Introduction to Proximal Algorithms

NSI - ML Masters Seminar

Karol Chojnacki 429229

December 2024

Introduction

- **Proximal algorithms** solve nonsmooth, constrained, large-scale, or distributed optimization problems.
- Applications include:
  - First DNN Homework
  - Lasso, matrix decomposition, minimizing loss, portfolio optimization, visual and audio processing etc.

Proximal Operator: For a closed, proper, convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the proximal operator is defined as:

$$\mathrm{prox}_f(v) = \arg\min_x \left(f(x) + \frac{1}{2}\|x - v\|^2 \right).$$

Scaled Proximal Operator:

$$\mathrm{prox}_{\lambda f}(v) = \arg\min_x \left(f(x) + \frac{1}{2\lambda}\|x - v\|^2 \right),$$

where $\lambda > 0$ scales the problem.

Intuition for Proximal Operator

- Similar to how gradient descent takes steps proportional to the gradient to move towards a minimizer for smooth problems
- Proximal methods handle nonsmoothness by adding the regularization term:

$$\mathrm{prox}_f(v) = \arg\min_x f(x) + \underbrace{\frac{1}{2}\|x - v\|^2}_{\textit{regularization}}$$

- Thanks to regularization term $\mathrm{prox}_f(v)$ is a point that compromises between minimizing f and being near to v $\Longrightarrow$ $\mathrm{prox}_f(v)$ is called a proximal point of v with respect to f

Separable Sum:

$$f(x, y) = g(x) + h(y) \implies \mathrm{prox}_f(v, w) = (\mathrm{prox}_g(v), \mathrm{prox}_h(w)).$$

Affine Addition:

$$f(x) = g(x) + a^T x + b \implies \mathrm{prox}_f(v) = \mathrm{prox}_g(v - \lambda a).$$

Fixed Point Property:

$$x^\star \text{ minimizes } f \iff x^\star = \mathrm{prox}_f(x^\star).$$

Theorem:

$$v = \text{prox}_f(v) + \text{prox}_{f^\star}(v),$$

where $f^\star$ is the convex conjugate of f:

$$f^\star(y) = \sup_x \left(y^T x - f(x) \right).$$

Geometric Intuition:

- v is split into two orthogonal components:

$$v = \text{prox}_f(v) \quad (\text{aligned with } f) + \text{prox}_{f^\star}(v) \quad (\text{aligned with } f^\star).$$

- These components are complementary and balance primal and dual perspectives.

Example:

- For $f(x) = \|x\|_1$:

$$\text{prox}_f(v) = \text{soft}(v, \lambda), \quad \text{prox}_{f^\star}(v) = \text{proj}_{\|\cdot\|_\infty \le \lambda}(v).$$

Problem: Solve $\min_x f(x) + g(x)$, where:

- f is differentiable with Lipschitz gradient.
- g is closed, proper, convex.

Iteration:

$$x_{k+1} = \mathrm{prox}_{\lambda g}(x_k - \lambda \nabla f(x_k)),$$

where λ is the step size.

Convergence:

- $O(1/k)$ for fixed λ.
- Accelerated (allowing to change λ) versions achieve $O(1/k^2)$.

Approaches to Evaluate:

- Closed-form solutions:
  - For simple $f(x)$, the proximal operator has an explicit formula.
  - Example: L1 norm $f(x) = \lambda \|x\|_1$ leads to soft-thresholding.
- Iterative solvers:
  - For complex $f(x)$, use gradient-based or numerical methods.
  - Solve by the definition iteratively.
- Decomposable functions:

$$f(x) = \sum_i f_i(x_i) \implies \operatorname{prox}_f(v) = \Big(\operatorname{prox}_{f_1}(v_1), \ldots, \operatorname{prox}_{f_n}(v_n)\Big).$$

Common Functions and Proximal Operators:

- L1 Norm:

$$f(x) = \lambda \|x\|_1 \implies \text{prox}_f(v) = \underbrace{\text{sgn}(v_i)\max(|v| - \lambda, 0)}_{\text{soft}(v,\lambda)},$$

- Quadratic Function: (from DNN Homework)

$$f(x) = \frac{1}{2}\|Ax - b\|^2 \implies \text{prox}_f(v) = (A^T A + I)^{-1}(A^T b + v).$$

- Elastic Net: Combination of the two above

Common Functions and Proximal Operators:

- Indicator function of a convex set C:

$$f(x) = I_C(x) \implies \text{prox}_f(v) = \Pi_C(v),$$

  where $\Pi_C(v)$ is the projection of v on C.

- Nuclear Norm (Low-Rank Matrices):

$$f(X) = \|X\|_* \implies \text{prox}_f(V) = U\text{soft}(\Sigma, \lambda)V^T,$$

  where $U\Sigma V^T$ is the SVD of V.

Applications: Denoising, Deblurring, Super-Resolution

Flexibility:

- Neural networks generalize across noise levels and problem settings without retraining.
- The same denoising network can be used for deblurring, demosaicking, and super-resolution.

Performance:

- Competitive **PSNR (Peak Signal-to-Noise Ratio)** compared to problem-specific methods.
- Faster runtime due to GPU-based denoising.

Example:

- **Image Demosaicking:**

$$\min_u \frac{\alpha}{2}\|Au - f\|^2 + R(u),$$

where $R(u)$ is regularization handled by a denoising network.

Audio De-clipping: Restoring clipped audio signals by solving an optimization problem.

Problem Formulation:

$$\min_x \frac{1}{2}\|y - Ax\|^2 + \lambda\|x\|_1,$$

where:

- y: Observed clipped audio signal.
- A: Linear operator (e.g., a matrix modeling the clipping process).
- $\|x\|_1$: Sparse regularization promoting a clean audio signal.

Proximal Gradient Method:

- Alternates between:
  - Solving the least squares data fidelity term.
  - Applying the proximal operator for sparsity (**soft-thresholding**).
- Ensures stable reconstruction of the unclipped signal.

Sources

- Proximal Algorithms - Neal Parikh Stanford University - for general overview of proximal algorithms
- Learning Proximal Operators: Using Denoising Networks for Regularizing Inverse Imaging Problems - Tim Meinhardt - Image problems mentioned on slide 11.
- Proximal gradient algorithms: Applications in signal processing - Niccolo Antonello - Audio problems mentioned on slide 12.

Thank for very much for listening (: