Mathematical Foundations of Data Sciences

Gabriel Peyré
CNRS & DMA
École Normale Supérieure
gabriel.peyre@ens.fr
https://mathematical-tours.github.io
www.numerical-tours.com

November 15, 2020

Presentation

This book draft presents an overview of important mathematical and numerical foundations for modern data sciences. It covers in particulars the basics of signal and image processing (Fourier, Wavelets, and their applications to denoising and compression), imaging sciences (inverse problems, sparsity, compressed sensing) and machine learning (linear regression, logistic classification, deep learning). The focus is on the mathematically-sounded exposition of the methodological tools (in particular linear operators, non-linear approximation, convex optimization, optimal transport) and how they can be mapped to efficient computational algorithms. These course notes are also intended to be the theoretical companion for the Numerical Tours[1] web site, which presents Matlab/Python/Julia/R detailed implementations of all the concepts covered here.

[1] www.numerical-tours.com

Contents

Chapter 1

Shannon Theory

Shannon theory of information, published in 1948/1949, is made of three parts:

1. Sampling: it studies condition under which sampling a continuous function to obtain a discrete vector is invertible. The discrete real values representing the signal are then typically quantized to a finite precision to obtain a set of symbols in a finite alphabet.

2. Source coding: it studies optimal ways to represent (code) such a set of symbols as a binary sequence. It leverages the statistical distributions to obtain the most possible compact code.

3. Channel coding (not studied here): it studies how to add some redundancy to the coded sequence in order to gain robustness to errors or attacks during transmission (flip of certain bits with some probability). It is often named "error correcting codes theory".

The main reference for this chapter is [17].

1.1 Analog vs. Discrete Signals

To develop numerical tools and analyze their performances, the mathematical modelling is usually done over a continuous setting (so-called "analog signals"). Such continuous setting also aims at representing the signal in the physical world, which are inputs to sensors hardwares such as microphone, digital cameras or medical imaging devices. An analog signal is a 1-D function $f_0 \in \mathrm{L}^2([0,1])$ where $[0,1]$ denotes the domain of acquisition, which might for instance be time. An analog image is a 2D function $f_0 \in \mathrm{L}^2([0,1]^2)$ where the unit square $[0,1]^2$ is the image domain.

Although these notes are focussed on the processing of sounds and natural images, most of the methods extend to multi-dimensional datasets, which are higher dimensional mappings

$$f_0 : [0,1]^d \to [0,1]^s$$

where d is the dimensionality of the input space ($d = 1$ for sound and $d = 2$ for images) whereas s is the dimensionality of the feature space. For instance, gray scale images corresponds to $(d = 2, s = 1)$, videos to $(d = 3, s = 1)$, color images to $(d = 2, s = 3)$ where one has three channels (R, G, B). One can even consider multi-spectral images where $(d = 2, s \gg 3)$ that is made of a large number of channels for different light wavelengths. Figures 1.1 and 1.2 show examples of such data.

1.1.1 Acquisition and Sampling

Signal acquisition is a low dimensional projection of the continuous signal performed by some hardware device. This is for instance the case for a microphone that acquires 1D samples or a digital camera that

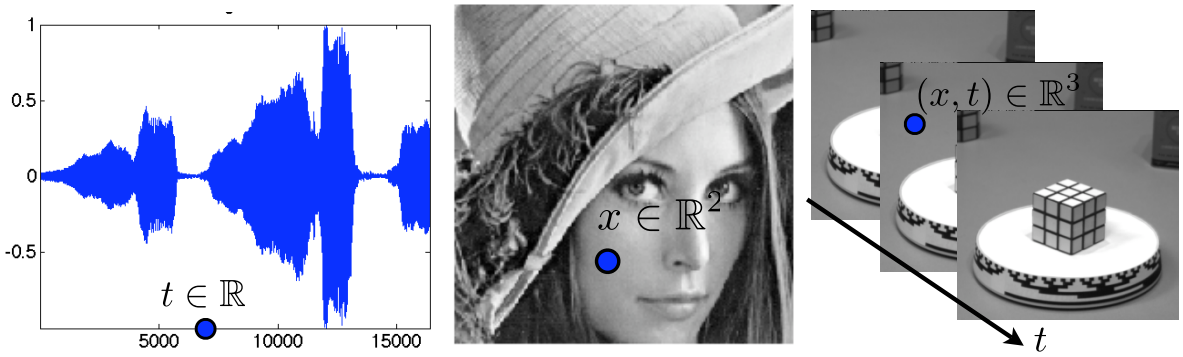


Figure 1.1: Examples of sounds ($d = 1$), image ($d = 2$) and videos ($d = 3$).

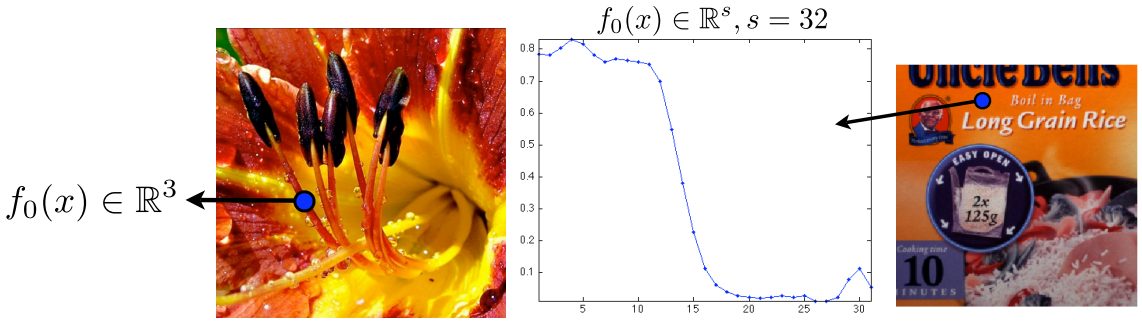


Figure 1.2: Example of color image $s = 3$ and multispectral image ($s = 32$).

acquires 2D pixel samples. The sampling operation thus corresponds to mapping from the set of continuous functions to a discrete finite dimensional vector with N entries.

$$f_0 \in \mathrm{L}^2([0,1]^d) \mapsto f \in \mathbb{C}^N$$

Figure 1.3 shows examples of discretized signals.

1.1.2 Linear Translation Invariant Sampler

A translation invariant sampler performs the acquisition as an inner product between the continuous signal and a constant impulse response h translated at the sample location

$$f_n = \int_{-S/2}^{S/2} f_0(x) h(n/N - x) \mathrm{d}x = f_0 \star h(n/N). \tag{1.1}$$

The precise shape of $h(x)$ depends on the sampling device, and is usually a smooth low pass function that is maximal around $x = 0$. The size S of the sampler determines the precision of the sampling device, and is usually of the order of $1/N$ to avoid blurring (if S is too large) or aliasing (if S is too small).

Section **??** details how to reverse the sampling operation in the case where the function is smooth.

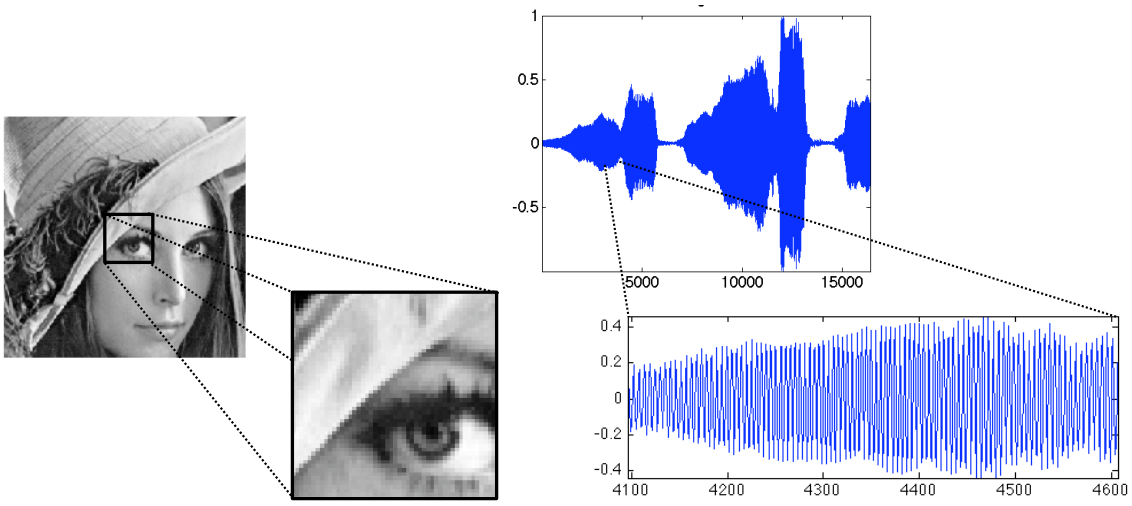


Figure 1.3: Image and sound discretization.

1.2 Shannon Sampling Theorem

Reminders about Fourier transform. For $f \in L^1(\mathbb{R})$, its Fourier transform is defined as

$$\forall\, \omega \in \mathbb{R}, \quad \hat{f}(\omega) \stackrel{\text{def.}}{=} \int_{\mathbb{R}} f(x) e^{-\mathrm{i}x\omega} \mathrm{d}x. \tag{1.2}$$

One has $\|\hat{f}\|^2 = (2\pi)^{-1}\|f\|^2$, so that $f \mapsto \hat{f}$ can be extended by continuity to $L^2(\mathbb{R})$, which corresponds to computing $\hat{f}$ as a limit when $T \to +\infty$ of $\int_{-T}^{T} f(x) e^{-\mathrm{i}x\omega}\mathrm{d}x$. When $\hat{f} \in L^1(\mathbb{R})$, one can invert the Fourier transform so that

$$f(x) = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\omega) e^{\mathrm{i}x\omega} \mathrm{d}\omega, \tag{1.3}$$

which shows in particular that f is continuous with vanishing limits at $\pm\infty$.

The Fourier transform $\mathcal{F} : f \mapsto \hat{f}$ exchanges regularity and decay. For instance, if $f \in C^p(\mathbb{R})$ with an integrable Fourier transform, then $\mathcal{F}(f^{(p)})(\omega) = (\mathrm{i}\omega)^p \hat{f}(\omega)$ so that $|\hat{f}(\omega)| = O(1/|\omega|^p)$. Conversely,

$$\int_{\mathbb{R}} (1+|\omega|)^p |\hat{f}(\omega)| \mathrm{d}\omega < +\infty \quad \Longrightarrow \quad f \in C^p(\mathbb{R}). \tag{1.4}$$

For instance, if $\hat{f}(\omega) = O(1/|\omega|^{p+2})$, one obtains that $f \in C^p(\mathbb{R})$.

Reminders about Fourier series. We denote $\mathbb{T} = \mathbb{R}/2\pi\mathbb{Z}$ the torus. A function $f \in L^2(\mathbb{T})$ is 2π-periodic, and can be viewed as a function $f \in L^2([0, 2\pi])$ (beware that this means that the boundary points are glued together), and its Fourier coefficients are

$$\forall\, n \in \mathbb{Z}, \quad \hat{f}_n \stackrel{\text{def.}}{=} \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-\mathrm{i}xn} \mathrm{d}x.$$

This formula is equivalent to the computation of an inner-product $\hat{f}_n = \langle f,\, e_n \rangle$ for the inner-product $\langle f,\, g \rangle \stackrel{\text{def.}}{=} \frac{1}{2\pi} \int_{\mathbb{T}} f(x)\bar{g}(x)\mathrm{d}x$. For this inner product, $(e_n)_n$ is orthonormal and is actually an Hilbert basis, meaning that one reconstructs with the following converging series

$$f = \sum_{n \in \mathbb{Z}} \langle f,\, e_n \rangle e_n \tag{1.5}$$

which means $\|f - \sum_{n=-N}^{N} \langle f, e_n \rangle e_n\|_{L^2(\mathbb{T})} \to 0$ for $N \to +\infty$. The pointwise convergence of (1.5) at some $x \in \mathbb{T}$ is ensured if for instance f is differentiable. The series is normally convergent (and hence uniform) if for instance f if of class C^2 on $\mathbb{T}$ sin ce in this case, $\hat{f}_n = O(1/n^2)$. If there is a step discontinuities, then there is Gibbs oscillations preventing uniform convergence, but the series still converges to the half of the left and right limit.

