From Gradient Descent to ISTA, FISTA, and ADMM for Sparse Signal Processing and Baseline Estimation

Contents

1 Motivation: Inverse Problems, Sparsity, and BEADS

In many signal processing problems we observe a signal

$$y \in \mathbb{R}^n$$

that is a noisy, distorted version of some underlying structure we care about: spikes, edges, a slowly-varying baseline, etc. A generic linear model is

$$y = Hx + \varepsilon, \tag{1}$$

where

- $x \in \mathbb{R}^p$ is the unknown signal/parameter vector,
- $H \in \mathbb{R}^{n \times p}$ is a known linear operator (e.g. convolution, sampling, mixing),
- ε is noise (often modeled as Gaussian).

Reminder (Linear operator / matrix). A matrix H represents a linear map $x \mapsto Hx$, i.e. $H(\alpha x + \beta z) = \alpha Hx + \beta Hz$ for all x, z and scalars α, β.

In BEADS–type problems (baseline estimation and denoising with sparsity), one often models

$$y = b + s + w,$$

where

- b is a smooth or slowly-varying *baseline*,
- s is a *sparse* component (e.g. spikes or events),
- w is noise.

Reminder (Sparse vector). A vector $x \in \mathbb{R}^p$ is called *sparse* if most of its entries are exactly zero, i.e. only few coordinates are nonzero.

A typical optimization model is then

$$\min_{b,s} \underbrace{\frac{1}{2} \|y - b - s\|_2^2}_{\text{data fit}} + \lambda_s \|Ws\|_1 + \lambda_b \|Db\|_1 + \iota_{\mathcal{C}}(b, s), \tag{2}$$

where W, D are linear operators that promote sparsity in appropriate domains (e.g. finite differences for piecewise-smoothness).

Reminder (ℓ_2–norm). For $x \in \mathbb{R}^n$, the Euclidean norm is $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$.

Reminder (ℓ_1–norm). For $x \in \mathbb{R}^n$, the ℓ_1 norm is $\|x\|_1 = \sum_{i=1}^n |x_i|$, often used to promote sparsity.

Reminder (Indicator function). For a set $\mathcal{C}$, $\iota_{\mathcal{C}}(x) = 0$ if $x \in \mathcal{C}$ and $+\infty$ otherwise; this encodes hard constraints $x \in \mathcal{C}$ inside an optimization problem.

Problems such as (2) are convex but *nonsmooth* because of the ℓ_1 terms and indicator constraints. First-order methods like ISTA, FISTA, and ADMM are the workhorses for solving them efficiently in high dimensions.

The goal of this note is to build up, from undergraduate-level gradient descent, the mathematical machinery needed to understand:

- ISTA (Iterative Shrinkage-Thresholding Algorithm),
- FISTA (Fast ISTA, Nesterov-accelerated),
- ADMM (Alternating Direction Method of Multipliers),

in the context of sparse signal processing and baseline estimation.

2 Preliminaries: Vectors, Norms, Convexity, and Prox

2.1 Basic linear algebra and norms

We work in finite-dimensional real vector spaces $\mathbb{R}^n$.
Reminder (Vector space). $\mathbb{R}^n$ with componentwise addition and scalar multiplication is a vector space; linear combinations $\alpha x + \beta y$ stay inside $\mathbb{R}^n$.

Inner products and norms. For $x, y \in \mathbb{R}^n$, the standard inner product is

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i.$$

Reminder (Inner product). An inner product is a bilinear, symmetric, positive-definite map $\langle \cdot, \cdot \rangle$ that generalizes the dot product and induces a notion of angle and length.

From the inner product we get the Euclidean norm

$$\|x\|_2 = \sqrt{\langle x, x \rangle}.$$

More generally, for $p \geq 1$,

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p \Big)^{1/p}$$

defines the ℓ_p norm.
Reminder (Norm). A norm $\|\cdot\|$ satisfies: $\|x\| \geq 0$, $\|x\| = 0 \Leftrightarrow x = 0$, $\|\alpha x\| = |\alpha| \, \|x\|$, and triangle inequality $\|x + y\| \leq \|x\| + \|y\|$.

