Proximal Gradient Methods for Convex Optimization

Lecture Notes on Iterative Soft Thresholding and Sparse Recovery

Course: Advanced Optimization Theory

July 16, 2025

Contents

1 Introduction to Proximal Methods

1.1 Motivation and Context

The fundamental challenge in modern optimization arises when dealing with composite objective functions of the form:

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is smooth and convex (possessing continuous derivatives), while $g : \mathbb{R}^n \to \mathbb{R}$ is convex but potentially non-smooth. This formulation encompasses a vast array of practical problems in signal processing, machine learning, and statistical inference.

The classical gradient descent method, while elegant and computationally efficient for smooth functions, fails catastrophically when confronted with non-differentiable terms. Consider the paradigmatic example of ℓ_1-regularized least squares (basis pursuit denoising):

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{2}$$

where the ℓ_1 norm $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$ induces sparsity but lacks differentiability at the origin.

1.2 Historical Development and Applications

The proximal gradient framework, pioneered by Moreau in the 1960s and extensively developed by Rockafellar, provides an elegant resolution to this dilemma. The method has found widespread applications in:

- **Compressed Sensing**: Recovery of sparse signals from underdetermined linear measurements
- **Image Processing**: Total variation denoising, image reconstruction, and restoration
- **Machine Learning**: Feature selection, sparse coding, and regularized regression
- **Portfolio Optimization**: Risk-constrained asset allocation with transaction costs

The key insight is to decouple the smooth and non-smooth components, handling each with its appropriate mathematical machinery.

2 Mathematical Foundations

2.1 Convexity and Smoothness

Definition 2.1 (Convex Function)**.** A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is *convex* if its domain $\text{dom}(f)$ is a convex set and for all $\mathbf{x}, \mathbf{y} \in \text{dom}(f)$ and $\theta \in [0, 1]$:

$$f(\theta\mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta)f(\mathbf{y}) \tag{3}$$

Definition 2.2 (L-Smooth Function)**.** A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is *L-smooth* if its gradient is Lipschitz continuous with constant $L > 0$:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \tag{4}$$

The Lipschitz constant L quantifies the maximum rate of change of the gradient, providing crucial information for algorithm design. For twice-differentiable functions, L corresponds to the maximum eigenvalue of the Hessian matrix over the domain.

2.2 The Descent Lemma and Majorization

Lemma 2.1 (Descent Lemma)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-smooth. Then for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$:

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 \tag{5}$$

Proof Sketch. By the fundamental theorem of calculus and Cauchy-Schwarz inequality:

$$f(\mathbf{y}) - f(\mathbf{x}) = \int_0^1 \langle \nabla f(\mathbf{x} + t(\mathbf{y} - \mathbf{x})), \mathbf{y} - \mathbf{x} \rangle dt \tag{6}$$

$$= \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \int_0^1 \langle \nabla f(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) - \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle dt \tag{7}$$

Applying the Lipschitz condition on ∇f completes the proof. □

This lemma establishes that any L-smooth function can be globally upper-bounded by a quadratic function, leading to the concept of *majorization*.

Definition 2.3 (Majorizer)**.** A function $Q(\mathbf{y}; \mathbf{x})$ is a *majorizer* of f at $\mathbf{x}$ if:

1. $Q(\mathbf{y}; \mathbf{x}) \geq f(\mathbf{y})$ for all $\mathbf{y} \in \mathbb{R}^n$ (global upper bound)
2. $Q(\mathbf{x}; \mathbf{x}) = f(\mathbf{x})$ (tangency condition)

For L-smooth functions, the descent lemma provides an explicit majorizer:

$$Q(\mathbf{y}; \mathbf{x}) = f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 \tag{8}$$

1. Initialize $\mathbf{x}^0 \in \mathbb{R}^n$
2. For $k = 0, 1, 2, \ldots$:
   (a) Construct majorizer $Q(\mathbf{y}; \mathbf{x}^k)$ of f at $\mathbf{x}^k$
   (b) Update: $\mathbf{x}^{k+1} = \arg\min_{\mathbf{y}} Q(\mathbf{y}; \mathbf{x}^k)$

2.3 The Majorize-Minimize Principle

The majorize-minimize (MM) algorithm generates a sequence $\{\mathbf{x}^k\}$ by iteratively minimizing majorizers:

Theorem 2.2 (Monotonic Descent)**.** The MM algorithm produces a monotonically decreasing sequence of function values:

$$f(\mathbf{x}^{k+1}) \leq f(\mathbf{x}^k), \quad \forall k \geq 0 \tag{9}$$

Proof. By the majorization property and the definition of $\mathbf{x}^{k+1}$:

$$f(\mathbf{x}^{k+1}) \leq Q(\mathbf{x}^{k+1}; \mathbf{x}^k) \leq Q(\mathbf{x}^k; \mathbf{x}^k) = f(\mathbf{x}^k) \tag{10}$$

□

When applied to the quadratic majorizer (8), the MM update becomes:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{y}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{y} - \mathbf{x}^k \rangle + \frac{L}{2} \|\mathbf{y} - \mathbf{x}^k\|_2^2 \right\} \tag{11}$$

$$= \mathbf{x}^k - \frac{1}{L} \nabla f(\mathbf{x}^k) \tag{12}$$

recovering gradient descent with step size $\gamma = 1/L$.

3 Proximal Operators and Their Properties

3.1 Definition and Geometric Interpretation

Definition 3.1 (Proximal Operator)**.** The *proximal operator* of a function $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is defined as:

$$\text{prox}_g(\mathbf{v}) = \arg\min_{\mathbf{x} \in \mathbb{R}^n} \left\{ g(\mathbf{x}) + \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2 \right\} \tag{13}$$

The proximal operator admits several interpretations:

- **Geometric**: Find the point minimizing g that is closest to $\mathbf{v}$ in Euclidean distance
- **Regularization**: Balance between minimizing g and staying near $\mathbf{v}$
- **Implicit gradient step**: Generalization of gradient descent to non-smooth functions

Remark 3.1. For smooth functions, the proximal operator approximates a gradient step. Indeed, if g is differentiable with L-Lipschitz gradient, then:

$$\text{prox}_{\gamma g}(\mathbf{v}) \approx \mathbf{v} - \gamma \nabla g(\mathbf{v}) \quad \text{for small } \gamma \tag{14}$$

3.2 Scaled Proximal Operators

For algorithmic purposes, we often work with scaled versions:

Definition 3.2 (Scaled Proximal Operator)**.** For $\gamma > 0$, the scaled proximal operator is:

$$\text{prox}_{\gamma g}(\mathbf{v}) = \arg\min_{\mathbf{x} \in \mathbb{R}^n} \left\{ g(\mathbf{x}) + \frac{1}{2\gamma}\|\mathbf{x} - \mathbf{v}\|_2^2 \right\} \tag{15}$$

The scaling parameter γ controls the trade-off between minimizing g and proximity to $\mathbf{v}$. Larger γ values allow solutions farther from $\mathbf{v}$.

3.3 Properties of Proximal Operators

Theorem 3.1 (Fundamental Properties)**.** Let $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be a proper, closed, convex function. Then:

1. **Existence and Uniqueness**: $\text{prox}_g(\mathbf{v})$ exists and is unique for all $\mathbf{v} \in \mathbb{R}^n$
2. **Firm Non-expansiveness**: For all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$:

$$\|\text{prox}_g(\mathbf{u}) - \text{prox}_g(\mathbf{v})\|_2^2 + \|(\mathbf{u} - \text{prox}_g(\mathbf{u})) - (\mathbf{v} - \text{prox}_g(\mathbf{v}))\|_2^2 \leq \|\mathbf{u} - \mathbf{v}\|_2^2 \tag{16}$$

3. **Moreau Decomposition**:

$$\mathbf{v} = \text{prox}_g(\mathbf{v}) + \text{prox}_{g^*}(\mathbf{v}) \tag{17}$$

where g^* is the Fenchel conjugate of g

4 The Proximal Gradient Algorithm

4.1 Algorithm Derivation

Consider the composite optimization problem (1). Applying the MM principle with the majorizer for f while keeping g unchanged:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{y}} \left\{ Q(\mathbf{y}; \mathbf{x}^k) + g(\mathbf{y}) \right\} \tag{18}$$

$$= \arg\min_{\mathbf{y}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{y} - \mathbf{x}^k \rangle + \frac{L}{2} \|\mathbf{y} - \mathbf{x}^k\|_2^2 + g(\mathbf{y}) \right\} \tag{19}$$

Proposition 4.1 (Equivalent Formulation)**.** The update can be rewritten as:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{y}} \left\{ g(\mathbf{y}) + \frac{L}{2} \|\mathbf{y} - (\mathbf{x}^k - \frac{1}{L} \nabla f(\mathbf{x}^k))\|_2^2 \right\} \tag{20}$$

Proof. Expanding the quadratic term and using the fact that terms independent of $\mathbf{y}$ don't affect the minimizer:

$$\arg\min_{\mathbf{y}} \left\{ g(\mathbf{y}) + \langle \nabla f(\mathbf{x}^k), \mathbf{y} - \mathbf{x}^k \rangle + \frac{L}{2} \|\mathbf{y} - \mathbf{x}^k\|_2^2 \right\} \tag{21}$$

$$= \arg\min_{\mathbf{y}} \left\{ g(\mathbf{y}) + \frac{L}{2} \left[\|\mathbf{y}\|_2^2 - 2\langle \mathbf{y}, \mathbf{x}^k - \frac{1}{L} \nabla f(\mathbf{x}^k) \rangle + \|\mathbf{x}^k\|_2^2 \right] \right\} \tag{22}$$

$$= \arg\min_{\mathbf{y}} \left\{ g(\mathbf{y}) + \frac{L}{2} \|\mathbf{y} - (\mathbf{x}^k - \frac{1}{L} \nabla f(\mathbf{x}^k))\|_2^2 \right\} \tag{23}$$

□

This leads to the proximal gradient algorithm:

---

1. Initialize $\mathbf{x}^0 \in \mathbb{R}^n$, step size $\gamma = 1/L$
2. For $k = 0, 1, 2, \ldots$:
   (a) Gradient step: $\mathbf{z}^k = \mathbf{x}^k - \gamma \nabla f(\mathbf{x}^k)$
   (b) Proximal step: $\mathbf{x}^{k+1} = \text{prox}_{\gamma g}(\mathbf{z}^k)$

4.2 Convergence Analysis

Theorem 4.2 (Convergence Rate)**.** Let f be L-smooth and convex, g be closed and convex, and $F = f + g$ have a minimizer $\mathbf{x}^*$. Then the proximal gradient method with $\gamma = 1/L$ satisfies:

$$F(\mathbf{x}^k) - F(\mathbf{x}^*) \leq \frac{\|\mathbf{x}^0 - \mathbf{x}^*\|_2^2}{2\gamma k} = \frac{L\|\mathbf{x}^0 - \mathbf{x}^*\|_2^2}{2k} \tag{24}$$

This $O(1/k)$ convergence rate matches gradient descent for smooth problems, demonstrating that non-smoothness doesn't degrade the convergence rate.

4.3 Practical Step Size Selection

In practice, the Lipschitz constant L is often unknown. Common strategies include:

1. **Backtracking Line Search**: Start with an estimate $\hat{L}$ and increase until the descent condition is satisfied
2. **Adaptive Step Sizes**: Use algorithms like FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) with momentum
3. **Conservative Estimates**: Use matrix norm bounds, e.g., $L \leq \|\mathbf{D}^T\mathbf{D}\|_2$ for quadratic functions

5 Computing Proximal Operators: The ℓ_1 Case

5.1 Derivation of the Soft Thresholding Operator

We now derive the proximal operator for the ℓ_1 norm, which is fundamental for sparse optimization.