Poisson formula. The poisson formula connects the Fourier transform and the Fourier series to sampling and periodization operators. For some function $h(\omega)$ defined on $\mathbb{R}$ (typically the goal is to apply this to $h = \hat{f}$), its periodization reads

$$h_P(\omega) \stackrel{\text{def.}}{=} \sum_n h(\omega - 2\pi n). \tag{1.6}$$

This formula makes sense if $h \in L^1(\mathbb{R})$, and in this case $\|h_P\|_{L^1(\mathbb{T})} \leqslant \|h\|_{L^1(\mathbb{R})}$ (and there is equality for positive functions). The Poisson formula, stated in Proposition 1 below, corresponds to proving that the following diagram

$$\begin{array}{ccccc} & f(x) & \stackrel{\mathcal{F}}{\longrightarrow} & \hat{f}(\omega) & \\ \text{sampling} & \downarrow & & \downarrow & \text{periodization} \\ & (f(n))_n & \stackrel{\text{Fourier serie}}{\longrightarrow} & \sum_n f(n) e^{-\mathrm{i}\omega n} & \end{array}$$

is actually commutative.

Proposition 1 (Poisson formula)**.** *Assume that $\hat{f}$ has compact support and that $|f(x)| \leqslant C(1+|x|)^{-3}$ for some C. Then one has*

$$\forall\, \omega \in \mathbb{R}, \quad \sum_n f(n) e^{-\mathrm{i}\omega n} = \hat{f}_P(\omega). \tag{1.7}$$

Proof. Since $\hat{f}$ is compactly supported, $\hat{f}_P$ is well defined (it involves only a finite sum) and since f has fast decay, using (1.4), $(\hat{f})_P$ is C^1. It is thus the sum of its Fourier series

$$(\hat{f})_P(\omega) = \sum_k c_k e^{\mathrm{i}k\omega}, \tag{1.8}$$

where

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} (\hat{f})_P(\omega) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = \frac{1}{2\pi} \int_0^{2\pi} \sum_n \hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega} \mathrm{d}\omega.$$

One has

$$\int_0^{2\pi} \sum_n |\hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega}| \mathrm{d}\omega = \int_{\mathbb{R}} |\hat{f}|$$

which is bounded because $\hat{f} \in L^1(\mathbb{R})$ (it has a compact support and is C^1), so one can exchange the sum and integral

$$c_k = \sum_n \frac{1}{2\pi} \int_0^{2\pi} \hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\omega) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = f(-k)$$

where we used the inverse Fourier transform formula (1.3), which is legit because $\hat{f} \in L^1(\mathbb{R})$. □

Shannon theorem. Shannon sampling theorem state a sufficient condition ensuring that the sampling operator $f \mapsto (f(ns))_n$ is invertible for some sampling step size $s > 0$. It require that $\mathrm{supp}(\hat{f}) \subset [-\pi/s, \pi/s]$, which, thanks to formula (1.3), implies that $\hat{f}$ is C^∞ (in fact it is even analytic). This theorem was first proved by Wittaker in 1915. It was re-proved and put in perspective in electrical engineering by Nyquist in 1928. It became famous after the paper of Shannon in 1949, which put forward its importance in numerical communications. Figure 1.4 give some insight on how the proof works (left) and more importantly, on what happens when the compact support hypothesis fails (in which case aliasing occurs, see also Figure 1.7).

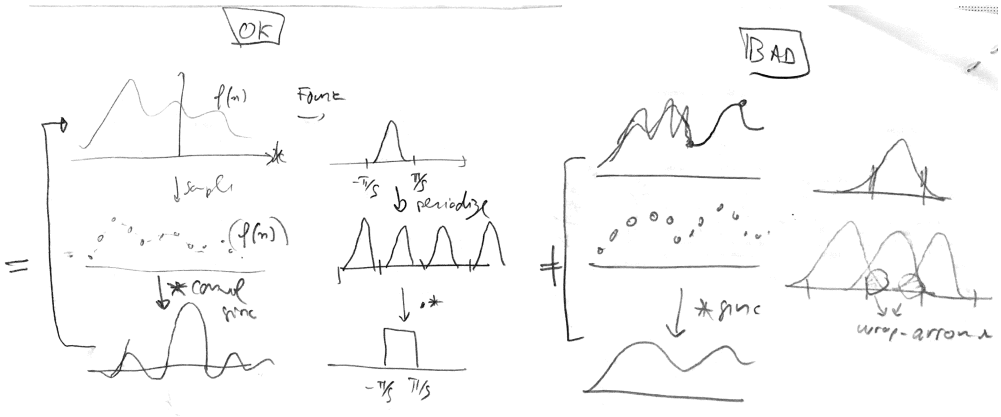


Figure 1.4: Schematic view for the proof of Theorem 1.

Theorem 1. *If* $|f(x)| \leqslant C(1+|x|)^{-3}$ *for some* C *and* $\mathrm{supp}(\hat{f}) \subset [-\pi/s, \pi/s]$*, then one has*

$$\forall\, x \in \mathbb{R}, \quad f(x) = \sum_n f(ns)\,\mathrm{sinc}(x/s - n) \quad \textit{where} \quad \mathrm{sinc}(u) = \frac{\sin(\pi u)}{\pi u} \tag{1.9}$$

with uniform convergence.

Proof. The change of variable $g \overset{\text{def.}}{=} f(s\cdot)$ results in $\hat{g} = 1/s\hat{f}(\cdot/s)$, indeed, denoting $z = sx$

$$\hat{g}(\omega) = \int f(sx)e^{-\mathrm{i}\omega x}\mathrm{d}x = \frac{1}{s}\int f(z)e^{-\mathrm{i}(\omega/s)z}\mathrm{d}z = \hat{f}(\omega/s)/s,$$

so that we can restrict our attention to $s = 1$. The compact support hypothesis implies $\hat{f}(\omega) = 1_{[-\pi,\pi]}(\omega)\hat{f}_P(\omega)$. Combining the inversion formula (1.3) with Poisson formula (1.8)

$$f(x) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{f}_P(\omega)e^{\mathrm{i}\omega x}\mathrm{d}\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi}\sum_n f(n)e^{\mathrm{i}\omega(x-n)}\mathrm{d}\omega.$$

Since f has fast decay, $\int_{-\pi}^{\pi}\sum_n |f(n)e^{\mathrm{i}\omega(x-n)}|\mathrm{d}\omega = \sum_n |f(n)| < +\infty$, so that one can exchange summation and integration and obtain

$$f(x) = \sum_n f(n)\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{\mathrm{i}\omega(x-n)}\mathrm{d}\omega = \sum_n f(n)\,\mathrm{sinc}(x-n).$$

□

One issue with this reconstruction formula is that it uses a slowly decaying and very oscillating sinc kernels. In practice, one rarely uses such a kernel for interpolation, and one prefers smoother and more localized kernel. If $\mathrm{supp}(\hat{f}) \subset [-\pi/s', \pi/s']$ with $s' > s$ (i.e. have a more compact spectrum), one can re-do the proof of the theorem, and one gains some degree of freedom to design the reconstruction kernel, which now can be chosen smoother in Fourier and hence have exponential decay in time.

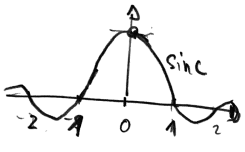


Figure 1.5: sinc kernel

Spline interpolation are defined by considering $\varphi_0 = 1_{[-1/2,1/2]}$ and $\varphi_k = \varphi_{k-1} \star \varphi_0$ which is a piecewise polynomial of degree k and has bounded derivative of order k (and is of class C^{k-1}) with compact support on $[-(k+1)/2, (k+1)/2]$.

The reconstruction formula reads $f \approx \tilde{f} \stackrel{\text{def.}}{=} \sum_n a_n \varphi(\cdot - n)$ where $(a_n)_n$ is computed from the $(f(n))_n$ by solving a linear system (associated to the interpolation property $\tilde{f}(n) = f(n)$). It is only in the cases $k \in \{0, 1\}$ (piecewise constant and affine interpolations) that one has $a_n = f(n)$. In practice, one typically use the cubic spline interpolation, which corresponds to $k = 3$.

`Associated code: test_sampling.m`

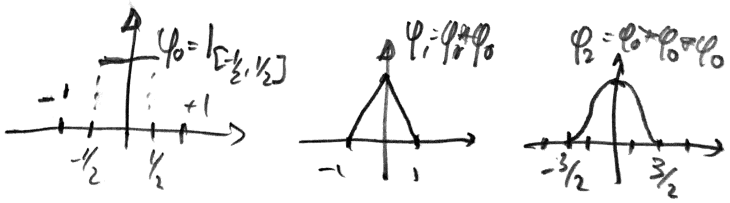


Figure 1.6: Cardinal splines as bases functions for interpolation.

This theorem also explains what happens if $\hat{f}$ is not supported in $[-\pi/s, \pi/s]$. This leads to aliasing, and high frequency outside this interval leads to low frequency artifacts often referred to as "aliasing". If the input signal is not bandlimitted, it is thus very important to pre-filter it (smooth it) before sampling to avoid this phenomena (of course this kills the high frequencies, which are lost), see Figure 1.7.

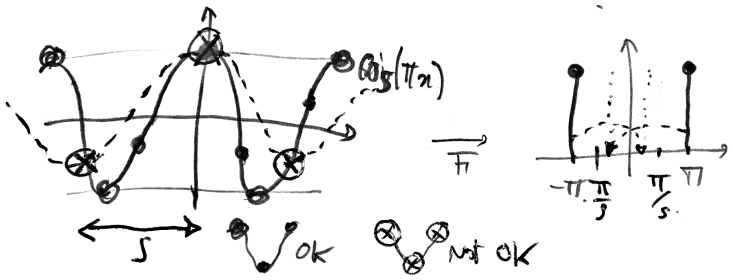


Figure 1.7: Aliasing in the simple case of a sine wave (beware however that this function does not have compact support).

Quantization. Once the signal have been sampled to obtain a discrete vector, in order to store it and transmit it, it is necessary to quantize the value to some finite precision. Section 5.1 presents transform coding, which is an efficient family of compression schemes which operate the quantization over some transformed domain (which correspond to applying a linear transform, usually orthogonal, to the sampled values). This is useful to enhance the performance of the source coding scheme. It is however common to operate directly the quantization over the sampled value.

Considering for instance a step size $s = 1/N$, one samples $(u_n \stackrel{\text{def.}}{=} f(n/N))_{n=1}^N \in \mathbb{R}^N$ to obtain a finite dimensional data vector of length N. Note that dealing with finite data corresponds to restricting the function f to some compact domain (here $[0, 1]$) and is contradictory with Shannon sampling theorem, since a function f cannot have a compact support in both space and frequency (so perfect reconstruction never holds when using finite storage).

Choosing a quantization step T, quantization $v_n = Q_T(u_n) \in \mathbb{Z}$ rounds to the nearest multiple of T, i.e.

$$v = Q_T(u) \quad \Leftrightarrow \quad v - \frac{1}{2} \leqslant u/T < v + \frac{1}{2},$$

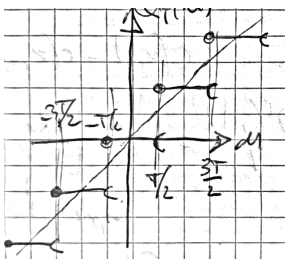


Figure 1.8:

see Fig. **??**. De-quantization is needed to restore a signal, and the best reconstruction (in average or in worse case) is defined by setting $D_T(v) \stackrel{\text{def.}}{=} Tv$. Quantizing and then de-quantizing introduce an error bounded by $T/2$, since $|D_T(Q_T(u)) - u| \leqslant T/2$. Up to machine precision, quantization is the only source of error (often called "lossy compression") in Shannon's standard pipeline.

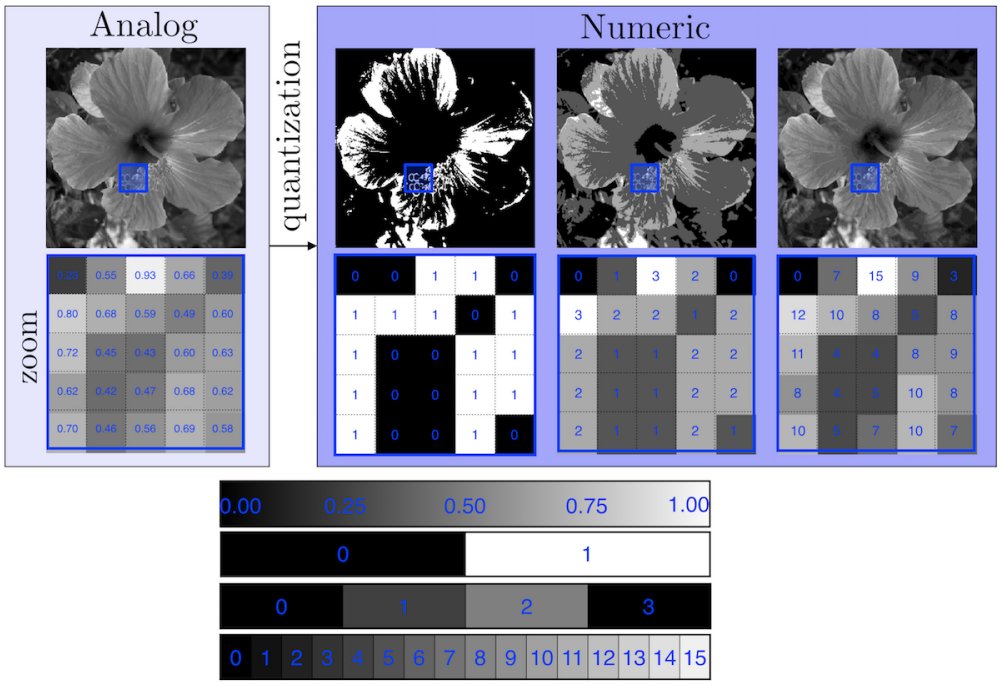


Figure 1.9: Quantizing an image using a decaying $T = 1/K$ where $K \in \{2, 3, 4, 16\}$ is the number of graylevels and the original image is normalized so that $0 \leqslant f_0 < 1$.

1.3 Shannon Source Coding Theorem

Uniform coding. We consider an alphabet $(s_1, \ldots, s_K)$ of K symbols. For instance, if one samples and quantize a bounded signal $0 \leqslant f_0 < 1$ using a step size $1/K$, then one can consider $s_k = k$ to be integer symbols. For text, these symbols include the letter plus extra punctuation symbols and blank. It is of course possible to code a sequence of such symbols using a uniform code (e.g. using the base 2 expansion) with $\lceil \log_2(K) \rceil$ bit per symbols. For instance if $K = 4$ and the symbols are $\{0, 1, 2, 3\}$, then the code words are $(c_0 = 00, c_1 = 01, c_2 = 10, c_3 = 11)$.

This uniform coding strategy is however extremely inefficient if the symbols are not uniformly distributed (i.e. if some symbols are more frequent than other, which is likely to be the case). We aim at designing better codes.

Prefix coding. A code $c_k = c(s_k)$ associate to each symbol s_k a code word $c_k \in \{0, 1\}^{\mathbb{N}}$ with a varying length $|c_k| \in \mathbb{N}^*$. A prefix code $c_k = c(s_k)$ is such that no word c_k is the beginning of another word c'_k. This is equivalent to be able to embed the $(c_k)_k$ as leaves of a binary tree T, with the code being output of a traversal from root to leaves (with a convention that going to a left (resp. right) child output a 0 (resp. a 1). We denote $c = \text{Leaves}(T)$ such prefix property.

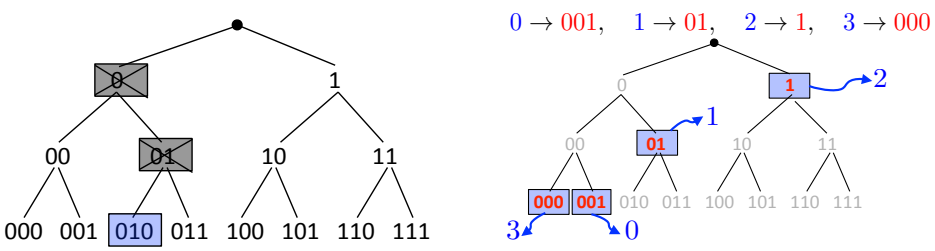


Figure 1.10: Left: complete tree of all codes of length 3; right: example of prefix code.

This tree-based representation is useful to decode a binary stream by simply performing tree traversal. One follows the tree, from top to bottom, and outputs a symbol each time a leaf is reached (and then re-start at the top).

Probabilistic modeling. We aim at designing the most possible compact code c_k. We assume at our disposal some probability distribution over this alphabet, which is just an histogram $p = (p_1, \dots, p_K) \in \mathbb{R}_+^K$ in the simplex, i.e. $\sum_k p_k = 1$. In practice, this probability is usually the empirical probability of appearance of the symbols x_k in the data to be coded.

The entropy of such an histogram is

$$H(p) \stackrel{\text{def.}}{=} -\sum_k p_k \log_2(p_k)$$

with the convention $0 \log_2(0) = 0$.

Denoting $h(u) = -u \log_2(u)$, $h'(u) \propto -\log(u) - 1$, $h''(u) \propto -1/u < 0$ so that H is strictly concave. The definition of the entropy extends to continuous density $f(x)$ for x on some measure space with reference measure $\mathrm{d}x$ (e.g. Lebesgue on $\mathbb{R}^d$) by setting $H(f) \stackrel{\text{def.}}{=} -\int f(x) \log(f(x)) \mathrm{d}x$.

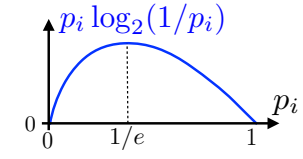


Figure 1.11:

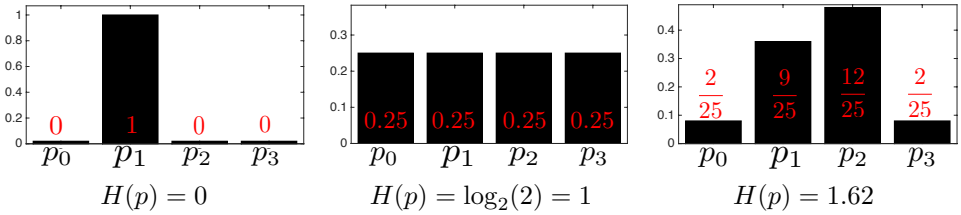


Figure 1.12: Three examples of probability distributions with corresponding entropies.

Lemma 1. *One has*

$$0 = H(\delta_i) \leqslant H(p) \leqslant H(1/K) = \log_2(K)$$

where δ_i is the Dirac histogram distribution at i.

Proof. First one notes that $-u \log_2(u) \geqslant 0$ for $u \in [0, 1]$ (see figure above), so that $H \geqslant 0$. Then we show the following inequality, for any histogram q

$$H(p) \leqslant -\sum_i p_i \log(q_i) \quad \Leftrightarrow \quad \sum_i p_i \log(q_i/p_i) \geqslant 0.$$

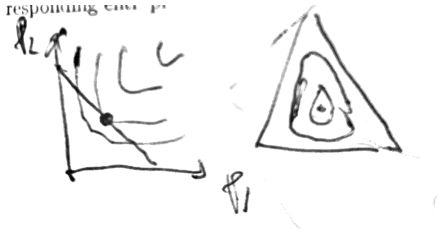


Figure 1.13: Linked extrema for the entropy.

This follows from $\log(u) \leqslant u-1$, since

$$\sum_i p_i \log(q_i/p_i) \leqslant \sum_i p_i(q_i/p_i - 1) = \sum_i q_i - \sum_i p_i = 0.$$

Applying this inequality to $q_i = 1/K$ gives

$$H(p) \leqslant -\sum_i p_i \log(1/K) = \log(1/K).$$

□

Shannon theorem. Assuming that $(p_k)_k$ is the empirical probability of appearance of the symbols x_k in the data to be coded, the average symbol length associated to some code c is

$$L(c) \stackrel{\text{def.}}{=} \sum_k p_k |c_k|.$$

The goal is to design the best possible c so that $L(c)$ is as small as possible. Shannon theorem of entropic coding, proved below, give both lower and upper bound for this question.

Theorem 2. *(i) If $c = \text{Leaves}(T)$ for some tree T, then*

$$L(c) \geqslant H(p).$$

(ii) Conversely, there exists a code c with $c = \text{Leaves}(T)$ such that

$$L(c) \leqslant H(p) + 1.$$

Before proving this theorem, we prove Kraft inequality, which describes the set of prefix codes using an inequality.

Lemma 2 (Kraft inequality)**.** *(i) For a code c, if there exists a tree T such that $c = \text{Leaves}(T)$ then*

$$\sum_k 2^{-|c_k|} \leqslant 1. \tag{1.10}$$

(ii) Conversely, if $(\ell_k)_k$ are such that

$$\sum_k 2^{-\ell_k} \leqslant 1 \tag{1.11}$$

then there exists a code $c = \text{Leaves}(T)$ such that $|c_k| = \ell_k$.

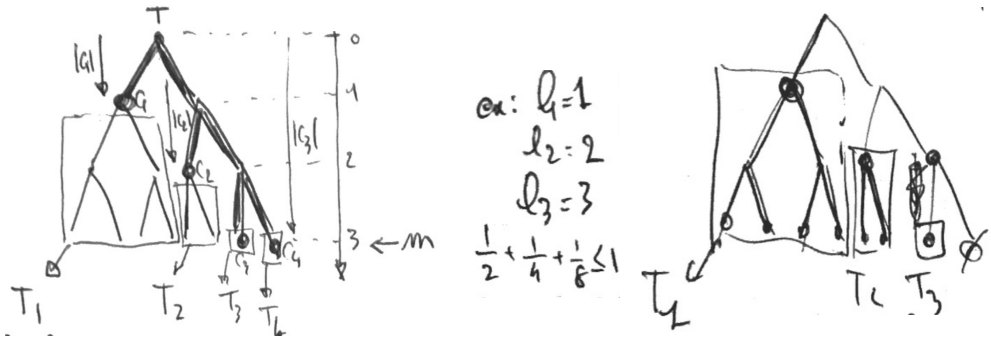


Figure 1.14: Left: full binary tree obtained by completing the tree associated to the code ($c_1 = 0, c_2 = 10, c_3 = 110, c_4 = 111$). Right: packing sub-trees associated to code length to form the left part of the full tree.

Proof. $\Rightarrow$ We suppose $c = \text{Leaves}(T)$. We denote $m = \max_k |c_k|$ and consider the full binary tree. Bellow each c_k, one has a sub-tree of height $m - |c_k|$, see Figure 2.1, left. This sub-tree has $2^{m-|c_k|}$ leaves. Since all these sub-trees do not overlap, the total number of leaf do not exceed the total number of leaves 2^m of the full binary tree, hence

$$\sum_k 2^{m-|c_k|} \leqslant 2^m,$$

hence (1.10).

$\Leftarrow$ Conversely, we assume (1.10) holds. Without loss of generality, we assume that $|c_1| \geqslant \ldots \geqslant |c_K|$. We start by putting a sub-tree of height $2^{m-|c_1|}$. Since the second tree is smaller, one can put it immediately aside, and continue this way, see Figure 2.1, right. Since $\sum_k 2^{m-|c_k|} \leqslant 2^m$, this ensure that we can stack side-by-side all these sub-tree, and this defines a proper sub-tree of the full binary tree. □

Shannon theorem. First, we consider the following optimization problem

$$\min_{\ell=(\ell_k)_k} \left\{ f(\ell) \stackrel{\text{def.}}{=} \sum_k \ell_k p_k \;;\; g(\ell) \stackrel{\text{def.}}{=} \sum_k 2^{-\ell_k} \leqslant 1 \right\}. \tag{1.12}$$

We fist show that at an optimal $\ell^\star$, the constraint is saturated, i.g. $g(\ell^\star) = 1$. Indeed, if $g(\ell^\star) = 2^{-u} < 1$, with $u > 0$, we define $\ell'_k \stackrel{\text{def.}}{=} \ell^\star_k - u$, which satisfies $g(\ell') = 1$ and also $f(\ell') = \sum_k (\ell_k - u)p_k < f(\ell^\star)$, which is a contradiction. So we can restrict in (1.12) the constraint to $g(\ell) = 1$ and apply the linked extrema theorem, which shows that necessarily, there exists $\lambda \in \mathbb{R}$ with $\nabla f(\ell^\star) = \nabla g(\ell^\star)$, i.e. $(p_k)_k = -\lambda \ln(2)(2^{-\ell^\star_k})_k$. Since

$$1 = \sum_k p_k = -\lambda \ln(2) \sum_k 2^{-\ell_k} = -\lambda \ln(2)$$

we deduce that $\ell^\star_k = -\log(p_k)$.