For matrices $A \in \mathbb{R}^{m \times n}$ we use the *operator norm* induced by ℓ_2:

$$\|A\|_2 = \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}.$$

Reminder (Spectral norm). The spectral norm $\|A\|_2$ is the largest singular value of A, i.e. the square root of the largest eigenvalue of $A^\top A$.

2.2 Convex sets and convex functions

Definition 1 (Convex set)**.** *A set $C \subset \mathbb{R}^n$ is convex if for all $x, y \in C$ and all $\theta \in [0,1]$, we have*

$$\theta x + (1-\theta)y \in C.$$

Reminder. Convex sets contain the line segments between any two of their points.

Definition 2 (Convex function)**.** *A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is convex if for all $x, y \in \operatorname{dom} f$ and $\theta \in [0,1]$,*

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta) f(y).$$

Reminder. Convex functions have no "bad" local minima; any local minimum is global.

Definition 3 (Lipschitz continuous gradient)**.** *Let $g : \mathbb{R}^n \to \mathbb{R}$ be differentiable. Its gradient ∇g is L–Lipschitz if there exists $L > 0$ such that*

$$\|\nabla g(x) - \nabla g(y)\|_2 \le L \|x - y\|_2 \quad \forall x, y.$$

Reminder. Lipschitz continuity of ∇g means the gradient does not change too fast; L controls the curvature of g.

One important consequence (used in ISTA/FISTA) is the *quadratic upper bound*:

$$g(x) \le g(y) + \langle \nabla g(y), x - y \rangle + \frac{L}{2} \|x - y\|_2^2 \quad \forall x, y.$$

2.3 Subgradients and nonsmooth functions

For nonsmooth convex functions we use *subgradients*.

Definition 4 (Subgradient)**.** *Let $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be convex. A vector $s \in \mathbb{R}^n$ is a subgradient of h at x if*

$$h(z) \ge h(x) + \langle s, z - x \rangle \quad \forall z.$$

The set of all subgradients at x is the subdifferential *$\partial h(x)$.*

Reminder. Subgradients generalize gradients to nonsmooth convex functions; at differentiable points, $\partial h(x) = \{\nabla h(x)\}$.

Example: for $h(x) = \|x\|_1$, the ith component of any subgradient s at x satisfies

$$s_i \in \begin{cases} \{+1\}, & x_i > 0, \\ [-1, +1], & x_i = 0, \\ \{-1\}, & x_i < 0. \end{cases}$$

2.4 Proximal operators

Central objects for ISTA/FISTA/ADMM are *proximal operators*.

Definition 5 (Proximal operator)**.** *Let $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ be convex, proper, and lower semicontinuous. For $\gamma > 0$, the proximal operator of γh is*

$$\operatorname{prox}_{\gamma h}(v) := \operatorname*{arg\,min}_{x \in \mathbb{R}^n} \left\{ h(x) + \frac{1}{2\gamma} \|x - v\|_2^2 \right\}.$$

Reminder. The prox of h at v is a compromise between being close to v and having small $h(x)$.
Reminder (arg min). $\arg\min_x \phi(x)$ denotes the set of points x that minimize ϕ; when ϕ is strictly convex, there is a unique minimizer.

The proximal operator can be viewed as a generalized projection: if h is the indicator ι_C of a closed convex set C, then

$$\mathrm{prox}_{\gamma\iota_C}(v) = P_C(v),$$

the Euclidean projection of v onto C.
Reminder (Projection). The projection $P_C(v)$ is the point in C closest to v in Euclidean distance.

