Image Processing

Rao
Politecnico di Milano

Originally written in: **2025-02-17**

Last updated at: **2025-07-07**

Contents

1 Sparsity and Parsimony

The principle of sparsity or "parsimony" consists in representing some phenomenon with as few variable as possible. Stretch back to philosopher William Ockham in the 14th century, Wrinch and Jeffreys relate simplicity to parsimony:

The **existence of simple laws** is, then, apparently, to be regarded as **a quality of nature**; and accordingly we may infer that it is justifiable to **prefer a simple law to a more complex one that fits our observations slightly better**.

1.1 Sparsity in Statistics

Sparsity is used to **prevent overfitting and improve interpretability of learned models**. In model fitting, the number of parameters is typically used as a criterion to perform model selection. See Bayes Information Criterion (BIC), Akaike Information Criterion (AIC), …., Lasso.

1.2 Sparsity in Signal Processing

Signal Processing: similar concepts but different terminology. **Vectors corresponds to signals** and data modeling is crucial for performing various operations such as **restoration, compression, solving inverse problems**.

Signals are approximated by sparse linear combinations of **prototypes**(basis elements / atoms of a dictionary), resulting in simpler and compact model.

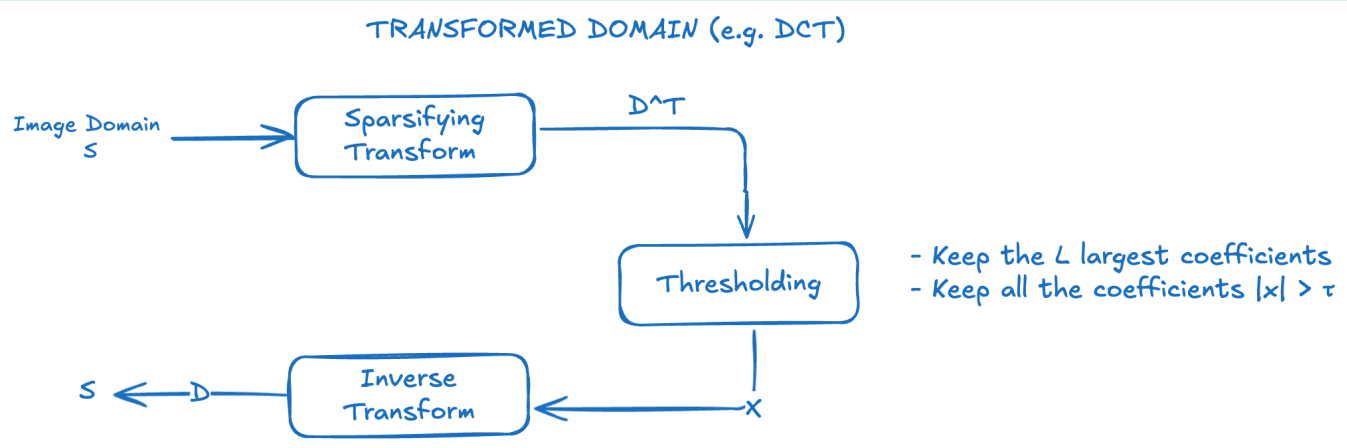

Figure 1.1: Enforce sparsity in signal processing

2 Signal Processing

2.1 Discrete Cosine Transform (DCT)

2.1.1 1D DCT

Generate the DCT basis according to the following formula, the k-th atom of the DCT basis in dimension M is defined as:

$$\mathrm{DCT}_{k(n)} = c_k \cos\left(k\pi \frac{2n+1}{2M}\right) \quad n, k = 0, 1, ..., M-1 \tag{2.1}$$

where $c_0 = \sqrt{\frac{1}{M}}$ and $c_k = \sqrt{\frac{2}{M}}$ for $k \neq 0$.

For each $k = 0, ..., M-1$, just sample each function

$$\mathrm{DCT}_{k(n)} = \cos\left(k\pi \frac{2n+1}{2M}\right) \tag{2.2}$$

at $n = 0, ..., M-1$, obtain a vector. Ignore the normalization coefficient. Divide each vector by its ℓ_2 norm.

Mathematically, suppose the image signal is $s \in \mathbb{R}^M$.

$$x = \mathrm{dct2}(s) = D^T s \tag{2.3}$$

where D^T represents the **DCT basis matrix**. x contains the **DCT coefficients**, which are a **sparse representation** of s.

The **inverse DCT transformation** reconstructs s from x:

$$s = \mathrm{idct2}(x) = Dx \tag{2.4}$$

2.1.2 2D DCT

2D Discrete Cosine Transform (DCT) can be used as a dictionary for representing image patches. A small patch of an image is extracted, represented as s, with dimension $p \times p$. This patch can be **flattened** into a vector of length $M = p^2$, meaning each patch is reshaped into a vector of length M. The **2D-DCT** is used to transform the patch s into DCT coefficients x.

Suppose the image signal is $S \in \mathbb{R}^{M \times N}$.The 2D DCT can be decomposed into two **1D DCT** operations:

1. **Column-wise DCT**: apply **1D DCT** to each column of the image patch: $\mathrm{Z} = D^T S$
2. **Row-wise DCT**: apply **1D DCT** to each column of the image patch: $X^T = D^T \mathrm{Z}^T \rightarrow X = D^T S D$

e.g. JPEG Compression — example 2.1.1

The image is divided into non-overlapping 8×8 blocks. Each block is treated separately during the compression process.

For each 8×8 block, the **DCT** is applied, transforming pixel values into frequency-domain coefficients. Each 8×8 block's coefficients are checked against a compression threshold τ, coefficients with absolute values below τ are **discarded**(set to zero). The larger the threshold τ, the more coefficients are discarded, leading to **higher compression**.

The compression ratio is defined as:

$$\text{Comp Ratio} = 1 - \frac{\#\text{Non-zero coefficients}}{\#\text{Pixels in the image}} \tag{2.5}$$

To measure how much the image quality is degraded after compression, **Peak Signal-to-Noise Ratio (PSNR)** is used:

$$\text{PSNR} = 10 \log_{10} \left(\frac{1}{\text{MSE}\left(Y, \hat{Y}\right)} \right) \tag{2.6}$$

3 Image Denoising

Image denoising provides a simple and clear problem formulation example. The ***observation model*** is:

$$z(x) = y(x) + \eta(x) \tag{3.1}$$

where:

- $z(x)$ is noisy observation at pixel coordinate x
- $y(x)$ is ideal (noise-free) image
- $\eta(x)$is the noise component
- x is pixel coordinate

This formulation assumes *additive white Gaussian noise*(AWGN).

We assume that the noise is:

- **Additive Gaussian noise**: $\eta(x) \sim N(0, \sigma^2)$.
- **Independent and identically distributed (i.i.d.)**: Noise realizations at different pixels are independent.

While real-world noise may exhibit correlations or non-Gaussian characteristics, the AWGN model remains prevalent for algorithm design. Practical systems often transform raw sensor data to approximate this model. The denoising objective is formalized as finding an estimator $\hat{y}$ minimizing the the mean squared error (MSE):

$$\text{MSE}(\hat{\boldsymbol{y}}) = \mathbb{E}[\|\hat{\boldsymbol{y}} - \boldsymbol{y}\|_2^2] \tag{3.2}$$

The observation model provides a **prior on noise** but we also need a **prior on images**.

Def Noise Prior **definition 3.0.1**

A **noise prior** refers to the assumed statistical properties or characteristics of the noise itself that is corrupting the image. Knowing how the noise behaves helps in modeling and subsequently removing it. For AWGN, the true image y is likely to be a in a circular neighborhood around the observation $z(x)$.

Def Image Prior **definition 3.0.2**

An **image prior** is a statistical model that captures the expected structure of natural images. It is used to regularize the denoising process, guiding the estimator towards plausible solutions.

3.1 Local Constancy Prior

Assumption: Images are locally constant within small patches. For a constant signal corrupted by Gaussian noise:

$$\hat{y} = \frac{1}{M} \sum_{i=1}^{M} z(x_i) \tag{3.3}$$

Properties of the this estimator:

- **Unbiased:** $\mathbb{E}[\hat{y}] = y$ (true signal)
- **Reduced Variance:** $\text{Var}[\hat{y}] = \frac{\sigma^2}{M}$

Limitations: Local averaging introduces *bias* at edges:

$$\text{Bias} = |\mathbb{E}[\hat{y}] - y| \gg 0 \text{ at discontinuities}$$

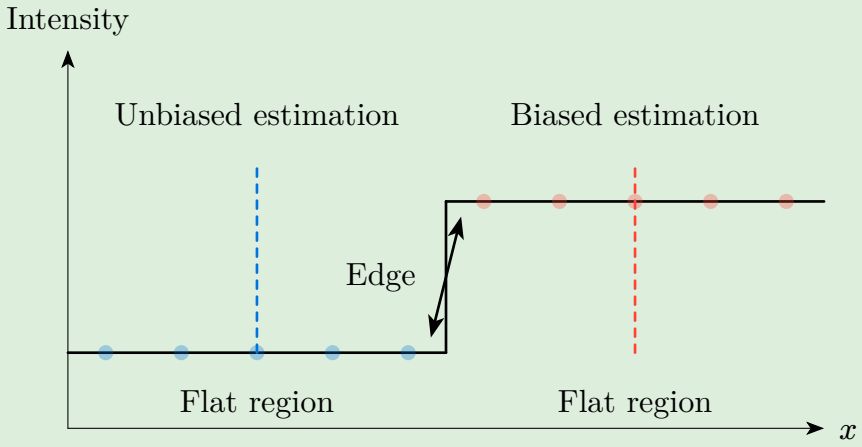

Figure 3.1: Bias-variance tradeoff in local averaging

3.2 Sparsity-Based Image Prior

3.2.1 Motivation for Sparsity

Natural images have **sparse representations** in certain transform domains (e.g., DCT), as evidenced by the success of JPEG compression. **Key Insight:** If images can be sparsely represented for compression, this same property can be leveraged for denoising.

3.2.2 DCT-Based Denoising Pipeline

Step 1 : Analysis

$$X = D^T S \tag{3.4}$$

where:

- S is the vectorized image patch
- D is the DCT basis matrix
- X is the DCT coefficients vector

Step 2: Enforce Sparsity (Thresholding)

$$\hat{X}_i = \begin{cases} X_i & \text{if } |X_i| \geq \gamma \\ 0 & \text{if } |X_i| < \gamma \end{cases} \tag{3.5}$$

Important: Apply thresholding only to the coefficients $i \geq 1$ (preserve the DC component).

Step 3: Synthesis

$$\hat{S} = D\hat{X} \tag{3.6}$$

3.2.3 Threshold Selection

Thm Universal Thresholding **theorem 3.2.1**

For AWGN with variance σ^2, the optimal threshold for denoising is given by:

$$\gamma = \sigma\sqrt{2\log(n)} \tag{3.7}$$

where:

- σ is the noise standard deviation
- n is the dimension of coefficients vector

For 8×8 patches: $\gamma \approx 3\sigma$

3.3 Noise Standard Deviation Estimation

To use the universal thresholding, we need to estimate σ from the noisy image itself.

Thm Robust Estimation of Noise Standard Deviation **theorem 3.3.1**

Given a noisy image Z, the noise standard deviation can be estimated using the following robust method:

$$\hat{\sigma} = \frac{\text{MAD}}{0.6745 \times \sqrt{2}} \tag{3.8}$$

where MAD = Median Absolute Deviation, defined as:

$$\text{MAD}(D) = \text{median}(|D - \text{median}(D)|) \tag{3.9}$$

where D denotes the horizontal differences of the image.