Theorem 5.1 (Proximal Operator of ℓ_1 Norm)**.** The proximal operator of $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ is given by the *soft thresholding operator*:

$$[\text{prox}_{\lambda\|\cdot\|_1}(\mathbf{v})]_i = \begin{cases} v_i - \lambda & \text{if } v_i > \lambda \\ 0 & \text{if } |v_i| \leq \lambda \\ v_i + \lambda & \text{if } v_i < -\lambda \end{cases} \tag{25}$$

Proof. The proximal operator computation:

$$\text{prox}_{\lambda\|\cdot\|_1}(\mathbf{v}) = \arg\min_{\mathbf{x}} \left\{ \lambda \sum_{i=1}^{n} |x_i| + \frac{1}{2} \sum_{i=1}^{n} (x_i - v_i)^2 \right\} \tag{26}$$

Since the objective decomposes across coordinates, we can minimize separately for each i:

$$[\text{prox}_{\lambda\|\cdot\|_1}(\mathbf{v})]_i = \arg\min_{x_i \in \mathbb{R}} \left\{ \lambda|x_i| + \frac{1}{2}(x_i - v_i)^2 \right\} \tag{27}$$

Define $h(x) = \lambda|x| + \frac{1}{2}(x - v)^2$ for scalar x and v. We analyze three cases:
Case 1: $x > 0$. Then $h(x) = \lambda x + \frac{1}{2}(x - v)^2$.

$$\frac{dh}{dx} = \lambda + (x - v) = 0 \tag{28}$$

$$\Rightarrow x^* = v - \lambda \tag{29}$$

This is valid only if $x^* > 0$, i.e., $v > \lambda$.
Case 2: $x < 0$. Then $h(x) = -\lambda x + \frac{1}{2}(x - v)^2$.

$$\frac{dh}{dx} = -\lambda + (x - v) = 0 \tag{30}$$

$$\Rightarrow x^* = v + \lambda \tag{31}$$

This is valid only if $x^* < 0$, i.e., $v < -\lambda$.
Case 3: $x = 0$. The subdifferential condition at $x = 0$ requires:

$$0 \in \partial h(0) = [-\lambda, \lambda] - v \tag{32}$$

This holds if and only if $|v| \leq \lambda$. □

5.2 Compact Representations

The soft thresholding operator admits several useful representations:

Proposition 5.2 (Alternative Forms)**.**

$$\text{prox}_{\lambda\|\cdot\|_1}(\mathbf{v}) = \text{sign}(\mathbf{v}) \odot \max(|\mathbf{v}| - \lambda, 0) \tag{33}$$

$$= \mathbf{v} - \lambda \cdot \text{proj}_{[-1,1]}(\mathbf{v}/\lambda) \tag{34}$$

where $\odot$ denotes element-wise multiplication and $\text{proj}_{[-1,1]}$ is projection onto the ℓ_∞ ball.

5.3 Comparison with Hard Thresholding

The soft thresholding operator exhibits markedly different behavior from the hard thresholding operator used in orthogonal matching pursuit:

Definition 5.1 (Hard Thresholding)**.**

$$[\text{hard}_\lambda(\mathbf{v})]_i = \begin{cases} v_i & \text{if } |v_i| > \lambda \\ 0 & \text{if } |v_i| \leq \lambda \end{cases} \tag{35}$$

Key differences:

- **Continuity**: Soft thresholding is continuous; hard thresholding has jump discontinuities
- **Bias**: Soft thresholding shrinks all coefficients by λ; hard thresholding preserves large coefficients unchanged
- **Optimization**: Soft thresholding arises from convex optimization; hard thresholding from combinatorial selection

6 The Iterative Soft Thresholding Algorithm (ISTA)

6.1 Algorithm Specification

Combining the proximal gradient framework with the soft thresholding operator yields:

Input: Matrix $\mathbf{D} \in \mathbb{R}^{m \times n}$, observations $\mathbf{b} \in \mathbb{R}^m$, regularization parameter $\lambda > 0$
Output: Sparse solution $\mathbf{x}^* \approx \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1$

1. Precompute: $\mathbf{D}^T\mathbf{D}$ and $\mathbf{D}^T\mathbf{b}$
2. Set step size: $\gamma = 1/\|\mathbf{D}^T\mathbf{D}\|_2$
3. Initialize: $\mathbf{x}^0 = \mathbf{0}$ (or warm start)
4. For $k = 0, 1, 2, \ldots$ until convergence:
   (a) Gradient step: $\mathbf{z}^k = \mathbf{x}^k - \gamma\mathbf{D}^T(\mathbf{D}\mathbf{x}^k - \mathbf{b})$
   (b) Soft threshold: $\mathbf{x}^{k+1} = \text{soft}_{\gamma\lambda}(\mathbf{z}^k)$

6.2 Computational Complexity

Per iteration:

- Matrix-vector products: $O(mn)$ for $\mathbf{D}\mathbf{x}^k$ and $\mathbf{D}^T(\cdot)$
- Soft thresholding: $O(n)$
- Total: $O(mn)$ per iteration

The precomputation of $\mathbf{D}^T\mathbf{D}$ requires $O(mn^2)$ operations but is amortized over all iterations.

6.3 Convergence Guarantees

Theorem 6.1 (ISTA Convergence)**.** Under the conditions of Theorem 4.2, ISTA produces iterates satisfying:

$$\frac{1}{2}\|\mathbf{D}\mathbf{x}^k - \mathbf{b}\|_2^2 + \lambda\|\mathbf{x}^k\|_1 - F^* \leq \frac{L\|\mathbf{x}^0 - \mathbf{x}^*\|_2^2}{2k} \tag{36}$$

where F^* is the optimal objective value and $L = \|\mathbf{D}^T\mathbf{D}\|_2$.

6.4 Acceleration: FISTA

The Fast ISTA (FISTA) incorporates momentum to achieve $O(1/k^2)$ convergence:

The extrapolation step $\mathbf{y}^{k+1}$ uses information from previous iterates to accelerate convergence, analogous to Nesterov's accelerated gradient method.

Initialize $\mathbf{x}^0 = \mathbf{y}^0$, $t_0 = 1$. For $k = 0, 1, 2, \ldots$:

1. $\mathbf{x}^{k+1} = \text{soft}_{\gamma\lambda}(\mathbf{y}^k - \gamma \mathbf{D}^T(\mathbf{D}\mathbf{y}^k - \mathbf{b}))$
2. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
3. $\mathbf{y}^{k+1} = \mathbf{x}^{k+1} + \frac{t_k - 1}{t_{k+1}}(\mathbf{x}^{k+1} - \mathbf{x}^k)$

7 Extensions and Advanced Topics

7.1 Other Proximal Operators

The proximal gradient framework extends beyond ℓ_1 regularization:

Example 7.1 (Common Proximal Operators)**.** 1. ℓ_2 **norm (group sparsity)**:

$$\text{prox}_{\lambda\|\cdot\|_2}(\mathbf{v}) = \max\left(1 - \frac{\lambda}{\|\mathbf{v}\|_2}, 0\right)\mathbf{v} \tag{37}$$

2. **Indicator function of convex set** C:

$$\text{prox}_{i_C}(\mathbf{v}) = \text{proj}_C(\mathbf{v}) \quad \text{(Euclidean projection)} \tag{38}$$

3. **Nuclear norm (low-rank matrices)**:

$$\text{prox}_{\lambda\|\cdot\|_*}(\mathbf{V}) = \mathbf{U}\text{soft}_\lambda(\mathbf{\Sigma})\mathbf{V}^T \tag{39}$$

where $\mathbf{V} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the SVD.