Soft-thresholding as a prox. For $h(x) = \lambda \|x\|_1$, the prox has a closed form:

$$\mathrm{prox}_{\gamma\lambda\|\cdot\|_1}(v) = S_{\gamma\lambda}(v),$$

where S_τ is the *soft-thresholding (shrinkage) operator* defined componentwise by

$$[S_\tau(v)]_i = \mathrm{sgn}(v_i)\ \max\{|v_i| - \tau,\ 0\}.$$

Reminder (Sign function). $\mathrm{sgn}(t) = 1$ if $t > 0$, $\mathrm{sgn}(t) = -1$ if $t < 0$, and any value in $[-1, 1]$ if $t = 0$ (choice at 0 does not matter in practice).

Soft-thresholding is the key nonlinearity in ISTA/FISTA for ℓ_1-regularized problems.

3 Composite Optimization Formulation

ISTA and FISTA solve problems of the form

$$\min_{x\in\mathbb{R}^p} F(x) := g(x) + h(x), \tag{3}$$

with the following structure:

- $g : \mathbb{R}^p \to \mathbb{R}$ is convex, differentiable, and has an L–Lipschitz gradient;
- $h : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is convex, possibly nonsmooth, and $\mathrm{prox}_{\gamma h}$ is easy to compute.

Reminder (Composite problem). A composite problem splits the objective into a smooth part g and a nonsmooth but prox-friendly part h.

Example: LASSO. Given $A \in \mathbb{R}^{n\times p}$, $y \in \mathbb{R}^n$, and $\lambda > 0$,

$$\min_{x\in\mathbb{R}^p} \frac{1}{2} \|Ax - y\|_2^2 + \lambda \|x\|_1 . \tag{4}$$

Here:

$$g(x) = \frac{1}{2} \|Ax - y\|_2^2, \qquad h(x) = \lambda \|x\|_1 .$$

We have

$$\nabla g(x) = A^\top (Ax - y),$$

and ∇g is Lipschitz with constant $L = \left\|A^\top A\right\|_2$.
Reminder (Adjoint / transpose). For real matrices, $A^\top$ is the transpose; it is the adjoint with respect to the standard inner product: $\langle Ax, y\rangle = \left\langle x, A^\top y\right\rangle$.

Example: Analysis-sparse denoising. Take a 1D signal $x \in \mathbb{R}^n$, observed as $y = x + w$. Let $D \in \mathbb{R}^{(n-1)\times n}$ be the first-difference operator

$$(Dx)_i = x_{i+1} - x_i.$$

Then the problem

$$\min_x \frac{1}{2} \|x - y\|_2^2 + \lambda \|Dx\|_1$$

is total variation-type denoising in analysis form.

In BEADS-style models, x may be a concatenation of components b, s, and D may be higher-order differences, but the composite form is similar.

4 Gradient Descent as a Starting Point

Before ISTA, recall vanilla gradient descent for smooth g:

$$\min_x g(x).$$

The iteration is

$$x^{k+1} = x^k - \alpha \nabla g(x^k), \tag{5}$$

where $\alpha > 0$ is a step size (learning rate).

Reminder (Gradient). For differentiable $g : \mathbb{R}^n \to \mathbb{R}$, the gradient $\nabla g(x)$ is the vector of partial derivatives and points in the direction of steepest increase of g.

If ∇g is L–Lipschitz and $0 < \alpha < 2/L$, gradient descent converges to a minimizer of g.

The problem: gradient descent cannot handle nonsmooth terms like $\lambda \|x\|_1$ or constraints encoded in $\iota_{\mathcal{C}}$ directly. We need a way to combine gradient steps with nondifferentiable regularizers. This is exactly what proximal gradient methods (ISTA/FISTA) do.

5 ISTA: Iterative Shrinkage-Thresholding Algorithm

5.1 Derivation from proximal gradient

Given (3), we want to exploit the Lipschitz property of ∇g:

$$g(x) \le g(y) + \langle \nabla g(y), x - y \rangle + \frac{L}{2} \|x - y\|_2^2. \tag{6}$$

Fix a point x^k. Define a local quadratic upper bound around x^k:

$$Q_L(x; x^k) := g(x^k) + \left\langle \nabla g(x^k), x - x^k \right\rangle + \frac{L}{2} \left\| x - x^k \right\|_2^2.$$

Reminder (Majorization). A function $Q(x)$ *majorizes* $g(x)$ if $Q(x) \ge g(x)$ for all x, with equality at some point; minimizing Q then gives a descent direction for g.