3.4 Sliding DCT Algorithm

3.4.1 Non-Overlapping Tiles (No Aggregation)

The simplest approach processes the image in non-overlapping 8×8 blocks. Let Z be the noisy image of size $M \times N$, partitioned into non-overlapping blocks $B_{i,j}$ of size 8×8. For each block:

$$X_{i,j} = \text{DCT}_2(B_{i,j}) \tag{3.10}$$

Apply hard thresholding with universal threshold $\gamma = 3\sigma$:

$$\hat{X}_{i,j}(u,v) = \begin{cases} X_{i,j}(u,v) & \text{if } |X_{i,j}(u,v)| \geq \gamma \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

Reconstruct each block:

$$\hat{S}_{i,j} = \text{IDCT}_2\left(\hat{X}_{i,j}\right) \tag{3.12}$$

The final denoised image is the union of all processed blocks:

$$\hat{Y} = \bigcup_{i,j} \hat{B}_{i,j} \tag{3.13}$$

Properties:

- Complexity: $O(N)$ operations
- Blocking artifacts due to independent processing
- Fast but lower quality

3.4.2 Sliding Window with Uniform Weights

For overlapping patches, each pixel receives multiple estimates that must be aggregated.

For each patch position (i, j) with step size $\text{STEP} = 1$, extract $p \times p$ patch $S_{i,j}$ from the noisy image and do the same DCT processing as before, getting the reconstructed patch $\hat{S}_{i,j}$.

The denoised image $\hat{I}$ is obtained by weighted aggregation:

$$\hat{I}(m,n) = \frac{\sum_{i,j \in \Omega_{m,n}} w \cdot \hat{S}_{i,j}(m-i, n-j)}{\sum_{i,j \in \Omega_{m,n}} w + \varepsilon} \tag{3.14}$$

where $w = 1.0$ is the uniform weight, $\varepsilon = 10^{-8}$ prevents division by zero, and $\Omega_{m,n}$ denotes the set of patch positions (i, j) that contain pixel (m, n).

For an image of size $M \times N$, each pixel (m, n) can be covered by at most $p \times p$ overlapping patches when using unit step size, but the actual number depends on the pixel's position relative to image boundaries.

3.4.3 Sparsity-Adaptive Weight Aggregation

For overlapping patches, each pixel receives multiple estimates that must be aggregated with weights adapted to the sparsity of the thresholded DCT coefficients.

For each patch position (i, j) with step size $\text{STEP} = 1$, extract $p \times p$ patch $S_{i,j}$ from the noisy image, getting the reconstructed patch $\hat{S}_{i,j}$.

The sparsity-adaptive weight for each patch is computed based on the number of **non-zero coefficients** after thresholding:

$$w_{i,j} = \text{nnz}\left(\hat{X}_{i,j}\right) \tag{3.15}$$

where $\text{nnz}\left(\hat{X}_{i,j}\right)$ counts the number of non-zero elements in the thresholded DCT coefficient matrix.

The denoised image $\hat{I}$ is obtained by sparsity-weighted aggregation:

$$\hat{I}(m,n) = \frac{\sum_{i,j \in \Omega_{m,n}} w_{i,j} \cdot \hat{S}_{i,j}(m-i, n-j)}{\sum_{i,j \in \Omega_{m,n}} w_{i,j} + \varepsilon} \tag{3.16}$$

where $\varepsilon = 10^{-8}$ prevents division by zero, and $\Omega_{m,n}$ denotes the set of patch positions (i, j) that contain pixel (m, n).

3.5 Wiener Filter

The **Wiener filter** is a powerful tool for image denoising, particularly when the noise characteristics are known. It operates in the frequency domain, leveraging the DCT coefficients to perform adaptive filtering based on local statistics.

3.5.1 Empirical Wiener Filter

Let $\hat{\boldsymbol{y}}^{\mathrm{HT}}$ be the **hard threshold estimate**, with DCT coefficients:

$$\hat{\boldsymbol{x}}^{\mathrm{HT}} = D^T \hat{\boldsymbol{y}}^{\mathrm{HT}} \tag{3.17}$$

The empirical Wiener filter attenuates the DCT coefficients as:

$$\hat{x}_i^{\mathrm{Wie}} = \frac{\left(\hat{x}_i^{\mathrm{HT}}\right)^2}{\left(\hat{x}_i^{\mathrm{HT}}\right)^2 + \sigma^2} x_i \tag{3.18}$$

The empirical Wiener estimate is thus:

$$\hat{\boldsymbol{y}}^{\mathrm{HT}} = D\hat{\boldsymbol{x}}^{\mathrm{Wie}} \tag{3.19}$$

3.5.2 Transform Domain Patch Processing

Given an image $\boldsymbol{Y}$ of size $M \times M$, we extract overlapping patches $\boldsymbol{P}_{i,j}$ of size $p \times p$ centered at pixel (i, j):

$$\boldsymbol{P}_{i,j} = \boldsymbol{Y}\left[i - \left\lfloor \frac{p}{2} \right\rfloor : i + \left\lfloor \frac{p}{2} \right\rfloor, j - \left\lfloor \frac{p}{2} \right\rfloor : j + \left\lfloor \frac{p}{2} \right\rfloor\right] \tag{3.20}$$

For each patch $\boldsymbol{P}_{i,j}$, we apply the following procedure:

1. **Vectorization**: Convert patch to vector $\boldsymbol{p}_{i,j} \in \mathbb{R}^{p^2}$
2. **Transformation**: Apply orthogonal transform $\tilde{\boldsymbol{p}}_{i,j} = \boldsymbol{T}\boldsymbol{p}_{i,j}$
3. **Preliminary Estimation**: Obtain initial estimate $\hat{\tilde{\boldsymbol{p}}}_{i,j}^{(0)}$ using a simple denoising method
4. **Wiener Filtering**: Apply empirical Wiener filter coefficient-wise
5. **Inverse Transform**: Reconstruct patch $\hat{\boldsymbol{p}}_{i,j} = \boldsymbol{T}^{-1}\hat{\tilde{\boldsymbol{p}}}_{i,j}$

4 Limitations of Sparsity-Based Denoising

4.1 The Sparsity Problem

While orthonormal bases provide computational convenience and guarantee unique representations, they suffer from a fundamental limitation: *no single orthonormal basis can provide sparse representations for all signals of interest.*

e.g. DCT Basis Limitation **example 4.1.1**

Consider a signal $\boldsymbol{s}_0 \in \mathbb{R}^n$ that admits a sparse representation with respect to the Discrete Cosine Transform (DCT) basis $\boldsymbol{D}_{\text{DCT}}$:

$$\boldsymbol{s_0} = \boldsymbol{D}_{\text{DCT}}\boldsymbol{x_0} \tag{4.1}$$

where $\boldsymbol{x_0}$ is sparse (most entries are zero).

Now consider the modified signal:

$$\boldsymbol{s} = \boldsymbol{s_0} + \lambda \boldsymbol{e_j} \tag{4.2}$$

where $\boldsymbol{e_j}$ is the j-th canonical basis vector and $\lambda \in \mathbb{R}$ is a scaling factor.

The DCT representation of $\boldsymbol{s}$ becomes:

$$\boldsymbol{x} = \boldsymbol{D}_{\text{DCT}}^T\boldsymbol{s} = \boldsymbol{D}_{\text{DCT}}^T\boldsymbol{s_0} + \lambda \boldsymbol{D}_{\text{DCT}}^T\boldsymbol{e_j} = \boldsymbol{x_0} + \lambda \boldsymbol{D}_{\text{DCT}}^T\boldsymbol{e_j} \tag{4.3}$$

Since $\boldsymbol{D}_{\text{DCT}}^T\boldsymbol{e_j}$ is typically dense (all entries are non-zero), the addition of a single spike destroys the sparsity of the representation.