7.2 Composite Minimization Beyond Sparsity

The framework $\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x})$ accommodates diverse applications:

- **Total Variation Denoising**: $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{b}\|_2^2$, $g(\mathbf{x}) = \lambda\|\nabla\mathbf{x}\|_1$
- **Matrix Completion**: $f(\mathbf{X}) = \frac{1}{2}\|\mathcal{P}_\Omega(\mathbf{X}) - \mathbf{Y}\|_F^2$, $g(\mathbf{X}) = \lambda\|\mathbf{X}\|_*$
- **Robust PCA**: $\min_{\mathbf{L},\mathbf{S}}\|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1$ s.t. $\mathbf{L} + \mathbf{S} = \mathbf{M}$

7.3 Practical Considerations

Remark 7.1 (Implementation Tips)**.** 1. **Warm Starting**: Use solutions from related problems or previous iterations

2. **Adaptive Parameters**: Adjust λ using cross-validation or stability selection
3. **Stopping Criteria**: Monitor relative change in objective: $\frac{|F^{k+1} - F^k|}{1 + |F^k|} < \epsilon$
4. **Preconditioning**: For ill-conditioned $\mathbf{D}$, use diagonal or approximate inverse preconditioners

7.4 Connections to Other Methods

The proximal gradient framework unifies several algorithmic paradigms:

Theorem 7.1 (Special Cases)**.** The proximal gradient method reduces to:

1. **Gradient Descent**: When $g \equiv 0$
2. **Proximal Point Algorithm**: When $f \equiv 0$
3. **Projected Gradient**: When $g = i_C$ (indicator of convex set C)
4. **Forward-Backward Splitting**: General formulation for f smooth, g non-smooth

8 Numerical Examples and Applications

8.1 Sparse Signal Recovery

Consider recovering a sparse signal $\mathbf{x}^* \in \mathbb{R}^{1000}$ with $k = 50$ non-zero entries from $m = 300$ noisy measurements:

$$\mathbf{b} = \mathbf{D}\mathbf{x}^* + \epsilon \tag{40}$$

where $\mathbf{D} \in \mathbb{R}^{300\times1000}$ has i.i.d. Gaussian entries and $\epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$.

Example 8.1 (ISTA Performance)**.** Key observations from numerical experiments:

- **Parameter Selection**: Optimal $\lambda \approx \sigma\sqrt{2\log n}$ (universal threshold)
- **Convergence**: ϵ-accurate solution in $O(L/\epsilon)$ iterations
- **Support Recovery**: Exact support recovery when λ properly chosen and signal sufficiently sparse

8.2 Image Denoising with Total Variation

For image denoising, we solve:

$$\min_{\mathbf{X}\in\mathbb{R}^{m\times n}} \frac{1}{2}\|\mathbf{X} - \mathbf{Y}\|_F^2 + \lambda \mathrm{TV}(\mathbf{X}) \tag{41}$$

where $\mathrm{TV}(\mathbf{X}) = \sum_{i,j} \sqrt{(X_{i+1,j} - X_{i,j})^2 + (X_{i,j+1} - X_{i,j})^2}$ is the isotropic total variation.

The proximal operator of TV requires solving a non-trivial optimization problem, often handled by:

- Dual formulation and projection onto the dual ball
- Chambolle's algorithm for exact computation
- Split Bregman methods for approximate solutions

8.3 Portfolio Optimization with Transaction Costs

In financial applications, we optimize portfolio weights $\mathbf{w} \in \mathbb{R}^n$ subject to transaction costs:

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{\Sigma}\mathbf{w} - \mu^T\mathbf{w} + \lambda\|\mathbf{w} - \mathbf{w}_0\|_1 \tag{42}$$

where $\mathbf{\Sigma}$ is the covariance matrix, μ expected returns, and $\mathbf{w}_0$ current holdings.

The ℓ_1 penalty on $\mathbf{w} - \mathbf{w}_0$ models proportional transaction costs, and ISTA provides an efficient solution method that naturally produces sparse portfolio updates.

9 Theoretical Guarantees and Optimality

9.1 Global Convergence for Convex Problems

Theorem 9.1 (Global Optimality)**.** For convex f and g, any accumulation point of the ISTA sequence $\{\mathbf{x}^k\}$ is a global minimizer of $F = f + g$.