We consider the surrogate

$$x^{k+1} = \arg\min_x \left\{ Q_L(x; x^k) + h(x) \right\}.$$

Dropping constants independent of x and rescaling, this is equivalent to

$$
\begin{aligned}
x^{k+1} &= \arg\min_x \left\{ \left\langle \nabla g(x^k), x - x^k \right\rangle + \frac{L}{2} \left\| x - x^k \right\|_2^2 + h(x) \right\} \\
&= \arg\min_x \left\{ \frac{L}{2} \left\| x - \left(x^k - \tfrac{1}{L} \nabla g(x^k)\right) \right\|_2^2 + h(x) \right\}.
\end{aligned}
$$

Therefore

$$
x^{k+1} = \mathrm{prox}_{\frac{1}{L}h} \left(x^k - \frac{1}{L} \nabla g(x^k) \right). \tag{7}
$$

Reminder (Proximal gradient step). A proximal gradient step first takes a gradient step on g, then applies the prox of h.

More generally, for any $\alpha \in (0, 1/L]$:

$$
x^{k+1} = \mathrm{prox}_{\alpha h} \left(x^k - \alpha \nabla g(x^k) \right). \tag{8}
$$

This is the Iterative Shrinkage-Thresholding Algorithm (ISTA).

5.2 ISTA for LASSO and soft-thresholding

Consider LASSO (4). Then

$$
x^{k+1} = \mathrm{prox}_{\alpha\lambda \|\cdot\|_1} \left(x^k - \alpha A^\top (A x^k - y) \right) = S_{\alpha\lambda}\left(x^k - \alpha A^\top (A x^k - y) \right).
$$

Reminder (Shrinkage operator). The shrinkage operator S_τ pulls each coordinate towards zero by τ and sets it to zero if the magnitude is smaller than τ.

Thus ISTA is:

$$
\boxed{x^{k+1} = S_{\alpha\lambda}\left(x^k - \alpha A^\top (A x^k - y) \right), \qquad 0 < \alpha \leq \frac{1}{L}.} \tag{9}
$$

5.3 Convergence rate of ISTA

Under standard assumptions (convex g, h, g with L–Lipschitz gradient), ISTA satisfies

$$
F(x^k) - F(x^\star) \leq \frac{C}{k}
$$

for some constant C depending on x^0 and $x^\star$, where $x^\star$ is a minimizer of F.
Reminder (Sublinear rate). A convergence rate $O(1/k)$ is called sublinear; roughly, the error shrinks inversely with the iteration count.

FISTA will improve this to $O(1/k^2)$.

5.4 Numerical example: 2D LASSO with ISTA

Consider $A \in \mathbb{R}^{2\times 2}$ and $y \in \mathbb{R}^2$:

$$
A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad y = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \quad \lambda = 0.5.
$$

We solve

$$
\min_{x \in \mathbb{R}^2} \frac{1}{2} \|Ax - y\|_2^2 + \lambda \|x\|_1 .
$$

Compute $A^\top A = \mathrm{diag}(4, 1)$, so

$$L = \left\| A^\top A \right\|_2 = 4.$$

Choose $\alpha = 1/L = 0.25$.
Iteration 0. Initialize $x^0 = (0, 0)^\top$.

$$\nabla g(x^0) = A^\top(Ax^0 - y) = A^\top(0 - y) = -A^\top y = -\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} = -\begin{bmatrix} 6 \\ 1 \end{bmatrix}.$$

