UNIVERSITY OF MARYLAND

MSML610: Advanced Machine Learning

Numerical Optimization

Instructor: Dr. GP Saggese - gsaggese@umd.edu

References:

Optimization / numerical methods

- **Optimization / numerical methods**

Unconstrained Optimization

- Optimization without any restrictions on variable values
- Goal: minimize $f(x)$ where $x \in \mathbb{R}^n$
- First-order condition: $\nabla f(x) = 0$
- Second-order condition uses the Hessian $\nabla^2 f(x)$
- Common in training ML models (e.g., logistic regression)

Gradient Descent

- Iterative optimization using update: $x_{t+1} = x_t - \eta \nabla f(x_t)$
- Step size η (learning rate) controls convergence
- Simple and widely used for differentiable functions
- Converges slowly near saddle points or with bad conditioning

Stochastic Gradient Descent (SGD)

- Approximates gradient using a mini-batch or single sample
- $x_{t+1} = x_t - \eta \nabla f_i(x_t)$ for sample i
- Introduces noise, enabling escape from saddle points
- Key algorithm in training deep neural networks

Convex Optimization

- Convex function: $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$
- Global optimum is also a local optimum
- Efficient and stable algorithms exist (e.g., interior point)
- Underpins SVMs, LASSO, ridge regression

Constrained Optimization

- Optimization with equality and/or inequality constraints
- Form: minimize $f(x)$ s.t. $g_i(x) \leq 0$, $h_j(x) = 0$
- Solved using Lagrange multipliers and KKT conditions
- Relevant for resource allocation, fairness, and policy optimization

Newton's Method

- Uses second-order information: $x_{t+1} = x_t - [\nabla^2 f(x_t)]^{-1} \nabla f(x_t)$
- Quadratic convergence near optimum
- Requires computing and inverting Hessian, expensive for large problems
- Used in logistic regression and classical ML

Quasi-Newton Methods

- Approximate Hessians to reduce computational cost
- Example: BFGS, L-BFGS (limited memory version)
- Efficient for medium-scale optimization problems
- Widely used in ML libraries (e.g., `scipy.optimize`)

Line Search and Trust Region Methods

- Line search: choose step size η minimizing $f(x - \eta \nabla f(x))$
- Trust region: approximate f locally and restrict step size
- Both improve convergence of gradient-based methods
- Essential in practical solvers

Numerical Linear Algebra

- Core to solving optimization problems (e.g., solving $Ax = b$)
- Includes LU/QR decomposition, matrix inversion, eigenvalue problems
- Precision and conditioning affect stability
- Enables efficient implementation of ML algorithms

Regularization Techniques

- Modify objective to improve generalization and numerical stability
- Examples: L_2 (ridge), L_1 (lasso)
- Encourages sparsity, reduces overfitting
- Important in ill-posed or high-dimensional problems

Duality

- Every constrained problem has an associated dual problem
- Dual variables correspond to Lagrange multipliers
- Strong duality: primal $=$ dual optimum under certain conditions
- Used in SVMs, variational inference, and Lagrangian relaxation

Backtracking and Adaptive Step Sizes

- Adjust learning rate based on local curvature or function decrease
- Backtracking line search reduces step size until sufficient decrease
- Adaptive methods include AdaGrad, RMSProp, Adam

Coordinate Descent

- Optimizes one variable at a time while fixing others
- Efficient for high-dimensional sparse problems
- Common in LASSO and logistic regression with L_1 penalty

Conjugate Gradient Method

- Iterative method for large symmetric positive definite systems
- Avoids matrix inversion; uses conjugate directions
- Preferred for solving large-scale linear problems in ML

Eigenvalue and SVD Computation

- Critical in PCA, spectral methods, kernel methods
- Numerical methods include power iteration, Lanczos algorithm
- SVD: $A = U\Sigma V^T$ decomposes data into orthogonal components

Automatic Differentiation

- Programmatic computation of exact derivatives
- Used in backpropagation for deep learning
- Enables gradient-based optimization in arbitrary computational graphs

Numerical Stability and Conditioning

- Measures sensitivity of output to input perturbations
- Poor conditioning leads to inaccurate results
- Matrix condition number: $\kappa(A) = \|A\| \|A^{-1}\|$
- Influences algorithm choice and precision handling

Optimization for Non-Smooth Functions

- Non-differentiable points (e.g., ReLU, hinge loss)
- Use subgradients or proximal methods
- Important in sparse modeling, SVMs, and robust ML

Metaheuristic Algorithms

- Heuristic search methods: gradient-free and global
- Examples: genetic algorithms, simulated annealing, particle swarm
- Used in hyperparameter tuning and combinatorial optimization

Convex Relaxation and Approximation

- Replace hard non-convex problems with convex surrogates
- Example: relax integer programming to continuous space
- Often used in sparse recovery, graphical models, and ML pipelines