4.2 Experimental Demonstration

The experimental verification of this limitation involves:

1. Generate a sparse signal $\boldsymbol{s_0}$ with respect to DCT basis
2. Add a single spike: $\boldsymbol{s} = \boldsymbol{s_0} + \lambda \boldsymbol{e_j}$
3. Compute DCT coefficients of both signals
4. Observe the loss of sparsity in the modified signal

The results consistently show that the addition of a single spike causes all DCT coefficients to become significant, effectively destroying the sparse structure that denoising algorithms rely upon.

4.3 Overcomplete Dictionaries: The Solution

4.3.1 Motivation for Redundancy

The solution to the sparsity limitation lies in abandoning the constraint of orthonormality and embracing redundancy. Instead of using a single $n \times n$ orthonormal basis, we construct an $n \times m$ dictionary matrix $\boldsymbol{D}$ where $m > n$.

Def Overcomplete Dictionary **definition 4.3.1**

An *overcomplete dictionary* is a matrix $\boldsymbol{D} \in \mathbb{R}^{n \times m}$ with $m > n$ such that:

$$\text{span}\{\boldsymbol{d}_1, \boldsymbol{d}_2, ..., \boldsymbol{d}_m\} = \mathbb{R}^n \tag{4.4}$$

where $\boldsymbol{d}_i$ are the columns of $\boldsymbol{D}$.

4.3.2 Construction of Overcomplete Dictionaries

For the DCT-spike example, we construct the overcomplete dictionary by concatenating the DCT basis with the **canonical basis**:

$$\boldsymbol{D} = [\boldsymbol{D}_{\text{DCT}} \;\; \boldsymbol{I}] \in \mathbb{R}^{n \times 2n} \tag{4.5}$$

This construction ensures that:

- Signals sparse in DCT domain remain sparse
- Signals sparse in canonical domain remain sparse
- Mixed signals (DCT-sparse + spikes) admit sparse representations

e.g. Sparse Representation with Overcomplete Dictionary **example 4.3.1**

Consider the signal $\boldsymbol{s} = \boldsymbol{s}_0 + \lambda \boldsymbol{e}_j$ where $\boldsymbol{s}_0 = \boldsymbol{D}_{\text{DCT}} \boldsymbol{x}_0$ with sparse $\boldsymbol{x}_0$.

The representation with respect to the overcomplete dictionary is:

$$\boldsymbol{s} = \boldsymbol{D} \begin{bmatrix} \boldsymbol{x}_0 \\ \lambda \boldsymbol{e}_j \end{bmatrix} \tag{4.6}$$

The coefficient vector $\begin{bmatrix} \boldsymbol{x}_0 \\ \lambda \boldsymbol{e}_j \end{bmatrix} \in \mathbb{R}^{2n}$ is sparse, containing only the non-zero entries of $\boldsymbol{x}_0$ plus the single entry λ at position j in the second block.

4.3.3 Theoretical Properties of Overcomplete Systems

Thm Rouché-Capelli Theorem **theorem 4.3.1**

Consider the linear system $\boldsymbol{D}\boldsymbol{x} = \boldsymbol{s}$ where $\boldsymbol{D} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{s} \in \mathbb{R}^n$. The system admits a solution if and only if:

$$\text{rank}(\boldsymbol{D}) = \text{rank}([\boldsymbol{D} \;\; \boldsymbol{s}]) \tag{4.7}$$

When $m > n$ and $\text{rank}(\boldsymbol{D}) = n$, the system has infinitely many solutions forming an affine subspace of dimension $m - n$.

Thm Solution Space Dimension **theorem 4.3.2**

If $\boldsymbol{D} \in \mathbb{R}^{n \times m}$ with $m > n$ and $\text{rank}(\boldsymbol{D}) = n$, then for any $\boldsymbol{s} \in \mathbb{R}^n$, the solution set of $\boldsymbol{D}\boldsymbol{x} = \boldsymbol{s}$ forms an affine subspace of dimension $m - n$.

4.4 Regularization and Sparse Recovery

4.4.1 The Ill-Posed Nature of Overcomplete Systems

The abundance of solutions in overcomplete systems necessitates additional criteria for solution selection. This is where regularization theory becomes essential.