Gradient step:

$$u^0 = x^0 - \alpha \nabla g(x^0) = 0 - 0.25(-6, -1)^\top = (1.5, 0.25)^\top.$$

Shrinkage with $\tau = \alpha\lambda = 0.25 \cdot 0.5 = 0.125$:

$$x^1 = S_{0.125}(u^0) = \begin{bmatrix} \mathrm{sgn}(1.5) \max(1.5 - 0.125, 0) \\ \mathrm{sgn}(0.25) \max(0.25 - 0.125, 0) \end{bmatrix} = \begin{bmatrix} 1.375 \\ 0.125 \end{bmatrix}.$$

Iteration 1. Compute

$$Ax^1 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1.375 \\ 0.125 \end{bmatrix} = \begin{bmatrix} 2.75 \\ 0.125 \end{bmatrix}, \quad Ax^1 - y = \begin{bmatrix} -0.25 \\ -0.875 \end{bmatrix}.$$

Then

$$\nabla g(x^1) = A^\top(Ax^1 - y) = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -0.25 \\ -0.875 \end{bmatrix} = \begin{bmatrix} -0.5 \\ -0.875 \end{bmatrix}.$$

Gradient step:

$$u^1 = x^1 - \alpha \nabla g(x^1) = \begin{bmatrix} 1.375 \\ 0.125 \end{bmatrix} - 0.25 \begin{bmatrix} -0.5 \\ -0.875 \end{bmatrix} = \begin{bmatrix} 1.375 + 0.125 \\ 0.125 + 0.21875 \end{bmatrix} = \begin{bmatrix} 1.5 \\ 0.34375 \end{bmatrix}.$$

Shrinkage:

$$x^2 = S_{0.125}(u^1) = \begin{bmatrix} \max(1.5 - 0.125, 0) \\ \max(0.34375 - 0.125, 0) \end{bmatrix} = \begin{bmatrix} 1.375 \\ 0.21875 \end{bmatrix}.$$

Continuing this way, x^k converges to the LASSO solution. This small example illustrates the mechanics of ISTA: gradient step + shrinkage.

6 FISTA: Fast Iterative Shrinkage-Thresholding

ISTA has an $O(1/k)$ convergence rate for the objective value. Nesterov's acceleration can improve this to $O(1/k^2)$. FISTA is the accelerated version of ISTA.

6.1 Nesterov-type acceleration

FISTA maintains two sequences: $\{x^k\}$ (like ISTA) and an auxiliary sequence $\{y^k\}$ that extrapolates momentum from past iterates.
FISTA algorithm (for $F(x) = g(x) + h(x)$).

- Choose $x^0 = x^{-1}$, set $t_0 = 1$.

- For $k = 0, 1, 2, \ldots$:

$$\begin{aligned} y^k &= x^k + \frac{t_k - 1}{t_k}(x^k - x^{k-1}), \\ x^{k+1} &= \operatorname{prox}_{\alpha h}\left(y^k - \alpha \nabla g(y^k)\right), \\ t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2}. \end{aligned}$$

Reminder (Momentum / extrapolation). The term $x^k + \theta_k(x^k - x^{k-1})$ uses a linear combination of the last two iterates to "predict" the next search point and injects momentum into the iteration.

The step size α is typically chosen as in ISTA, e.g. $\alpha = 1/L$.

6.2 Convergence rate

For convex $F = g + h$ with g having L–Lipschitz gradient, FISTA satisfies

$$F(x^k) - F(x^\star) \le \frac{2L \left\| x^0 - x^\star \right\|_2^2}{(k+1)^2}.$$

This is an order-of-magnitude improvement over ISTA: to get error less than ε, ISTA needs $O(1/\varepsilon)$ iterations while FISTA needs $O(1/\sqrt{\varepsilon})$.
Reminder (Big-O notation). $f(k) = O(g(k))$ means there are constants C, k_0 such that $|f(k)| \le C|g(k)|$ for all $k \ge k_0$; it captures asymptotic growth.

