Lecture 25: Plug-and-Play — Motivation; Plug-In Denoisers

Charles A. Bouman

Purdue University

Friday, October 23, 2026

Outline

1. Regularized inversion and its two limitations
2. Objectives, actions, and the proximal maps F and H
3. PnP-ADMM and the design of the plug-in denoiser
4. Are plug-in denoisers proximal maps? Moreau's Theorem

Reading: Sections 10.1 to 10.2.

Today's Big Ideas

1. Regularized inversion is not modular and needs cost functions.
2. The proximal map converts an objective into an action.
3. PnP-ADMM uses two actions; any action can replace a proximal map.
4. Plug-in prior: a denoiser for Gaussian noise, trained synthetically.
5. Most trained denoisers are not proximal maps; no cost is minimized.

Why Plug-and-Play

Machine learning and DNNs can represent very complex phenomena. How should we use them in computational imaging?

- One approach learns the map from data y to unknown x directly.
- Direct learning replaces the sensor model, which we want to keep.
- Direct learning is one-size-fits-all and lacks modular structure.

Goal of Plug-and-Play (PnP)

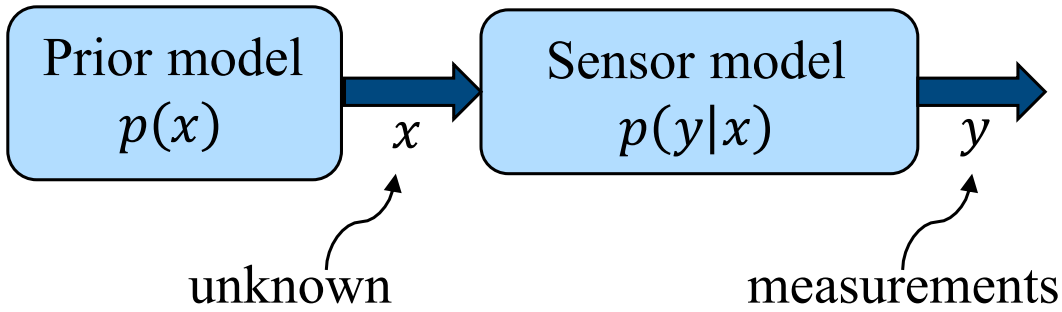
Fuse cost minimization models with **action-based** models, such as DNNs, that generate outputs from inputs.

Regularized inversion

The MAP estimate and the regularized ML estimate have the form

$$x^* = \arg\min_x \{f(x) + h(x)\} \quad . \tag{10.1}$$

- For the MAP estimate, $f(x) = -\log p(y|x) + \text{const}$ and $h(x) = -\log p(x) + \text{const}$.
- More generally, $f(x)$ fits the data; $h(x)$ penalizes irregular x.



Two limitations of regularized inversion

It is not modular

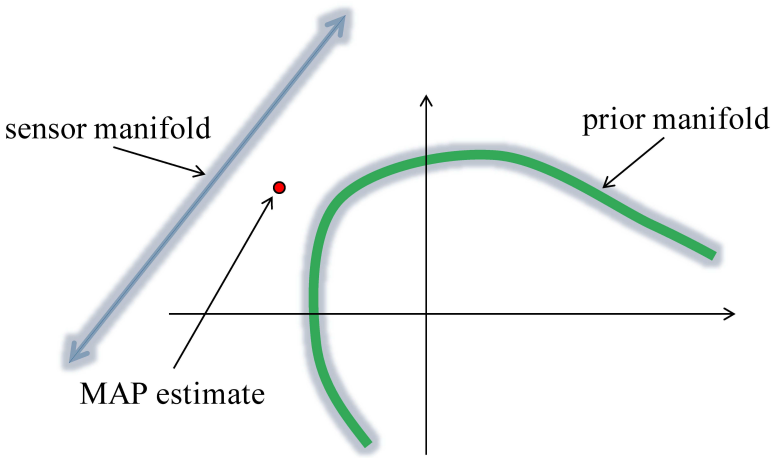
Both terms must be optimized together. In real applications the forward and prior models can each be tens of thousands of lines of code.

It is objective-based

The solution of (10.1) is based only on the **objective** of minimizing the cost functions $f(x)$ and $h(x)$. Many powerful techniques are instead **actions**: machine learning functions $F(x)$ that take an input and transform it to a desired output.

Variable splitting breaks a large optimization problem into smaller pieces. PnP builds on it.

The sensor manifold and the prior manifold



The regularized inverse seeks a point close to both the sensor manifold and the prior manifold.

- A linear sensor with AWGN gives a simple data term:

$$f(x) = \frac{1}{2}\|y - Ax\|^2 \ .$$

- The prior manifold is almost always complex and nonlinear, because real images have intricate structure.
- DNNs increasingly model the prior.

The proximal maps F and H

Define the proximal maps of f and h as

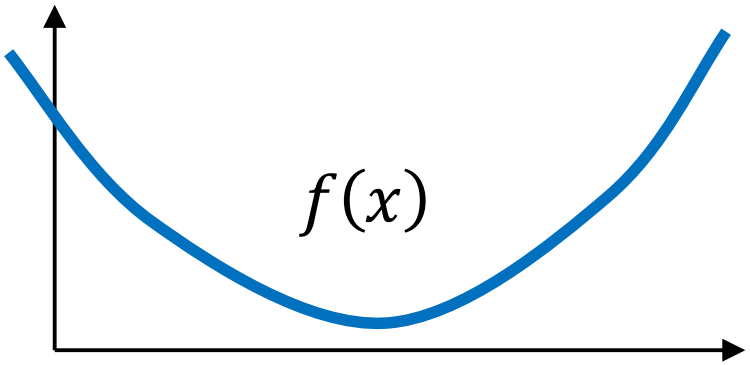
$$F(x) = \arg\min_v \left\{ f(v) + \frac{1}{2\sigma^2}\|v - x\|^2 \right\} , \tag{10.2}$$

$$H(x) = \arg\min_v \left\{ h(v) + \frac{1}{2\sigma^2}\|v - x\|^2 \right\} , \tag{10.3}$$

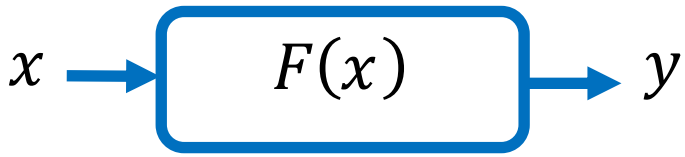
where $\sigma^2 > 0$ is a selectable parameter.

- σ^2 plays the role of $1/a$, where a is the ADMM penalty parameter.
- For convergence, f and h are proper closed convex on $\mathbb{R} \cup \{\infty\}$.

Objectives versus actions



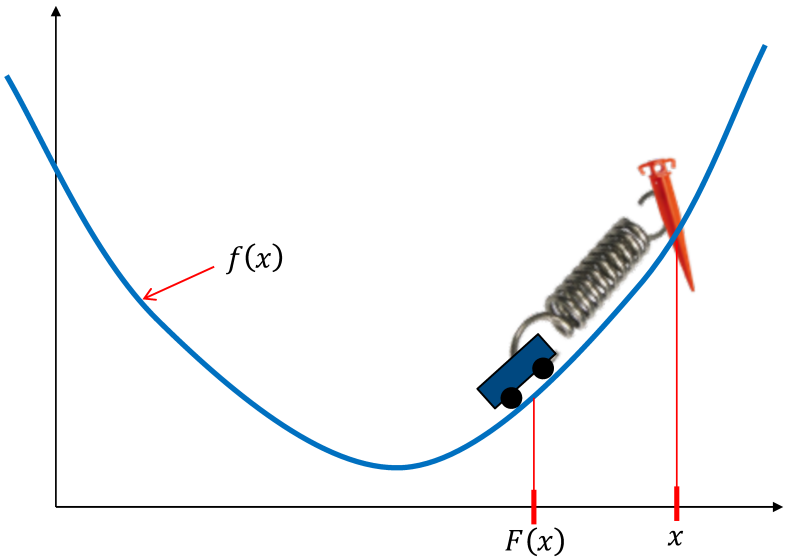
(a) Model of an objective



(b) Model of an action

- For the objective $f(x)$, the goal is to change x so that $f(x)$ decreases.
- Applying the action F to x creates the new value $F(x)$.

The proximal map converts an objective into an action



- Spring at x holds the cart at $F(x)$.
- Gravity pulls the cart down f.
- The spring keeps it off the bottom.
- So f makes F, which moves x down.

The central question

How do we fuse models that are a mixture of objectives and actions?

The PnP-ADMM algorithm

PnP-ADMM Algorithm

Initialize $u \leftarrow \mathbf{0}$ and $v \leftarrow \mathbf{0}$

Until Converged $\{$

$$\begin{aligned} x &\leftarrow F(v-u) \\ v &\leftarrow H(x+u) \\ u &\leftarrow u+(x-v) \end{aligned}$$

$\}$

Return(x)

- u is the scaled Lagrange multiplier.
- With (10.2) and (10.3), this is ADMM minimizing $f(x)+h(x)$.