(i) If $c = \text{Leave}(T)$, the by Kraft inequality (1.10), necessarily $\ell_k = |c_k|$ satisfy the constraints of (1.12), and thus $H(p) = f(\ell^\star) \leqslant f(\ell) = L(\ell)$.

(ii) We define $\ell_k \stackrel{\text{def.}}{=} \lceil -\log_2(p_k) \rceil \in \mathbb{N}^*$. Then $\sum_k 2^{-\ell_k} \leqslant \sum_k 2^{\log_2(p_k)} = 1$, so that these lengths satisfy (1.11). Thanks to Proposition 2 (ii), there thus exists a prefix code c with $|c_k| = \lceil -\log_2(p_k) \rceil$. Furthermore

$$L(c) = \sum_k p_k \lceil -\log_2(p_k) \rceil \leqslant \sum_k p_k (-\log_2(p_k) + 1) = H(p) + 1.$$

□

Note that this proof is constructing, i.e. it gives an algorithm that construct an almost optimal c, and this code is often called the Shannon-Fano code. It is usually a good code, although it is not necessarily the optimal code with the smallest $L(c)$. Such an optimal code can easily be computed in almost linear time (only sorting of the probability is needed, so it is $K(\log(K))$) by Huffman's dynamic programming algorithm (invented in 1952). The proof of correctness of this algorithm is however a bit tedious. Figure 1.15 shows an example of application of this algorithm.

`Associated code: coding/test_text.m`

In practice, such an entropic coding, although optimal, is not very efficient when one of the symbol has a large probability p_k. This is because then $2^{-p_k} \ll 1$ but one cannot allocate a fractional number of bit. This is why $L(c)$ can be as large as $H(p)+1$. A simple workaround is to artificially increase the size of the alphabet from K to K^r by grouping together sets of r consecutive symbols, and thus reducing the gap to $H(p)+1/r$. Constructing the code and coding however becomes very slow for large r. The standard way to achieve this without explicitly doing the grouping is by using arithmetic coding, invented in 1976, which using interval arithmetic to allocate fractional number of bits and leveraging the fact that one usually code large sequence, thus approaching to arbitrary precision Shannon bound $H(p)$ as the length of the data increases.

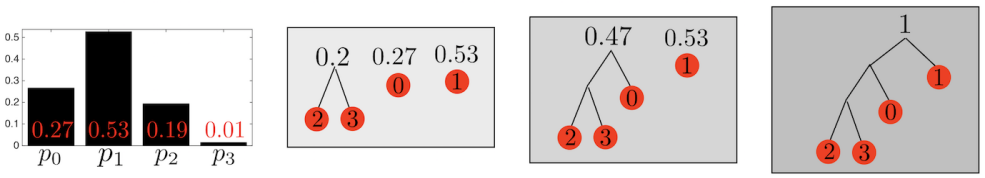


Figure 1.15: Huffman coding algorithm in action.

Note that while we give some statistical and probabilistic interpretation of entropy (measure of uncertainty) and of Shannon theorem, this theory is fully deterministic and give a bound for the actual length $NL(c)$ of coding some sequence of length N if the probability p are the empirical probability of the sequence.

If one choose a different probability q and use it to code the sequence, one necessarily obtain a worse average coding length, and this is reflected by the positivity of the so-called relative entropy (beware that it is a convex function while the entropy is concave), which is often called the Kulback-Leibler divergence

$$\mathrm{KL}(p|q) = -\sum_k p_k \log q_k - H(p) = \sum_k p_k \log \frac{p_k}{q_k} \geqslant 0.$$

This KL divergence is similar to a distance in the sense that $\mathrm{KL}(p|q)=0$ if and only if $p=q$ (note however that KL is not symmetric and does not satisfies the triangular inequality). It also has the remarkable property that it is jointly convex in (p,q). It is of paramount importance to compare probability distributions and measures, and form the basis of the fields of information theory and information geometry.

Doing better. One can wonder if it is possible to go below the entropy bound. By the virtue of Shannon theorem, it is not possible if only can only code in sequential order the symbols themselves. From a statistical perspective, it is as if the symbols where considered to be independent. If there is some redundancy in the sequence of symbol (for instance if they are discretization of a smooth function, so that to consecutive symbols are likely to be equal), it is possible to re-transform (in a bijective way) the sequence to make them "more independent". A simple illustration of this idea is given in Figure 1.16, where one computes successive difference of a 1D sequence of symbols (beware to also retain the initial symbol to be able to do the decoding.

Another, more systematic way to leverage such temporal redundancy is by performing run-length-coding, which operate by grouping together sequence of similar symbols and thus coding first a symbol and then the length of the associated group (which is coded entropically). If the sequence is generated by a Markov chain, this method can be shown to asymptotically reach the Shannon bound where now the entropy is the

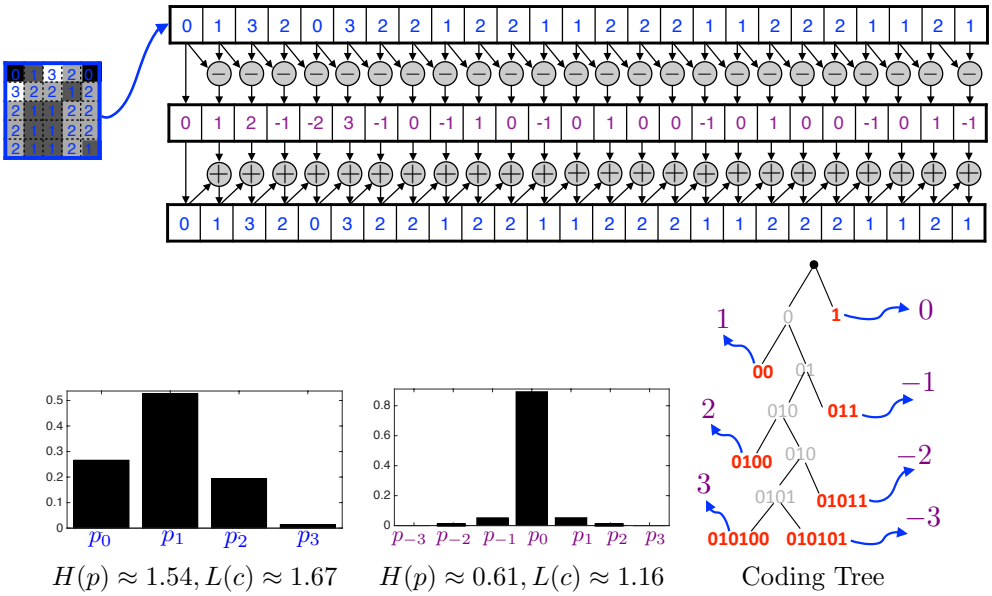


Figure 1.16: Top: retransformation by successive differences. Bottom: comparison of histograms of pixel values and differences, and a code tree for these differences.

entropy associated to the distribution of the Markov chain on infinite sequences (which can be computed as the limit of the entropy for finite sequences).

Chapter 2

Fourier Transforms

The main references for this chapter is [17]. The Fourier transforms offers a perfect blend of analysis (solution of PDEs, approximation of functions), algebra (characters of groups, representation theory) and computer science (the FFT). It is the basics of signal processing because it allows to compute efficiently and study theoretically convolution operator, which are the shift-invariant operators. This chapter offers a glimpse of all these different facets.

2.1 Hilbert spaces and Fourier Transforms

2.1.1 Hilbertian bases.

A large class of method in data sciences (including for instance most signal processing tasks, but also data pre-processing for machine learning) operate by first projecting the input data on some basis. A particularly simple instance of this is when the basis is orthogonal, since in this case one has a simple reconstruction formula and conservation of energy, which is crucial to do a theoretical analysis. We explain this in a general Hilbert space, which can be for instance $\mathcal{H} = L^2([0,1]^d)$ when dealing with continuous signal, or $\mathcal{H} = \mathbb{R}^N$ for discrete signal.

An (complex) Hilbert space $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ is complete, where $\langle \cdot, \cdot \rangle$ is an hermitian inner product (i.e. $\langle f, g \rangle$ is the conjugate of $\langle g, f \rangle$). If it is separable, it can be equipped with an Hilbertian orthogonal basis $(\varphi_n)_{n \in \mathbb{N}}$, which means that one can expand any $f \in \mathcal{H}$ as

$$f = \sum_n \langle f, \varphi_n \rangle \varphi_n$$

where the convergence is in the sense of $\|f\|^2 \overset{\text{def.}}{=} \langle f, f \rangle$, i.e. $\|f - \sum_{n=0}^N \langle f, \varphi_n \rangle \varphi_n\| \to 0$ as $N \to +\infty$. One also have the conservation of energy

$$\|f\|^2 = \sum_n \langle f, \varphi_n \rangle^2.$$

A way to construct such an ortho-basis is using Gram-Schmidt orthogonalization procedure. From some family $(\bar{\varphi}_n)_n$, one defines $\varphi_0 = \bar{\varphi}_0 / \|\bar{\varphi}_0\|$, $\varphi_n = \tilde{\varphi}_n / \|\tilde{\varphi}_n\|$ where $\tilde{\varphi}_n$ is computed as $\tilde{\varphi}_n = \bar{\varphi}_n - \sum_{i<n} a_i \varphi_i$, and by orthogonality $a_i = \langle \bar{\varphi}_n, \varphi_i \rangle$.

On $L^2([-1,1])$ equipped with the usual inner product, orthogonalization of monomials defines the Legendre polynomials, so that

$$\varphi_0(x) = 1, \quad \varphi_1(x) = x, \quad \varphi_2(x) = \frac{1}{2}(3x^2 - 1), \quad \text{etc.}$$

On $L^2(\mathbb{R})$ equipped with a Gaussian measure $e^{x^2} \mathrm{d}x$, this leads to functions of the form $P_n(x) e^{-x^2}$ where P_n are Hermite polynomials, $P_0(x) = 1, P_1(x) = x, P_2(x) = x^2 - 1$. Intuitively, orthogonality forces φ_n to have n "oscillations", e.g. orthogonal polynomials have exactly n zeros.

Another example is provided by the Shannon interpolation theorem, which states that $(\mathrm{sinc}(x-n))_n$ is an orthogonal basis of $\left\{ f \;;\; \mathrm{supp}(\hat{f}) \subset [-\pi,\pi] \right\}$ and the reconstruction formula is $f = \sum_n f(n)\,\mathrm{sinc}(x-n)$ so that $f_n = \langle f, \mathrm{sinc}(x-n)\rangle$. The convergence is in L^2 but it is actually also pointwise as shown in the proof of the previous chapter.

Another way (that we do not detail here) to construct orthogonal bases is to consider the eigenvectors of some symmetric operator. We will show below that Fourier bases can be obtained this way, by considering translation-invariant operators (convolutions).

2.1.2 Fourier basis on $\mathbb{R}/2\pi\mathbb{Z}$.

There is a flurry of different Fourier basis depending on the space on which it is defined. To get an orthogonal basis, it is important to consider compact spaces (otherwise one obtain a notion of Fourier transform, which is not a decomposition on a basis).

On $L^2(\mathbb{T})$ where $\mathbb{T} = \mathbb{R}/2\pi\mathbb{Z}$, equipped with $\langle f, g\rangle \overset{\text{def.}}{=} \frac{1}{2\pi}\int_{\mathbb{T}} f(x)\bar{g}(x)\mathrm{d}x$, one can use the Fourier basis

$$\varphi_n(x) \overset{\text{def.}}{=} e^{inx} \quad \text{for} \quad n \in \mathbb{Z}. \tag{2.1}$$

One thus has

$$f = \sum_n \hat{f}_n e^{in\cdot} \quad \text{where} \quad \hat{f}_n \overset{\text{def.}}{=} \frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-inx}\mathrm{d}x, \tag{2.2}$$

in $L^2(\mathbb{T})$ sense. Pointwise convergence is delicate, see Section 1.2.

Figure 2.1, left, shows examples of the real part of Fourier atoms.