6.3 FISTA for LASSO

Using the same LASSO setup, the FISTA update reads

$$x^{k+1} = S_{\alpha\lambda}\big(y^k - \alpha A^\top (Ay^k - y)\big).$$

Momentum is only applied on the smooth part (through y^k), not on the nonlinear shrinkage.

Practical notes.

- FISTA can overshoot; a *monotone* variant resets momentum if $F(x^{k+1}) > F(x^k)$.
- Restarts (setting $t_k = 1$ and $y^k = x^k$) are often used when the method oscillates.

7 ADMM: Alternating Direction Method of Multipliers

ISTA/FISTA handle problems of the form $g(x) + h(x)$. ADMM is useful for more structured problems, especially when we can separate variables.

7.1 Augmented Lagrangian

Consider

$$\min_{x,z} f(x) + g(z) \quad \text{s.t.} \quad Ax + Bz = c \tag{10}$$

with matrices A, B and vector c.

Reminder (Equality-constrained problem). Constraints of the form $Ax + Bz = c$ enforce a linear relationship between x and z.

The augmented Lagrangian is

$$\mathcal{L}_\rho(x, z, u) = f(x) + g(z) + u^\top(Ax + Bz - c) + \frac{\rho}{2} \|Ax + Bz - c\|_2^2,$$

where u is the Lagrange multiplier and $\rho > 0$ is a penalty parameter.

Reminder (Lagrange multiplier). Lagrange multipliers u enforce constraints by penalizing violations $Ax + Bz - c$ in the objective.

Defining the scaled dual variable $w = u/\rho$, the *scaled* augmented Lagrangian becomes

$$\mathcal{L}_\rho(x, z, w) = f(x) + g(z) + \frac{\rho}{2} \|Ax + Bz - c + w\|_2^2 - \frac{\rho}{2} \|w\|_2^2.$$

7.2 ADMM iterations

ADMM alternates minimization over x and z and then updates the dual:

$$x^{k+1} = \arg\min_x \left\{ f(x) + \frac{\rho}{2} \left\| Ax + Bz^k - c + w^k \right\|_2^2 \right\}, \tag{11}$$

$$z^{k+1} = \arg\min_z \left\{ g(z) + \frac{\rho}{2} \left\| Ax^{k+1} + Bz - c + w^k \right\|_2^2 \right\}, \tag{12}$$

$$w^{k+1} = w^k + Ax^{k+1} + Bz^{k+1} - c. \tag{13}$$

Reminder (Alternating minimization). Alternating minimization optimizes over subsets of variables in turn, holding the others fixed.

In many problems, the x- and z-updates have closed forms or reduce to simple linear systems and proximal steps.

7.3 ADMM for analysis-sparse problems

Consider the analysis-sparse problem

$$\min_x \frac{1}{2} \|Hx - y\|_2^2 + \lambda \|Dx\|_1, \tag{14}$$

where D is (for instance) a first- or second-order difference operator.

Introduce $z = Dx$ and rewrite:

$$\min_{x,z} \frac{1}{2} \|Hx - y\|_2^2 + \lambda \|z\|_1 \quad \text{s.t.} \quad z = Dx.$$

This fits (10) with

$$f(x) = \frac{1}{2} \|Hx - y\|_2^2, \quad g(z) = \lambda \|z\|_1, \quad A = D, \quad B = -I, \quad c = 0.$$

The ADMM updates become

$$x^{k+1} = \arg\min_x \left\{ \frac{1}{2} \|Hx - y\|_2^2 + \frac{\rho}{2} \left\| Dx - z^k + w^k \right\|_2^2 \right\}, \tag{15}$$

$$z^{k+1} = \arg\min_z \left\{ \lambda \|z\|_1 + \frac{\rho}{2} \left\| Dx^{k+1} - z + w^k \right\|_2^2 \right\}, \tag{16}$$

$$w^{k+1} = w^k + Dx^{k+1} - z^{k+1}. \tag{17}$$

z-update. The z-update is proximal:

$$z^{k+1} = \operatorname{prox}_{\frac{\lambda}{\rho}\|\cdot\|_1}(Dx^{k+1} + w^k) = S_{\lambda/\rho}(Dx^{k+1} + w^k).$$

x-update. The x-update solves a quadratic problem and has the normal equations

$$(H^\top H + \rho D^\top D)x^{k+1} = H^\top y + \rho D^\top (z^k - w^k).$$

Reminder (Normal equations). For $\min_x \frac{1}{2} \|Ax - b\|_2^2$, the optimality condition is $A^\top Ax = A^\top b$, called the normal equations.