Why it is called plug-and-play

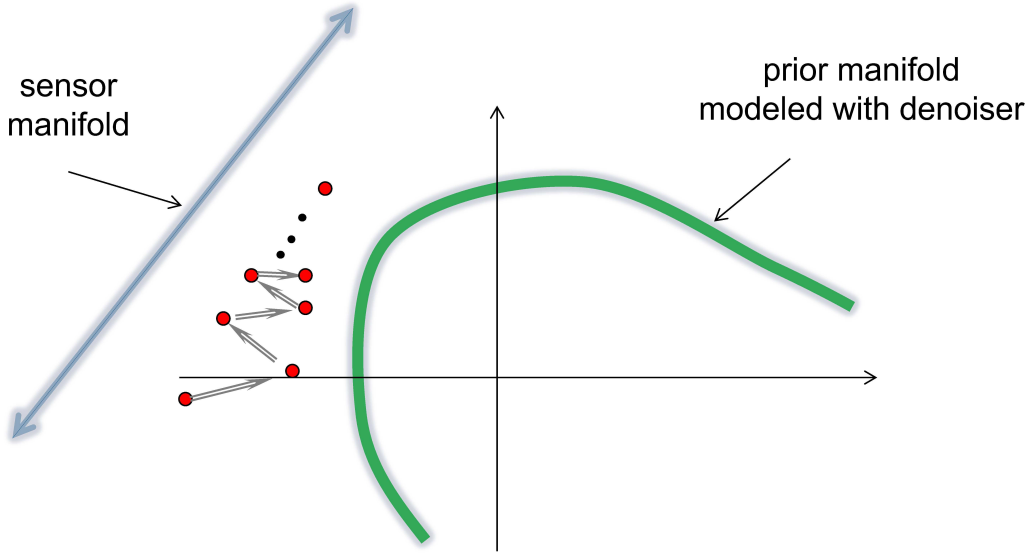
- PnP-ADMM uses f and h only through the actions F and H.
- So we can replace F or H with a more general action.
- We then **plug in** a different action for the proximal map.

A large conceptual step

When F and H are not proximal maps, PnP-ADMM is **no longer solving an optimization problem**.

The **PnP prior** replaces H with an action designed or trained to remove AWGN with variance σ^2.

What PnP-ADMM does



- The action F moves the solution closer to the sensor manifold.
- The action H moves the solution closer to the prior manifold.
- We hope the iterations converge to a balance of the two actions.

Agents and the proximal map of the prior

Define an **agent** to be an operation that replaces a proximal map in the ADMM algorithm. Each agent moves the solution toward its own goal.

The proximal map of the prior is

$$H(x) = \arg\min_v \left\{ \frac{1}{2\sigma^2}\|x - v\|^2 + h(v) \right\} . \tag{10.4}$$

Observation

$H(x)$ has the form of the **MAP estimate of an image corrupted by AWGN with variance** σ^2.

The virtual model

Define the **virtual model**

$$X = V + W \ ,$$

where V is an image typical of the reconstruction problem and W is synthetically generated AWGN with variance σ^2.

Assume the prior $V \sim \frac{1}{z} \exp\{-h(v)\}$. Then

$$\log p(x|v) = -\frac{1}{2\sigma^2} \|x - v\|^2 - \log z \ .$$

Result

The MAP estimate of V given X is **exactly the proximal map** $H(X)$ of (10.4).

Supervised training of the denoiser

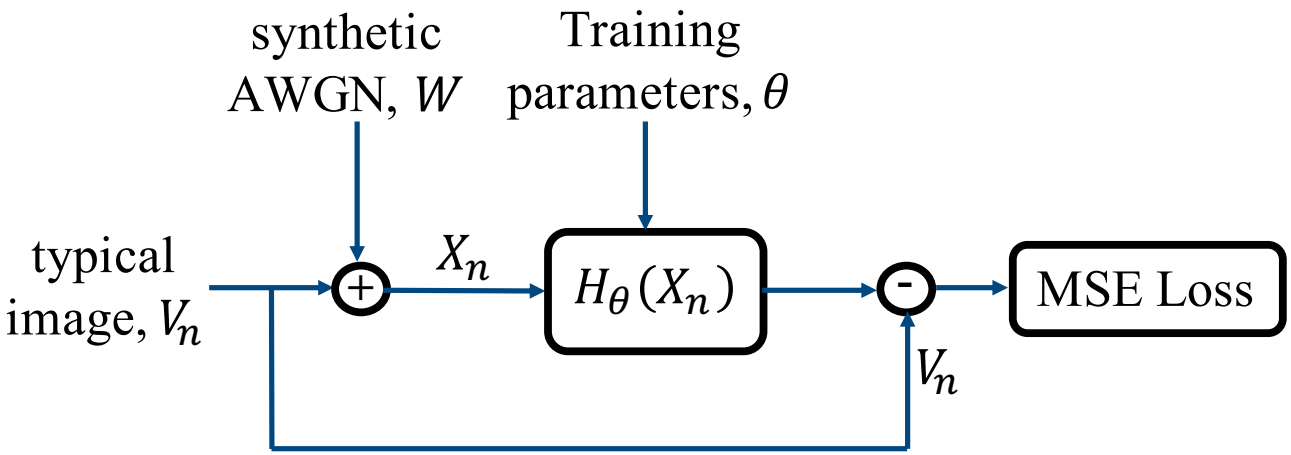
X and V are virtual images, but we can generate them synthetically.