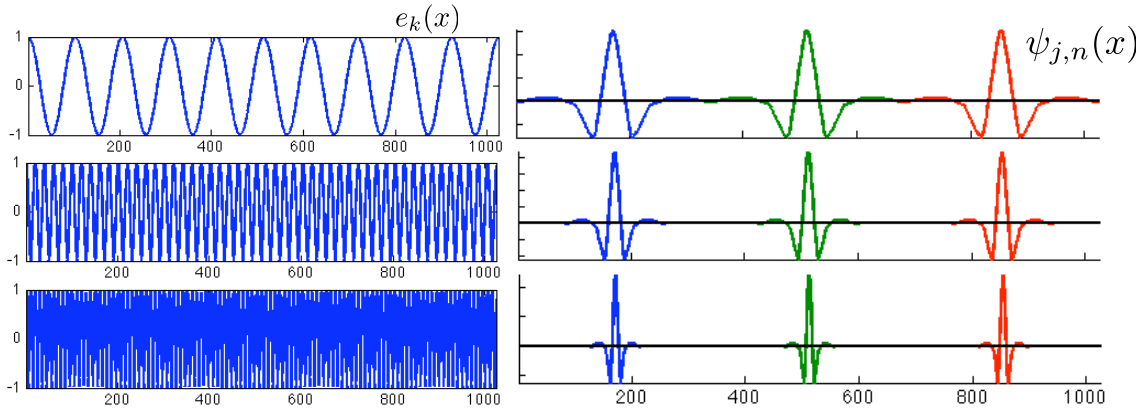


Figure 2.1: Left: 1D Fourier (real part), right: wavelet bases.

We recall that for $f \in L^1(\mathbb{R})$, its Fourier transform is defined as

$$\forall\, \omega \in \mathbb{R}, \quad \hat{f}(\omega) \overset{\text{def.}}{=} \int_{\mathbb{R}} f(x)e^{-\mathrm{i}x\omega}\mathrm{d}x.$$

and this is extended to $L^2(\mathbb{R})$ by density.

The connexion between the Fourier transform on $\mathbb{R}$ and the Fourier coefficients on $\mathbb{T}$ is given by the following diagram

$$\begin{array}{ccc} f(x) & \overset{\mathcal{F}}{\longrightarrow} & \hat{f}(\omega) \\ \text{sampling}\ \downarrow & & \downarrow\ \text{periodization} \\ (f(n))_n & \overset{\text{Fourier serie}}{\longrightarrow} & \sum_n f(n)e^{-\mathrm{i}\omega n} \end{array}.$$

Its commutativity sates

$$\sum_n f(n)e^{-\mathrm{i}\omega n} = \sum_n \hat{f}(\omega - 2\pi n) \tag{2.3}$$

and this is in fact the celebrated Poisson formula (Proposition 1).

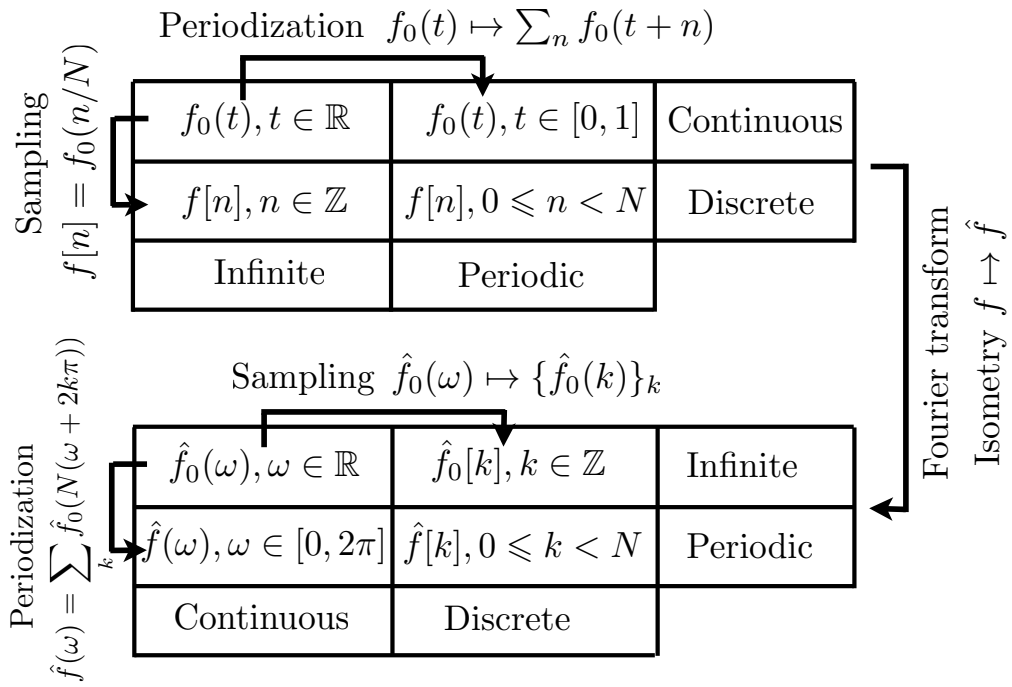


Figure 2.2: The four different settings for Fourier analysis, and the sampling-periodization relationship.

2.2 Convolution on $\mathbb{R}$ and $\mathbb{T}$

2.2.1 Convolution

On $\mathbb{X} = \mathbb{R}$ or $\mathbb{T}$, one defines

$$f \star g(x) = \int_{\mathbb{X}} f(t)g(x-t)\mathrm{d}t. \tag{2.4}$$

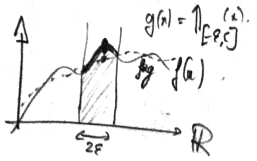


Figure 2.3: Convolution on $\mathbb{R}$.

Young's inequality shows that this quantity is well defined if $(f, g) \in L^p(\mathbb{X}) \times L^q(\mathbb{X})$

$$\frac{1}{p} + \frac{1}{q} = 1 + \frac{1}{r} \quad \Longrightarrow \quad f \star g \in L^r(\mathbb{X}) \quad \text{and} \quad \|f \star g\|_{L^r(\mathbb{X})} \leqslant \|f\|_{L^p(\mathbb{X})}\|g\|_{L^q(\mathbb{X})}. \tag{2.5}$$

This shows that if $f \in L^1(\mathbb{X})$, then one has the map $g \in L^p(\mathbb{X}) \mapsto f \star g \in L^p(\mathbb{X})$ is a continuous map on $L^p(\mathbb{X})$. Furthermore, when $r = \infty$, $f \star g \in \mathcal{C}^0(\mathbb{X})$ is a continuous function (which shows some regularizing effect). Note that for $\mathbb{X} = \mathbb{T}$, $p < q \quad \Longrightarrow \quad L^q(\mathbb{T}) \subset L^p(\mathbb{T})$, so that $L^\infty(\mathbb{X})$ is the smallest space.

Convolution is mostly used in order to regularize functions. For instance, if $f \in L^1(\mathbb{X})$ and $g \in \mathcal{C}^1(\mathbb{X})$ is bounded, then $f \star g$ is differentiable and $(f \star g)' = f \star g'$. This is used to produce smooth approximate

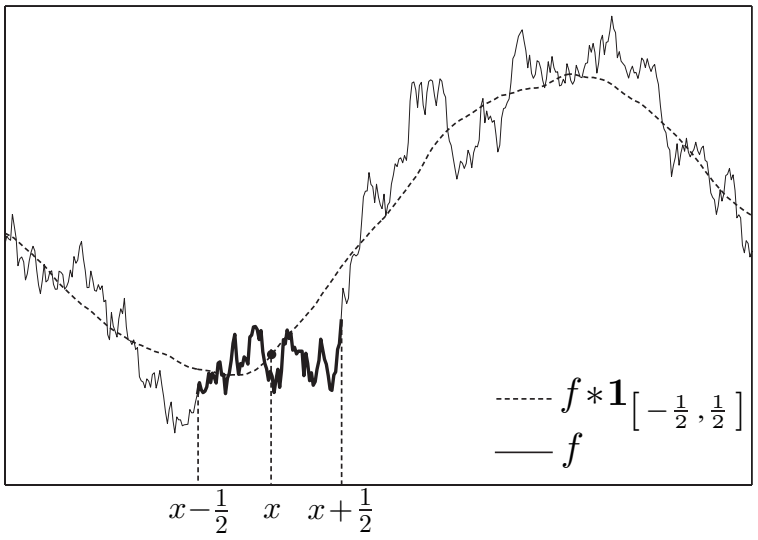


Figure 2.4: Signal filtering with a box filter (running average).

identity ($\rho_\varepsilon = \frac{1}{\varepsilon}\rho(\cdot/\varepsilon)$)$_\varepsilon$ with convergence $f \star \rho_\varepsilon \to f$ in $L^p(\mathbb{X})$ for $1 \leqslant p < +\infty$ of smooth approximations (and convergence in $L^\infty(\mathbb{X})$ if f is uniformly continuous). This is also used for denoising applications in signal and image processing.

For $(f, g) \in L^1(\mathbb{X})^2$ (so on $\mathbb{X} = \mathbb{T}$, also in any $L^p(\mathbb{X})$), one has

$$\mathcal{F}(f \star g) = \hat{f} \odot \hat{g} \tag{2.6}$$

which means that $\mathcal{F}$ is a morphism of algebra. For instance if $\mathbb{X} = \mathbb{R}$, its range is included in the algebra of continuous functions with vanishing limits in $\pm\infty$.

As shown in Figure 2.7

Note also that the convolution is used extensively in probability and statistics, because if we denote f_X the probability density (with respect to Lebesgue or any measure) of some random vector X, then if X and Y are independent random variable, then $f_{X+Y} = f_X \star f_Y$.

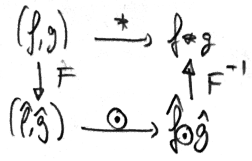


Figure 2.6: Commutative diagram of convolution-Fourier.

`Associated code: fourier/test_denoising.m and fourier/test_fft2.m`

2.2.2 Translation Invariant Operators

Translation invariant operators (which commutes with translation) are fundamental because in most situations, input (signal, image, etc) should be processed without spatial preference. The following propositions shows that any translation invariant[1] (i.e. which commutes with translations) operator is actually a "convolution" against a distribution with bounded Fourier transform. The proof and conclusion (regularity of the convolution kernel) vary depending on the topology on the input and output spaces. We first study the case of convolution mapping to continuous functions.

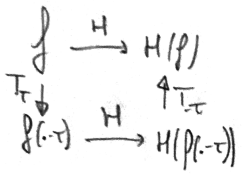


Figure 2.8: Commutative diagram for translation invariance.

Proposition 2. *We define $T_\tau f = f(\cdot - \tau)$. A bounded linear operator $H : (L^2(\mathbb{X}), \|\cdot\|_2) \to (C^0(\mathbb{X}), \|\cdot\|_\infty)$ is such that for all τ, $H \circ T_\tau = T_\tau \circ H$ if and only if*

$$\forall f \in L^2(\mathbb{T}), \quad H(f) = f \star g$$

[1] One should rather actually say "translation equivariant".

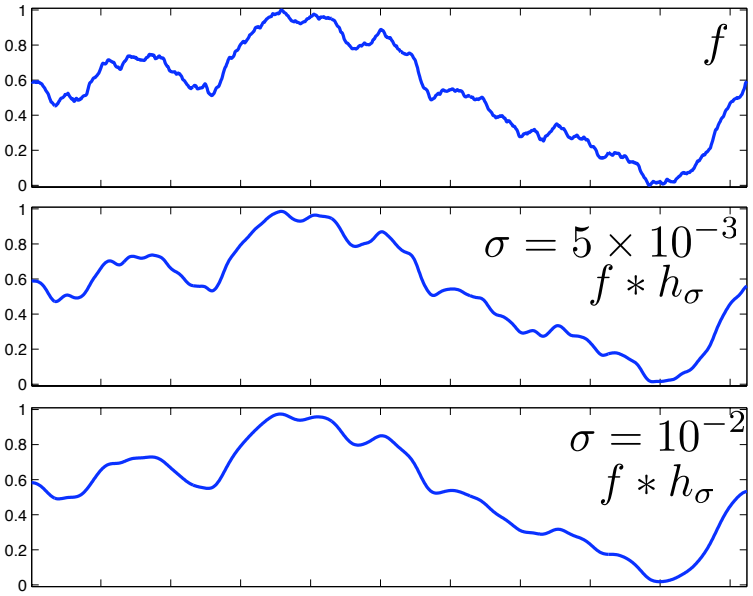


Figure 2.5: Filtering an irregular signal with a Gaussian filter of increasing filter size σ.