For 1D signals and difference operators D, $D^\top D$ is (block) tridiagonal, so the system can be solved efficiently by banded solvers or even FFTs if H and D are convolution operators under periodic boundary conditions.

7.4 ADMM and BEADS-like models

In BEADS-like decompositions, one often has two components, b and s, with different regularizers:

$$\min_{b,s} \frac{1}{2} \|y - b - s\|_2^2 + \lambda_s \|Ws\|_1 + \lambda_b \|Db\|_1 + \iota_{\mathcal{C}}(b, s),$$

where $\mathcal{C}$ may enforce constraints like $s \geq 0$ (nonnegative spikes) or shape constraints on b.

Introduce auxiliary variables

$$z_s = Ws, \qquad z_b = Db,$$

and constraints $z_s = Ws$, $z_b = Db$, plus $(b, s) \in \mathcal{C}$. ADMM splits this into subproblems:

- a quadratic problem for (b, s) (similar to (15)),
- two shrinkage steps for z_s and z_b,
- projection onto constraints $\mathcal{C}$ if needed.

This is the backbone of many practical BEADS implementations:

- global structure enforced via Db (smooth baseline),
- local sparsity enforced via Ws (spikes),
- efficient solving via ADMM.

8 Optimality Conditions and Proximal Perspective

8.1 First-order optimality for composite problems

For the composite problem $\min_x g(x) + h(x)$ with convex g, h, a point $x^\star$ is optimal iff

$$0 \in \nabla g(x^\star) + \partial h(x^\star),$$

i.e. there exists $s^\star \in \partial h(x^\star)$ such that

$$\nabla g(x^\star) + s^\star = 0.$$

Reminder (Inclusion $0 \in A(x)$). Writing $0 \in A(x)$ for a set-valued map A means there exists an element of $A(x)$ that equals 0.

This is the subgradient generalization of $\nabla F(x^\star) = 0$.

8.2 Proximal operator as resolvent of subgradient

For convex h, the subdifferential ∂h is a maximally monotone operator (in the finite-dimensional sense), and the proximal operator is its *resolvent*:

$$\mathrm{prox}_{\gamma h}(v) = (I + \gamma \partial h)^{-1}(v).$$

Reminder (Monotone operator). A set-valued operator A is monotone if $\langle u - v, x - y \rangle \geq 0$ whenever $u \in A(x)$ and $v \in A(y)$; it encodes a generalized notion of nonnegative slope.
Reminder (Resolvent). The resolvent of A is $(I + \gamma A)^{-1}$; for subgradients, this coincides with the proximal operator.

ISTA and FISTA can thus be viewed as forward-backward splitting methods for finding a zero of $\nabla g + \partial h$:

- forward step: explicit Euler step for ∇g,
- backward step: implicit step for ∂h via the prox.

ADMM can be viewed as a splitting scheme for the dual or as a Douglas–Rachford splitting on related monotone operators, but for most signal processing use it suffices to remember the practical iteration formulas.

9 Choosing Between ISTA, FISTA, and ADMM

9.1 ISTA vs FISTA

- **ISTA:**
  - Simple, stable, easy to implement.
  - Convergence: $O(1/k)$ in objective gap.
  - Good when high accuracy is not needed or when a warm start is available.
- **FISTA:**
  - Slightly more complex (keeps two sequences x^k, y^k and scalar t_k).
  - Faster convergence: $O(1/k^2)$.
  - Can overshoot; monotone variants and restarts are common.