Def Regularization **definition 4.4.1**

Given an **ill-posed** problem $\boldsymbol{D}\boldsymbol{x} = \boldsymbol{s}$ with multiple solutions, *regularization* involves solving:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} J(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{D}\boldsymbol{x} = \boldsymbol{s} \tag{4.8}$$

where $J : \mathbb{R}^m \to \mathbb{R}_+$ is a regularization functional encoding our prior knowledge about the desired solution.

4.4.2 ℓ_2 Regularization: Ridge Regression

The most mathematically tractable regularization is the ℓ_2 norm:

$$J(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{x}\|_2^2 = \frac{1}{2} \sum_{i=1}^{m} x_i^2 \tag{4.9}$$

This leads to the constrained optimization problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{x}\|_2^2 \quad \text{subject to} \quad \boldsymbol{D}\boldsymbol{x} = \boldsymbol{s} \tag{4.10}$$

Alternatively, we can formulate the unconstrained version:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{s}\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{x}\|_2^2 \tag{4.11}$$

4.4.3 Analytical Solution via Matrix Calculus

The unconstrained ℓ_2 regularization problem admits a closed-form solution. To derive this, we use matrix calculus.

Thm Ridge Regression Solution **theorem 4.4.1**

The solution to the ridge regression problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{s}\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{x}\|_2^2 \tag{4.12}$$

is given by:

$$\hat{\boldsymbol{x}} = \left(\boldsymbol{D}^T\boldsymbol{D} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{D}^T\boldsymbol{s} \tag{4.13}$$

where $\lambda > 0$ ensures the matrix $\left(\boldsymbol{D}^T\boldsymbol{D} + \lambda \boldsymbol{I}\right)$ is invertible.

Proof: Define the objective function:

$$f(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{s}\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{x}\|_2^2 \tag{4.14}$$

Expanding the squared norms:

$$\begin{aligned} f(\boldsymbol{x}) &= \frac{1}{2}(\boldsymbol{D}\boldsymbol{x} - \boldsymbol{s})^T(\boldsymbol{D}\boldsymbol{x} - \boldsymbol{s}) + \frac{\lambda}{2}\boldsymbol{x}^T\boldsymbol{x} \\ &= \frac{1}{2}[x]^T\boldsymbol{D}^T\boldsymbol{D}[x] - [s]^T\boldsymbol{D}[x] + \frac{1}{2}[s]^T[s] + \frac{\lambda}{2}[x]^T[x] \end{aligned} \tag{4.15}$$

Taking the gradient with respect to $\boldsymbol{x}$:

$$\nabla f(\boldsymbol{x}) = \boldsymbol{D}^T\boldsymbol{D}\boldsymbol{x} - \boldsymbol{D}^T\boldsymbol{s} + \lambda \boldsymbol{x} \tag{4.16}$$

Setting $\nabla f(\boldsymbol{x}) = \boldsymbol{0}$:

$$(\boldsymbol{D}^T\boldsymbol{D} + \lambda \boldsymbol{I})\boldsymbol{x} = \boldsymbol{D}^T\boldsymbol{s} \tag{4.17}$$

Since $\lambda > 0$, the matrix $\left(\boldsymbol{D}^T\boldsymbol{D} + \lambda \boldsymbol{I}\right)$ is positive definite and therefore invertible, yielding the stated solution.

4.4.4 Limitations of ℓ_2 Regularization

While ℓ_2 regularization provides a computationally efficient solution, it does not promote sparsity. The solution $\hat{\boldsymbol{x}}$ typically has all non-zero entries, which contradicts our goal of sparse representation.

Sparsity vs. ℓ_2 Regularization **attention 4.4.1**

The ℓ_2 norm penalizes large coefficients but does not drive small coefficients to zero. For sparse recovery, we need regularization functionals that promote sparsity, such as the ℓ_1 norm or ℓ_0 pseudo-norm.