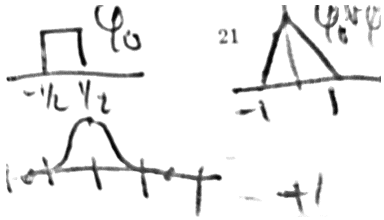


Figure 2.7: Cardinal splines are defined by successive convolutions;

with $g \in L^2(\mathbb{X})$.

The heuristic proof using distributions is that $f(x) = (f \star \delta)(x) = \int f(t)\delta(x-t)\mathrm{d}t$ and thus by linearity

$$Hf = H(\int f(t)\delta(\cdot - t)\mathrm{d}t) = \int f(t)H(\delta(\cdot - t))\mathrm{d}t = \int f(t)H(T_t\delta)\mathrm{d}t \int f(t)T_t(H(\delta))\mathrm{d}t \int f(t)H(\delta)(\cdot - t)\mathrm{d}t = f \star H(\delta).$$

Proof. Thanks to (2.5) (and the remark in the case $r = \infty$), $T : f \mapsto f \star g$ with $g \in L^2(\mathbb{X})$ is indeed a continuous operator from $L^2(\mathbb{X})$ to $C^0(\mathbb{X})$. Furthermore

$$(H \circ T_\tau)(f) = \int_{\mathbb{X}} f((\cdot - \tau) - y)g(y)\mathrm{d}\tau = (f \star g)(\cdot - \tau) = T_\tau(Hf),$$

so that such an H is translation-invariant.

Conversely, we define $\ell : f \mapsto H(f)(0) \in \mathbb{R}$, which is legit since $H(f) \in \mathcal{C}^0(\mathbb{X})$. Since H is continuous, there exists C such that $\|Hf\|_\infty \leqslant C\|f\|_2$, and hence $|\ell(f)| \leqslant C\|f\|_2$, so that f is a continuous linear form on the Hilbert space $L^2(\mathbb{X})$. Hence, according to Fréchet-Riesz theorem, there exists $h \in L^2(\mathbb{X})$ such that $\ell(f) = \langle f, h\rangle$. Hence, $\forall\, x \in \mathbb{X}$,

$$H(f)(x) = T_{-x}(Hf)(0) = H(T_{-x}f)(0) = \ell(T_{-x}f) = \langle T_{-x}f, h\rangle = \int_{\mathbb{X}} f(y+x)h(y)\mathrm{d}y = f \star \bar{h}(x).$$

where $g \overset{\text{def.}}{=} \bar{h} = h(-\cdot) \in L^2(\mathbb{X})$. □

We now study, on $\mathbb{T}$, the case of convolution which can output non-continuous functions. In this case, the kernel can be a "distribution", so the convolution is defined over the Fourier domain.

Proposition 3. *A bounded linear operator $H : L^2(\mathbb{T}) \to L^2(\mathbb{T})$ is such that for all τ, $H \circ T_\tau = T_\tau \circ H$ if and only if*

$$\forall\, f \in L^2(\mathbb{T}), \quad \mathcal{F}(H(f)) = \hat{f} \odot c$$

where $c \in \ell^\infty(\mathbb{Z})$ (a bounded sequence).

Proof. We denote $\varphi_n \overset{\text{def.}}{=} e^{\mathrm{i}n\cdot}$. One has

$$H(\varphi_n) = e^{\mathrm{i}n\tau} H(T_\tau(\varphi_n)) = e^{\mathrm{i}n\tau} T_\tau(H(\varphi_n)).$$

Thus, for all n,

$$\langle H(\varphi_n),\, \varphi_m\rangle = e^{\mathrm{i}n\tau}\langle T_\tau \circ H(\varphi_n),\, \varphi_m\rangle = e^{\mathrm{i}n\tau}\langle H(\varphi_n),\, T_{-\tau}(\varphi_m)\rangle = e^{\mathrm{i}(n-m)\tau}\langle H(\varphi_n),\, \varphi_m\rangle$$

So for $n \neq m$, $\langle H(\varphi_n),\, \varphi_m\rangle = 0$, and we define $c_n \overset{\text{def.}}{=} \langle H(\varphi_n),\, \varphi_n\rangle$. Since H is continuous, $\|Hf\|_{L^2(\mathbb{T})} \leqslant C\|f\|_{L^2(\mathbb{T})}$ for some constant C, and thus by Cauchy-Schwartz

$$|c_n| = |\langle H(\varphi_n),\, \varphi_n\rangle| \leqslant \|H(\varphi_n)\|\|\varphi_n\| \leqslant C$$

because $\|\varphi_n\| = 1$, so that $c \in \ell^\infty(\mathbb{Z})$. By continuity, recalling that by definition $\hat{f}_n \overset{\text{def.}}{=} \langle f,\, \varphi_n\rangle$,

$$H(f) = \lim_N H\Big(\sum_{n=-N}^{N} \hat{f}_n \varphi_n\Big) = \lim_N \sum_{n=-N}^{N} \hat{f}_n H(\varphi_n) = \lim_N \sum_{n=-N}^{N} c_n \hat{f}_n \varphi_n = \sum_{n\in\mathbb{Z}} c_n \hat{f}_n \varphi_n$$

so that in particular one has the desired result. □

This theorem thus states that translation invariant operators are those which are "diagonal" in the Fourier ortho-basis.

2.2.3 Revisiting Poisson formula using distributions.

Informally, the Fourier series

$$\sum_n f(n) e^{-\mathrm{i}\omega n}$$

can be thought as the Fourier transform $\mathcal{F}(\Pi_1 \odot f)$ of the discrete distribution

$$\Pi_1 \odot f = \sum_n f(n)\delta_n \quad \text{where} \quad \Pi_s = \sum_n \delta_{sn}$$

for $s \in \mathbb{R}$, where δ_a is the Dirac mass at location $a \in \mathbb{R}$, i.e. the distribution such that $\int f \mathrm{d}(\mathrm{d}_a) = f(a)$ for any continuous f. Indeed, one can multiply a distribution by a continuous function, and the definition of the Fourier transform of a distribution μ is a distributions $\mathcal{F}(\mu)$ such that that

$$\forall\, g \in \mathcal{S}(\mathbb{R}), \quad \int_{\mathbb{R}} g(x)\mathrm{d}\mathcal{F}(\mu) = \int_{\mathbb{R}} \mathcal{F}^*(g)\mathrm{d}\mu, \quad \text{where} \quad \mathcal{F}^*(g) \overset{\text{def.}}{=} \int_{\mathbb{R}} g(x) e^{\mathrm{i}x\cdot}\mathrm{d}x,$$

where $\mathcal{S}(\mathbb{R})$ are smooth and rapidly decaying (Schwartz class) functions.

The Poisson formula (2.3) can thus be interpreted as

$$\mathcal{F}(\Pi_1 \odot f) = \sum_n \hat{f}(\cdot - 2\pi n) = \int_{\mathbb{R}} \hat{f}(\cdot - \omega) \mathrm{d}\Pi_{2\pi}(\omega) = \hat{f} \star \Pi_{2\pi}$$

Since $\mathcal{F}^{-1} = \frac{1}{2\pi}\mathcal{S} \circ \mathcal{F}$ where $\mathcal{S}(f) = f(-\cdot)$, one has, applying this operator on both sides

$$\Pi_1 \odot f = \frac{1}{2\pi}\mathcal{S} \circ \mathcal{F}(\hat{f} \star \Pi_{2\pi}) = \mathcal{S}(\frac{1}{2\pi}\mathcal{F}(\hat{f}) \odot \hat{\Pi}_{2\pi}) = \mathcal{S}(\frac{1}{2\pi}\mathcal{F}(\hat{f})) \odot \mathcal{S}(\hat{\Pi}_{2\pi}) = \hat{\Pi}_{2\pi} \odot f.$$

This can be interpreted as the relation

$$\hat{\Pi}_{2\pi} = \Pi_1 \quad \Longrightarrow \quad \hat{\Pi}_1 = 2\pi\Pi_{2\pi}.$$

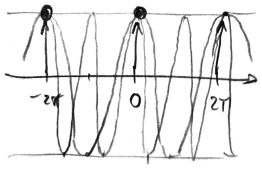


Figure 2.9: Sine wave being summed in the Poisson formula.

To intuitively understand this relation, one can compute a finite Fourier series

$$\sum_{n=-N}^{N} e^{-\mathrm{i}n\omega} = \frac{\sin((N+1/2)x)}{\sin(x/2)}$$

which is a smooth function which grows unbounded with $N \to +\infty$ as $N \to +\infty$.

2.3 Finite Fourier Transform and Convolution

2.3.1 Discrete Ortho-bases

Discrete signals are finite dimensional vector $f \in \mathbb{C}^N$ where N is the number of samples and where each f_n is the value of the signal at a 1D or 2D location. For a 2-D images $f \in \mathbb{C}^N \simeq \mathbb{C}^{N_0 \times N_0}$, $N = N_0 \times N_0$, where N_0 is the number of pixels along each direction.

Discrete signals and images are processed using a discrete inner product that mimics the continuous L^2 inner product

$$\langle f, g \rangle = \sum_{n=0}^{N-1} f_n \bar{g}_n.$$

One thus defines a distance between discretized vectors as

$$\|f - g\|^2 = \sum_{n=0}^{N-1} |f_n - g_n|^2.$$

Exactly as in the continuous case, a discrete orthogonal basis $\{\psi_m\}_{0 \leqslant m < N}$ of $\mathbb{C}^N$, satisfies

$$\langle \psi_m, \psi_{m'} \rangle = \delta_{m-m'}. \tag{2.7}$$

The decomposition of a signal in such an ortho-basis is written

$$f = \sum_{m=0}^{N-1} \langle f, \psi_m \rangle \psi_m.$$

It satisfies a conservation of energy

$$\|f\|^2 = \sum_{n=0}^{N-1} |f_n|^2 = \sum_{m=0}^{N-1} |\langle f, \psi_m \rangle|^2$$

Computing the set of all inner product $\{\langle f, \psi_m \rangle\}_{0 \leqslant m < N}$ is done in a brute force way in $O(N^2)$ operations. This is not feasible for large datasets where N is of the order of millions. When designing an ortho-basis, one should keep this limitation in mind and enforce some structure in the basis elements so that the decomposition can be computed with fast algorithm. This is the case for the Fourier and wavelet bases, that enjoy respectively $O(N \log(N))$ and $O(N)$ algorithms.

2.3.2 Discrete Fourier transform

We denote $f = (f_n)_{n=0}^{N-1} \in \mathbb{R}^N$, but we insists that such vector should really be understood as being indexed by $n \in \mathbb{Z}/N\mathbb{Z}$, which is a finite commutative group for the addition. This corresponds to using periodic boundary conditions.

The discrete Fourier transform is defined as

$$\forall\, k = 0, \ldots, N-1, \quad \hat{f}_k \stackrel{\text{def.}}{=} \sum_{n=0}^{N-1} f_n e^{-\frac{2\mathrm{i}\pi}{N}kn} = \langle f,\, \varphi_k \rangle \quad \text{where} \quad \varphi_k \stackrel{\text{def.}}{=} (e^{\frac{2\mathrm{i}\pi}{N}kn})_{n=0}^{N-1} \in \mathbb{C}^N \tag{2.8}$$

where the canonical inner product on $\mathbb{C}^N$ is $\langle u,\, v \rangle = \sum_{n=1}^{N} u_n \bar{v}_n$ for $(u, v) \in (\mathbb{C}^N)^2$. This definition can intuitively be motivated by sampling the Fourier basis $x \mapsto e^{\mathrm{i}kx}$ on $\mathbb{R}/2\pi\mathbb{Z}$ at equi-spaced points $(\frac{2\pi}{N}n)_{n=0}^{N-1}$. The following proposition shows that this corresponds to a decomposition in an ortho-basis.