Proof Sketch. The proof relies on three key facts:

1. The objective sequence $\{F(\mathbf{x}^k)\}$ is monotonically decreasing and bounded below
2. The iterates satisfy a sufficient decrease condition
3. The subdifferential of F at accumulation points contains zero

□

9.2 Linear Convergence under Strong Convexity

When additional structure is present, convergence accelerates dramatically:

Definition 9.1 (Strong Convexity)**.** A function f is μ-strongly convex if:

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 \tag{43}$$

Theorem 9.2 (Linear Convergence)**.** If f is μ-strongly convex and L-smooth, then ISTA with $\gamma = 1/L$ achieves:

$$\|\mathbf{x}^k - \mathbf{x}^*\|_2 \leq \left(1 - \frac{\mu}{L}\right)^k \|\mathbf{x}^0 - \mathbf{x}^*\|_2 \tag{44}$$

The condition number $\kappa = L/\mu$ determines the convergence rate, with smaller κ yielding faster convergence.

9.3 Recovery Guarantees for Sparse Signals

Under appropriate conditions on the measurement matrix $\mathbf{D}$, ISTA provably recovers sparse signals:

Definition 9.2 (Restricted Isometry Property)**.** Matrix $\mathbf{D}$ satisfies the RIP of order k with constant δ_k if:

$$(1 - \delta_k)\|\mathbf{x}\|_2^2 \leq \|\mathbf{D}\mathbf{x}\|_2^2 \leq (1 + \delta_k)\|\mathbf{x}\|_2^2 \tag{45}$$

for all k-sparse vectors $\mathbf{x}$.

Theorem 9.3 (Sparse Recovery)**.** If $\mathbf{D}$ satisfies RIP with $\delta_{2k} < \sqrt{2}-1$ and λ is appropriately chosen, then the ISTA solution $\mathbf{x}^*$ satisfies:

$$\|\mathbf{x}^* - \mathbf{x}_0\|_2 \leq C_1 \cdot \frac{\|\mathbf{x}_0 - \mathbf{x}_{0,k}\|_1}{\sqrt{k}} + C_2 \cdot \|\epsilon\|_2 \tag{46}$$

where $\mathbf{x}_{0,k}$ is the best k-sparse approximation to $\mathbf{x}_0$.

10 Summary and Future Directions

10.1 Key Takeaways

The proximal gradient framework provides a powerful and flexible approach to composite optimization:

- **Generality**: Handles arbitrary combinations of smooth and non-smooth convex functions
- **Efficiency**: Computational cost comparable to gradient descent
- **Modularity**: Proximal operators can be computed independently and reused
- **Convergence**: Achieves optimal rates for first-order methods
- **Practicality**: Simple implementation with few tuning parameters

10.2 Current Research Directions

Active areas of investigation include:

1. **Stochastic Variants**: Proximal stochastic gradient methods for large-scale learning
2. **Non-convex Extensions**: Proximal methods for weakly convex and difference-of-convex functions
3. **Distributed Algorithms**: Consensus-based proximal methods for decentralized optimization
4. **Adaptive Methods**: Learning problem-specific metrics and preconditioners
5. **Proximal Newton Methods**: Second-order proximal algorithms for faster convergence

10.3 Software and Implementation

Modern optimization packages implementing proximal methods include:

- `CVXPY`: High-level convex optimization modeling
- `ProximalOperators.jl`: Julia library with extensive proximal operator collection
- `TFOCS`: MATLAB toolbox for first-order conic solvers
- `scikit-learn`: Includes ISTA/FISTA for Lasso and related problems

10.4 Concluding Remarks

The elegance of proximal gradient methods lies in their ability to decompose complex optimization problems into simpler subproblems, each solved with the most appropriate technique. This divide-and-conquer strategy, combined with rigorous convergence guarantees, makes proximal methods indispensable tools in the modern optimization toolkit.