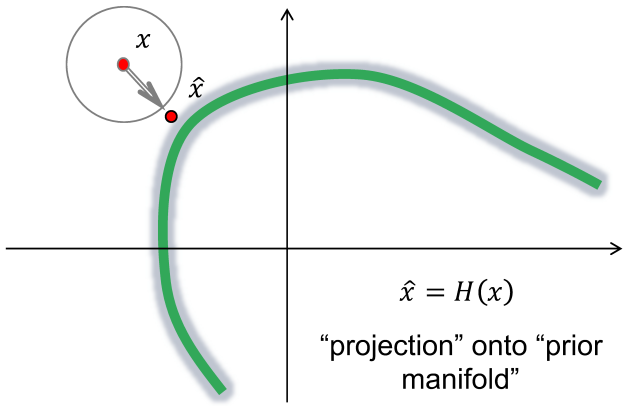
1. Take real images V_n from the application, for $n = 1, \ldots, N$.
2. Add white Gaussian noise with variance σ^2 to form X_n.
3. Train the denoiser H_θ on the pairs (V_n, X_n) with the MSE loss

$$L(\theta) = \sum_n \|V_n - H_\theta(X_n)\|^2 \ .$$

The MSE loss gives the conditional mean rather than the MAP estimate. In practice, it trains effective PnP denoisers.

Training of the PnP prior





- $H_\theta(x)$ is an image denoiser. Its output is an MMSE estimate of the noiseless image V_n.
- Like a proximal map, H_θ moves inputs toward the prior.

The questions PnP raises

Replacing H with a trained denoiser works very well. What does it mean?

1. Does PnP-ADMM converge? Anonymous reviewer: "plug-and-pray."
2. If it converges, what is the limit?
3. Is a trained denoiser H still a proximal map?

- With proximal maps, it is ADMM and the limit minimizes (10.1).
- For question 3, the answer is usually "no."

Nonexpansive and conservative functions

Let $H : \mathbb{R}^N \to \mathbb{R}^N$.

Definition: Nonexpansive function

H is **nonexpansive** if $\forall x, y \in \mathbb{R}^N$,

$$\|H(x) - H(y)\| \leq \|x - y\| \ .$$

Definition: Conservative function

H is **conservative** if there exists a continuously differentiable $\phi : \mathbb{R}^N \to \mathbb{R}$ such that $\forall x \in \mathbb{R}^N$,

$$H(x) = \nabla \phi(x) \ .$$

Moreau's Theorem

Theorem: Moreau's Theorem

Let $H : \mathbb{R}^N \to \mathbb{R}^N$ be a function. Then H is a proximal map of a proper closed convex function if and only if

- H is nonexpansive and
- H is conservative.

- Moreau proved this in the 1960s.
- A proximal map must be conservative: a gradient of a scalar.

The self-adjoint gradient condition

Theorem: Conservation Theorem for Vector Fields

Let $H : \mathbb{R}^N \to \mathbb{R}^N$ be continuously differentiable. Then H is conservative if and only if for all $x \in \mathbb{R}^N$,

$$\nabla H(x) = [\nabla H(x)]^t \ .$$

- Example: $H(x) = Ax$ has $\nabla H = A$; conservative only if $A^t = A$.
- In electromagnetics: a field is a gradient only if curl is zero.

Most denoisers are not proximal maps

- A self-adjoint gradient is a strong constraint; most functions lack it.
- So many useful actions $H(x)$ are not proximal maps.
- Non-proximal agents form a richer class that may reconstruct better.

The important point

A DNN trained for optimal image denoising is **not likely to be a proximal map**, but it can serve as an excellent prior model in PnP.

Summary

1. Regularized inversion computes $x^* = \arg\min_x \{f(x) + h(x)\}$. It is not modular, and it requires every model to be a cost function.
2. The proximal map $F(x) = \arg\min_v \{f(v) + \frac{1}{2\sigma^2}\|v - x\|^2\}$ converts the objective f into the action F.
3. PnP-ADMM iterates $x \leftarrow F(v - u)$, $v \leftarrow H(x + u)$, $u \leftarrow u + (x - v)$, and a general action can replace F or H.
4. The plug-in prior H_θ removes AWGN of variance σ^2, and it is trained with $L(\theta) = \sum_n \|V_n - H_\theta(X_n)\|^2$.
5. A differentiable H is a proximal map only if $\nabla H(x) = [\nabla H(x)]^t$, so most trained denoisers are not proximal maps, and PnP-ADMM with them minimizes no cost function.