9.2 ADMM vs proximal gradient

ADMM is preferable when:

- The problem has multiple terms that are easy to handle separately, e.g. multiple ℓ_1 penalties on linear transforms $W_1 x, W_2 x$.
- The smooth part does not have a cheap gradient but its quadratic part leads to a structured linear system that can be solved efficiently (e.g. via FFT or banded solvers).
- We want explicit access to primal and dual residuals (good stopping criteria in constrained problems).

ISTA/FISTA are preferable when:

- There is a single simple nonsmooth term, e.g. $\lambda \|x\|_1$ or $\lambda \|Dx\|_1$ with a cheap prox.
- The gradient of g is cheap (e.g. convolution) and we can apply A and $A^\top$ efficiently.

10 Implementation Notes in Signal Processing Contexts

10.1 Step sizes and Lipschitz constants

For $g(x) = \frac{1}{2} \|Ax - y\|_2^2$, we have

$$\nabla g(x) = A^\top (Ax - y),$$

with Lipschitz constant $L = \left\|A^\top A\right\|_2$.

Reminder (Spectral radius). For a symmetric matrix M, the spectral radius is the largest absolute eigenvalue; for $M = A^\top A$ this equals $\|A\|_2^2$.

In practice:

- If A is convolution with impulse response h, then $\|A\|_2$ is the maximum magnitude of the discrete-time Fourier transform of h, so L is the maximum squared magnitude; this can be estimated by FFT.
- A backtracking line search can be used instead of a fixed L if L is unknown or hard to estimate.

10.2 Stopping criteria

Common criteria:

- Relative change in iterate: $\frac{\left\|x^{k+1} - x^k\right\|_2}{\max(1, \left\|x^k\right\|_2)} < \varepsilon$.
- Relative change in objective: $\frac{|F(x^{k+1}) - F(x^k)|}{\max(1, |F(x^k)|)} < \varepsilon$.
- For ADMM: primal and dual residual norms below thresholds, e.g. $\left\|Ax^{k+1} + Bz^{k+1} - c\right\|_2 < \varepsilon_{\text{pri}}$ and $\left\|\rho A^\top B(z^{k+1} - z^k)\right\|_2 < \varepsilon_{\text{dual}}$.

10.3 Matrix-free implementations

In signal processing, A, D, W are often convolution or difference operators. Instead of building explicit matrices, we implement *matrix-free* operators:

- Given x, compute Ax via convolution or difference.
- Given u, compute $A^\top u$ via correlation or adjoint difference.

Reminder (Matrix-free operator). A matrix-free operator is represented by code that applies $x \mapsto Ax$ rather than storing A explicitly.

ISTA/FISTA then only need these operator applications, and ADMM's linear systems may be solved in the Fourier domain if H, D are circulant.

11 Summary

- Many BEADS-like and sparse signal processing problems can be written as composite convex optimizations $g(x)+h(x)$ with a smooth data-fidelity term and nonsmooth sparsity-promoting regularizers.
- The key mathematical ingredients are convexity, subgradients, and proximal operators; in particular, the prox of the ℓ_1 norm is soft-thresholding.
- ISTA performs proximal gradient descent and converges at rate $O(1/k)$.
- FISTA adds Nesterov momentum and achieves $O(1/k^2)$ convergence while retaining simple shrinkage steps.
- ADMM handles more complicated structures (multiple penalties, explicit constraints) by splitting variables and alternating between quadratic solves and prox steps.
- In the context of BEADS and related baseline+spike decompositions, these algorithms provide practical tools for enforcing sparsity, smoothness, and constraints in large-scale 1D and multi-dimensional signal problems.

As you build BEADS-like models, you can mix and match:

- analysis sparsity via Db and Ws,
- ISTA/FISTA when the structure is simple,
- ADMM when you want to separate baseline, spikes, and constraints cleanly.