4.5 Towards Sparsity: ℓ_0 and ℓ_1 Regularization

4.5.1 The ℓ_0 "Norm" and True Sparsity

The most natural regularization for sparse recovery is the ℓ_0 "norm" (technically a pseudo-norm):

$$\|\boldsymbol{x}\|_0 = |\{i : x_i \neq 0\}| \tag{4.18}$$

This counts the number of non-zero entries in $\boldsymbol{x}$. The corresponding optimization problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \boldsymbol{D}\boldsymbol{x} = \boldsymbol{s} \tag{4.19}$$

directly seeks the sparsest representation.

4.5.2 Computational Challenges

The ℓ_0 minimization problem is NP-hard in general, making it computationally intractable for large-scale problems. This has led to the development of convex relaxations and approximation algorithms.

5 Appendix: Fundamental Concepts in Linear Algebra

5.1 Vector Spaces and Linear Combinations

Def Span of Vectors **definition 5.1.1**

Given a set of vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n\} \subset \mathbb{R}^m$, the *span* of these vectors is defined as:

$$\text{span}\{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n\} = \left\{ \sum_{i=1}^{n} \lambda_i \boldsymbol{v}_i : \lambda_i \in \mathbb{R} \right\} \tag{5.1}$$

The span represents the set of all possible linear combinations of the given vectors, forming a vector subspace of $\mathbb{R}^m$. This concept is fundamental to understanding how different sets of vectors can generate different subspaces.

5.2 Linear Independence and Basis

Def Linear Independence **definition 5.2.1**

A set of vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n\}$ is *linearly independent* if and only if:

$$\sum_{i=1}^{n} \lambda_i \boldsymbol{v}_i = \boldsymbol{0} \quad \Rightarrow \quad \lambda_i = 0 \ \text{ for all } \ i = 1, 2, ..., n \tag{5.2}$$

This definition captures the fundamental property that no vector in the set can be expressed as a linear combination of the others. The importance of linear independence becomes clear when we consider the uniqueness of representations.

Thm Uniqueness of Representation **theorem 5.2.1**

Let $\{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n\} \subset \mathbb{R}^m$ be a linearly independent set of vectors. If $\boldsymbol{s} \in \text{span}\{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n\}$, then there exists a unique representation:

$$\boldsymbol{s} = \sum_{i=1}^{n} x_i \boldsymbol{e}_i \tag{5.3}$$

where the coefficients $x_i \in \mathbb{R}$ are uniquely determined.

Proof: Suppose $\boldsymbol{s}$ admits two different representations:

$$\begin{aligned} \boldsymbol{s} &= \sum_{i=1}^{n} x_i \boldsymbol{e}_i \\ \boldsymbol{s} &= \sum_{i=1}^{n} y_i \boldsymbol{e}_i \end{aligned} \tag{5.4}$$

Subtracting these equations:

$$\boldsymbol{0} = \sum_{i=1}^{n} (x_i - y_i) \boldsymbol{e}_i \tag{5.5}$$

By linear independence, $(x_i - y_i) = 0$ for all i, implying $x_i = y_i$ for all i. Therefore, the representation is unique.

5.3 Orthogonal Vectors

Def Orthonormal Basis **definition 5.3.1**

A set of vectors $\{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n\} \subset \mathbb{R}^n$ forms an *orthonormal basis* if:

1. $\langle \boldsymbol{e}_i, \boldsymbol{e}_j \rangle = \delta_{ij}$ (orthonormality condition)
2. $\text{span}\{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n\} = \mathbb{R}^n$ (spanning condition)

where δ_{ij} is the Kronecker delta.

The power of orthonormal bases lies in their computational convenience. For any signal $\boldsymbol{s} \in \mathbb{R}^n$ and orthonormal basis $\boldsymbol{D} = [\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_n]$, the coefficient computation is straightforward:

$$\boldsymbol{x} = \boldsymbol{D}^T \boldsymbol{s} \tag{5.6}$$

where $\boldsymbol{x} = (x_1, x_2, ..., x_n)^T$ and $x_i = \langle \boldsymbol{e}_i, \boldsymbol{s} \rangle$.