Proposition 4. *One has*

$$\langle \varphi_k,\, \varphi_\ell \rangle = \begin{cases} N & \text{if} \quad k = \ell, \\ 0 & \text{otherwise.} \end{cases}$$

In particular, this implies

$$\forall\, n = 0, \ldots, N-1, \quad f_n = \frac{1}{N} \sum_k \hat{f}_k e^{\frac{2\mathrm{i}\pi}{N}kn}. \tag{2.9}$$

Proof. One has, if $k \neq \ell$

$$\langle \varphi_k,\, \varphi_\ell \rangle = \sum_n e^{\frac{2\mathrm{i}\pi}{N}(k-\ell)n} = \frac{1 - e^{\frac{2\mathrm{i}\pi}{N}(k-\ell)N}}{1 - e^{\frac{2\mathrm{i}\pi}{N}(k-\ell)}} = 0$$

which is the sum of a geometric serie (equivalently, sum of equi-spaced points on a circle). The inversion formula is simply $f = \sum_k \langle f,\, \varphi_k \rangle \frac{\varphi_k}{\|\varphi_k\|^2}$. □

2.3.3 Fast Fourier transform

Assuming $N = 2N'$, one has

$$\hat{f}_{2k} = \sum_{n=0}^{N'-1} (f_n + f_{n+N/2}) e^{-\frac{2\mathrm{i}\pi}{N'}kn}$$

$$\hat{f}_{2k+1} = \sum_{n=0}^{N'-1} e^{-\frac{2\mathrm{i}\pi}{N'}n} (f_n - f_{n+N/2}) e^{-\frac{2\mathrm{i}\pi}{N'}kn}.$$

For the second line, we used the computation

$$e^{-\frac{2\mathrm{i}\pi}{N}(2k+1)(n+N/2)} = e^{-\frac{2\mathrm{i}\pi}{N}(2kn+kN+n+N/2)} = -e^{-\frac{2\mathrm{i}\pi}{N'}n} e^{-\frac{2\mathrm{i}\pi}{N'}kn}.$$

Denoting $\mathcal{F}_N(f) = \hat{f}$ the discrete Fourier transform on $\mathbb{R}^N$, and introducing the notation $f_e = (f_n + f_{n+N/2})_n \in \mathbb{R}^{N'}$, $f_o = (f_n - f_{n+N/2})_n \in \mathbb{R}^{N'}$ and $\alpha_N = (e^{-\frac{2\mathrm{i}\pi}{N'}n})_n \in \mathbb{R}^{N'}$, one has the following recursion formula

$$\mathcal{F}_N(f) = \mathcal{I}_N(\mathcal{F}_{N/2}(f_e), \mathcal{F}_{N/2}(f_o \odot \alpha_N))$$

where $\mathcal{I}_N$ is the "interleaving" operator, defined by $\mathcal{I}_N(a, b) \stackrel{\text{def.}}{=} (a_1, b_1, a_2, b_2, \ldots, a_{N'}, b_{N'})$. There iterations define the so-called Fast Fourier Transform algorithm, which works here when N is a power of 2. These iterations can be extended to arbitrary number N, but a workaround is to simply pad with 0 (or use more complicated extensions) to have vector with size that are power of 2.

This algorithm can also be interpreted as a factorization of the Fourier matrix into $O(\log(N))$ product of matrices.

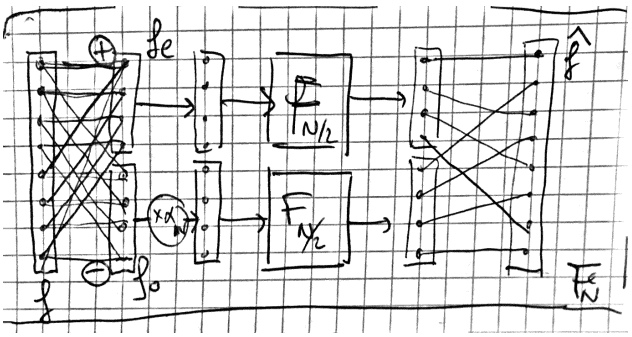


Figure 2.10: Diagram of one step of the FFT.

Denoting $C(N)$ the numerical complexity (number of elementary operations) associated to the computation of $\hat{f}$, one thus has

$$C(N) = 2C(N/2) + NK \tag{2.10}$$

where KN is the complexity of N complex additions and $N/2$ multiplications. Making the change of variable

$$\ell \stackrel{\text{def.}}{=} \log_2(N) \quad \text{and} \quad T(\ell) \stackrel{\text{def.}}{=} \frac{C(N)}{N}$$

i.e. $C(N) = 2^\ell T(\ell)$, the relation (2.10) becomes

$$2^\ell T(\ell) = 2 \times 2^{\ell-1} T(\ell-1) + 2^\ell K \quad \Longrightarrow \quad T(\ell) = T(\ell-1) + K \quad \Longrightarrow \quad T(\ell) = T(0) + K\ell$$

and using the fact that $T(0) = C(1)/1 = 0$, one obtains

$$C(N) = KN \log_2(N).$$

This complexity should be contrasted with the complexity $O(N^2)$ of directly computing the N coefficients (2.8), each involving a sum of size N.

2.3.4 Finite convolution

For $(f, g) \in (\mathbb{R}^N)^2$, one defines $f \star g \in \mathbb{R}^N$ as

$$\forall\, n = 0, \ldots, N-1, \quad (f \star g)_n \stackrel{\text{def.}}{=} \sum_{k=0}^{N-1} f_k g_{n-k} = \sum_{k+\ell=n} f_k g_\ell \tag{2.11}$$

where one should be careful that here $+$ and $-$ should be understood modulo N (vectors should be seen as being defined on the group $\mathbb{Z}/N\mathbb{Z}$, or equivalently, one uses periodic boundary conditions). This defines an commutative algebra structure $(\mathbb{R}^N, +, \star)$, with neutral element the "Dirac" $\delta_0 \stackrel{\text{def.}}{=} (1, 0, \ldots, 0)^\top \in \mathbb{R}^N$. The following proposition shows that $\mathcal{F} : f \mapsto \hat{f}$ is an algebra bijective isometry (up to a scaling by $\sqrt{N}$ of the norm) mapping to $(\mathbb{R}^N, +, \odot)$ with neutral element $\mathbb{1}_N = (1, \ldots, 1) \in \mathbb{R}^N$.

Proposition 5. *One has* $\mathcal{F}(f \star g) = \hat{f} \odot \hat{g}$.

Proof. We denote $T : g \mapsto f \star g$. One has

$$(T\varphi_\ell)_n = \sum_k f_k e^{\frac{2\mathrm{i}\pi}{N}\ell(n-k)} = e^{\frac{2\mathrm{i}\pi}{N}\ell n} \hat{f}_\ell.$$

This shows that $(\varphi_\ell)_\ell$ are the N eigenvectors of T with associated eigenvalues $\hat{f}_\ell$. So T is diagonalizable in this basis. Denoting $F = (e^{-\frac{2\mathrm{i}\pi}{N}kn})_{k,n}$ the matrix of the Fourier transform, the Fourier inversion formula (2.9)

reads $F^{-1} = \frac{1}{N} F^*$ where $F^* = \bar{F}^\top$ is the adjoint matrix (trans-conjugate). The diagonalization of T now reads

$$T = F^{-1} \operatorname{diag}(\hat{f}) F = \quad \Longrightarrow \quad \mathcal{F}(Tg) = \operatorname{diag}(\hat{f}) F g \quad \Longrightarrow \quad \mathcal{F}(f \star g) = \operatorname{diag}(\hat{f}) \hat{g}.$$

□

This proposition shows that one can compute in $O(N \log(N))$ operation via the formula

$$f \star g = \mathbb{F}^{-1}(\hat{f} \odot \hat{g}).$$

This is very advantageous with respect to the naive implementation of formula (2.11), in the case where f and g have large support. In case where $|\operatorname{Supp}(g)| = P$ is small, then direct implementation is $O(PN)$ which might be advantageous. An example is $g = [1, 1, 0, \ldots, 0, 1]/3$, the moving average, where

$$(f \star g)_n = \frac{f_{n-1} + f_n + f_{n+1}}{3}$$

needs $3N$ operations.

WARNING: this is wrong, one needs also to carie reminders in the multiplication. An example of application of the FFT is the multiplication of large polynomial, and thus of large integers (viewing the expansion in a certain basis as a polynomial). Indeed

$$(\sum_{i=0}^{A} a_i X_i)(\sum_{j=0}^{B} b_i X^j) = \sum_{k=0}^{A+B} (\sum_{i+j=k} a_i b_j) X^k$$

One can write $\sum_{i+j=k} a_i b_j = (\bar{a} \star \bar{b})_k$ when one defines $\bar{a}, \bar{b} \in \mathbb{R}^{A+B}$ by zero padding.

2.4 Discretisation Issues

Beside computing convolutions, another major application of the FFT is to approximate the Fourier transform and its inverse, thus leading to a computationally efficient spectral interpolation method.

2.4.1 Fourier approximation via spatial zero padding.

It is possible to view the discrete finite Fourier transform (2.8) as a first order approximation to compute Fourier coefficients, or rather actually samples from the Fourier transform (1.2). Supposing that f is a smooth enough function is supported on $[0, 1]$, we consider the discrete Fourier transform of the vector $f^Q \overset{\text{def.}}{=} (f(n/N))_{n=0}^{Q-1} \in \mathbb{R}^Q$ where $Q \geqslant N$ induced a padding by 0 (since $f(n/N) = 0$ for $n > N$)

$$\forall\, k \in [\![-\frac{Q}{2}, \frac{Q}{2}]\!], \quad \frac{1}{N} \hat{f}_k^Q = \frac{1}{N} \sum_{n=0}^{N-1} f\left(\frac{n}{N}\right) e^{-\frac{2\mathrm{i}\pi}{Q} nk} \approx \int_0^1 f(x) e^{-\frac{2k\mathrm{i}\pi}{T} x} \mathrm{d}x = \hat{f}\left(\frac{2k\pi}{T}\right) \quad \text{where} \quad T \overset{\text{def.}}{=} \frac{Q}{N}.$$

The approximation is first order accurate, i.e. $O(1/N)$ for a C^1 function f.

2.4.2 Fourier interpolation via spectral zero padding.

One can reverse the roles of space and frequency in the previous construction. If one has at its disposal N uniform discrete samples $f^N = (f_n^N)_{n=0}^{N-1}$, one can compute its discrete Fourier transform $\mathcal{F}(f^N) = \hat{f}^N$ (in $O(N \log(N))$ operations with the FFT),

$$\hat{f}_k^N \overset{\text{def.}}{=} \sum_{n=0}^{N-1} f_n^N e^{-\frac{2\mathrm{i}\pi}{N} nk},$$

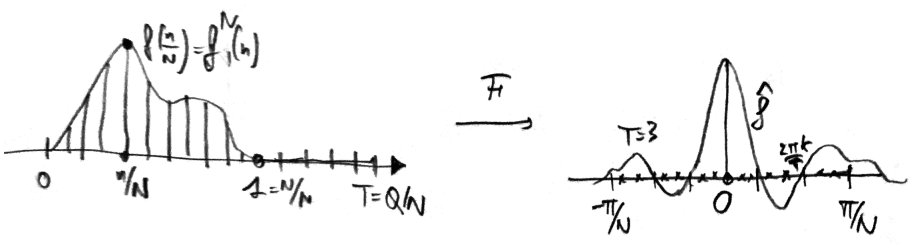


Figure 2.11: Fourier transform approximation by zero-padding in the spatial domain.

and then zero-pad it to obtain a vector of length Q. For simplicity, we assume $N = 2N' + 1$ is odd, and this computation can be also done (but is more involved) with even size. Indexing the frequencies as $-N' \leqslant k \leqslant N'$ The padding vector is of the form,

$$\tilde{f}^Q \stackrel{\text{def.}}{=} (0, \ldots, 0, \hat{f}^N, 0, \ldots, 0) \in \mathbb{R}^Q$$

