessian: Globally linear with fast local quadratic convergence, once it is close enough to optimum. **When to Use**: For smooth problems requiring high precision and solving linear systems with the Hessian isn't too expensive.
BFGS	**Description**: A quasi-Newton method approximating the Hessian matrix iteratively using the secant condition. **Convergence**: μ-strongly convex, L-smooth, Lipschitz continuous Hessian: Globally linear. Exhibits fast superlinear convergence once it is close enough to optimum. **When to Use**: When exact Hessians in Newton's method are too costly to compute or invert, but faster convergence than GD is desired.
L-BFGS	**Description**: Memory-efficient version of BFGS, storing only a limited amount of information to approximate the Hessian. **Convergence**: Linear convergence, performs better than (A)GD in practice but no supporting theory. **When to Use**: Ill-conditioned large-scale optimization problems with memory constraints.
Interior-Point Methods (IPM)	**Description**: Solves constrained optimization problems using barrier functions. **Convergence**: Polynomial time for LPs, QPs, and conic optimization problems. **When to Use**: For small to medium-scale problems with complex constraints. When a reliable, high-accuracy solution is required.
Accelerated Gradient Descent (Nesterov's)	**Description**: Enhances basic GD by incorporating a momentum term. **Convergence Rates**: Convex and L-smooth: Optimal $O(1/k^2)$. μ-strongly convex and L-smooth: $O((1-\sqrt{\mu/L})^k)$. **When to Use**: For smooth, convex problems needing faster convergence than standard GD and provable rates.
Stochastic Gradient Descent (SGD)	**Description**: Computes the gradient based on a subset of data at each iteration. **Convergence Rates**: Convex and Lipschitz continuous: $O(1/\sqrt{k})$ with decaying step size $\eta_k = O(1/\sqrt{k})$. μ-strongly convex, L-smooth: 1. linear to a ball centered at the optimum of radius $O(\epsilon)$, when using appropriate *fixed* step size equal to $O(\epsilon)$. 2. $O(1/k)$ with decaying step size $\eta_k = O(1/k)$. **When to Use**: Large-scale or online learning problems where full gradient computation is intractable.
Stochastic Variance Reduced Gradient (SVRG)	**Description**: Mitigates gradient variance in SGD by periodically computing a full gradient. **Convergence Rates**: L-smooth and convex: $O(1/k)$. μ-strongly convex and L-smooth: Linear, $O((1-\mu/L)^k)$. **When to Use**: Large-scale learning with variance reduction needs, feasible when full gradient computation intermittently is affordable.
Projected Gradient Descent	**Description**: Extends GD by projecting onto the feasible set after each iteration. **Convergence Rates**: Convex and L-smooth: $O(1/k)$. μ-strongly convex and L-smooth: Linear $O((1-\mu/L)^k)$. **When to Use**: Constrained optimization within a convex feasible set with an easy-to-compute projection operator.