WARNING: there is most likely an error in the following derivation, should be rewritten. One can then compute the (with a normalization constant Q/N) inverse discrete Fourier transform of size Q (in $O(Q \log(Q))$ operations with the FFT) to obtain

$$\begin{aligned}
\frac{Q}{N}\mathcal{F}^{-1}(\tilde{f}^Q)_\ell &= \frac{Q}{N} \times \frac{1}{N} \sum_{k=-N'}^{N'} \hat{f}^N_k e^{\frac{2\mathrm{i}\pi}{Q}\ell k} = \frac{1}{N} \sum_{k=-N'}^{N'} \sum_{n=0}^{N-1} f^N_n e^{\frac{2\mathrm{i}\pi}{N}nk} e^{\frac{2\mathrm{i}\pi}{Q}\ell k} \\
&= \sum_{n=0}^{N-1} f^N_n \frac{1}{N} \sum_{k=-N'}^{N'} e^{2\mathrm{i}\pi(-\frac{n}{N}+\frac{\ell}{Q})k} = \sum_{n=0}^{N-1} f^N_n \frac{\sin\left[\pi N\left(\frac{\ell}{Q}-\frac{n}{N}\right)\right]}{N\sin\left[\pi\left(\frac{\ell}{Q}-\frac{n}{N}\right)\right]} \\
&= \sum_{n=0}^{N-1} f^N_n \operatorname{sinc}_N\left(\frac{\ell}{T}-n\right) \quad \text{where} \quad T \stackrel{\text{def.}}{=} \frac{Q}{N} \quad \text{and} \quad \operatorname{sinc}_N(u) \stackrel{\text{def.}}{=} \frac{\sin(\pi u)}{N\sin(\pi u/N)}.
\end{aligned}$$

Here we use the following summation rule for geometric series for $\rho = e^{\mathrm{i}\omega}$, $a = -b$, $\omega = 2\pi\left(-\frac{n}{N} + \frac{\ell}{Q}\right)$,

$$\sum_{i=a}^{b} \rho^i = \frac{\rho^{a-\frac{1}{2}} - \rho^{b+\frac{1}{2}}}{\rho^{-\frac{1}{2}} - \rho^{\frac{1}{2}}} = \frac{\sin((b+\frac{1}{2})\omega)}{\sin(\omega/2)}.$$

This zero-padding method leads to a discrete version of the Shannon interpolation formula (1.9), which allows to comput the interpolation on a grid of size Q are cost $O(Q\log(Q))$. Increasing N increases the accuracy of the formula, since $\operatorname{sinc}_N \to \operatorname{sinc}$ as $N \to +\infty$.

2.5 Fourier in Multiple Dimensions

The Fourier transform is extended from 1-D to arbitrary finite dimension $d > 1$ by tensor product.

2.5.1 On Continuous Domains

On $\mathbb{R}^d$. The crux of the power of Fourier transform in arbitrary dimension is that a product of elementary 1-D sine waves is still a sine wave

$$\prod_{\ell=1}^{d} e^{\mathrm{i}x_\ell\omega_\ell} = e^{\mathrm{i}\langle x,\omega\rangle}$$

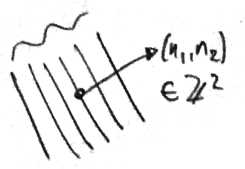


Figure 2.13: 2-D sine wave.

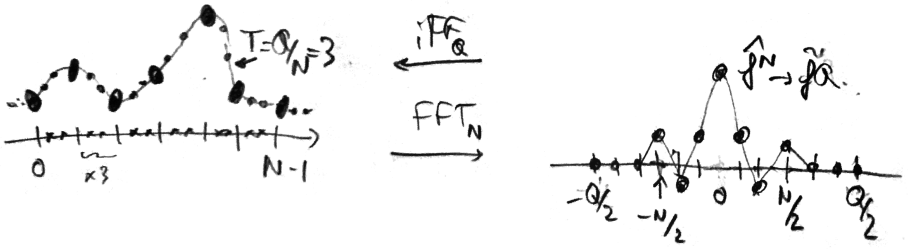


Figure 2.12: Interpolation by zero-padding in the frequency domain.

moving orthogonally to the wave vector $\omega = (\omega_\ell)_{\ell=1}^d \in \mathbb{R}^d$. Here $\langle x, \omega \rangle = \sum_\ell x_\ell \omega_\ell$ is the canonical inner product on $\mathbb{R}^d$.

The definition of the Fourier transform and its inverse are

$$\forall\, \omega \in \mathbb{R}^d, \quad \hat{f}(\omega) \stackrel{\text{def.}}{=} \int_{\mathbb{R}^d} f(x) e^{-\mathrm{i}\langle x, \omega \rangle} \mathrm{d}x,$$
$$\forall\, x \in \mathbb{R}^d, \quad f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} f(x) e^{\mathrm{i}\langle x, \omega \rangle} \mathrm{d}\omega,$$

under hypotheses of integrability matching exactly those in 1-D.

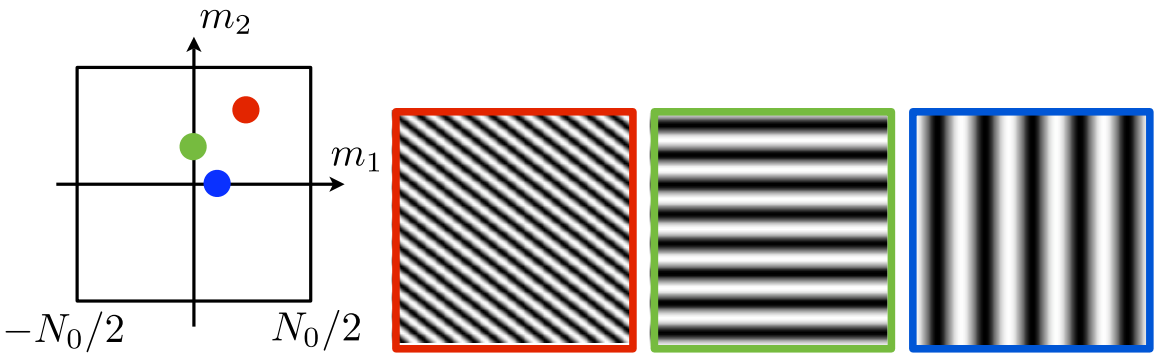


Figure 2.14: 2D Fourier orthogonal bases.

On $(\mathbb{R}/2\pi\mathbb{Z})^d$. Given an Hilbertian basis $(\varphi_{n_1})_{n_1 \in \mathbb{N}}$ of $L^2(\mathbb{X})$, one construct an Hilbertian basis of $L^2(\mathbb{X}^d)$ by tensorization

$$\forall\, n = (n_1, \ldots, n_d) \in \mathbb{N}^d, \quad \forall\, x \in \mathbb{X}^d, \quad \varphi_n(x) = \varphi_{n_1}(x_1) \ldots \varphi_{n_d}(x_d). \tag{2.12}$$

Orthogonality is simple to check, and one can also prove convergence for sum of the form $\sum_{\|n\|_\infty \leqslant N} \langle f, \varphi_n \rangle \varphi_n \to f$ in $L^2(\mathbb{X}^d)$.

For the multi-dimensional torus $(\mathbb{R}/2\pi\mathbb{Z})^d$, using the Fourier basis (2.1), this leads to consider the basis

$$\forall\, n \in \mathbb{R}^d, \quad \varphi_n(x) = e^{\mathrm{i}\langle x, n \rangle}$$

which is indeed an Hilbertian orthonormal basis for the inner product $\langle f, g \rangle \stackrel{\text{def.}}{=} \frac{1}{(2\pi)^d} \int_{\mathbb{T}^d} f(x) \bar{g}(x) \mathrm{d}x$. This defines the Fourier transform and the reconstruction

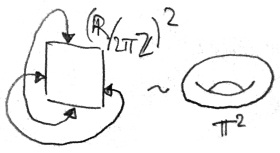


Figure 2.15: The 2-dimensional torus $\mathbb{T}^2 = (\mathbb{R}/2\pi\mathbb{Z})^2$

formula on $L^2(\mathbb{T}^d)$

$$\hat{f}_n \overset{\text{def.}}{=} \frac{1}{(2\pi)^d} \int_{\mathbb{T}^d} f(x) e^{-\mathrm{i}\langle x,\, n\rangle} \mathrm{d}x \quad \text{and} \quad f = \sum_{n \in \mathbb{Z}^d} \hat{f}_n e^{\mathrm{i}\langle x,\, n\rangle}.$$

2.5.2 On Discrete Domains

Discrete Fourier Transform. On d-dimensional discrete domain of the form

$$n = (n_1, \ldots, n_d) \in \mathbb{Y}_d \overset{\text{def.}}{=} [\![1, N_1]\!] \times \ldots \times [\![1, N_d]\!]$$

(we denote $[\![a, b]\!] \overset{\text{def.}}{=} \{i \in \mathbb{Z} \,;\, a \leqslant i \leqslant b\}$) of $N = N_1 \ldots N_d$ points, with periodic boundary conditions, one defines an orthogonal basis $(\varphi_k)_k$ by the same tensor product formula as (2.12) but using the 1-D discrete Fourier basis (2.8)

$$\forall\, (k, n) \in \mathbb{Y}_d^2, \quad \varphi_k(n) = \varphi_{k_1}(n_1) \ldots \varphi_{k_d}(n_d) = \prod_{\ell=1}^{d} e^{\frac{2\mathrm{i}\pi}{N_\ell} k_\ell n_\ell} = e^{2\mathrm{i}\pi \langle k,\, n\rangle_{\mathbb{Y}_d}} \tag{2.13}$$

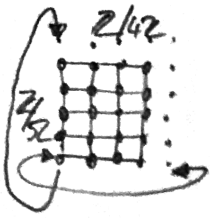


Figure 2.16: Discrete 2-D torus.

where we used the (rescaled) inner product

$$\langle k,\, n\rangle_{\mathbb{Y}_d} \overset{\text{def.}}{=} \sum_{\ell=1}^{d} \frac{k_\ell n_\ell}{N_\ell}. \tag{2.14}$$

The basis $(\varphi_k)_k$ is indeed orthonormal for this inner product. The Fourier transform gathers inner products in this basis, and (similarly to the 1-D case) the convention is to not normalize them with $(N_\ell)_\ell$, so that

$$\forall\, k \in \mathbb{Y}_d, \quad \hat{f}_k \overset{\text{def.}}{=} \sum_{n \in \mathbb{Y}_d} f_n e^{-\mathrm{i}\langle k,\, n\rangle_{\mathbb{Y}_d}},$$
$$\forall\, n \in \mathbb{Y}_d, \quad f_n = \frac{1}{N} \sum_{k \in \mathbb{Y}_d} \hat{f}_k e^{\mathrm{i}\langle k,\, n\rangle_{\mathbb{Y}_d}}.$$

Fast Fourier Transform. We detail the algorithm in dimension $d = 2$ for notation simplicity, but this extends similarly in arbitrary dimension. The general idea is that if a fast algorithm is available to compute ortho-decompositions on two 1-D bases $(\varphi^1_{k_1})_{k_1=1}^{N_1}$, $(\varphi^2_{k_2})_{k_2=1}^{N_2}$, is extended to compute decomposition on the tensor product basis $(\varphi^1_{k_1} \otimes \varphi^2_{k_2})_{k_1,k_2}$ by apply succesively the algorithm on the "rows" and then "columns" (the order does not matters) of the matrix $(f_n)_{n=(n_1,n_2)} \in \mathbb{R}^{N_1 \times N_2}$. Indeed

$$\forall\, k = (k_1, k_2), \quad \langle f,\, \varphi^1_{k_1} \otimes \varphi^2_{k_2}\rangle = \sum_{n=(n_1,n_2)} f_n \varphi^1_{k_1}(n_1) \varphi^1_{k_1}(n_2) = \sum_{n_1} \left(\sum_{n_2} f_{n_1,n_2} \varphi^1_{k_1}(n_2) \right) \varphi^1_{k_1}(n_1).$$

Denoting $C(N_1)$ the complexity of the 1-D algorithm on $\mathbb{R}^{N_1}$, the complexity of the resulting 2-D decomposition is $N_2 C(N_1) + N_1 C(N_2)$, and hence for the FFT, it is $O(N_1 N_2 \log(N_1 N_2)) = O(N \log(N))$ for $N = N_1 N_2$.

If we represent $f \in \mathbb{R}^{N_1 \times N_2}$ as a matrix, and denote $F_N = (e^{-\frac{2\mathrm{i}\pi}{N} kn})_{k,n}$ the Fourier transform matrix (or the matrix where rows are the φ_k^*), then one can compute the 2-D Fourier transform as matrix-matrix products

$$\hat{f} = F_{N_1} \times f \times F^*_{N_2} \in \mathbb{R}^{N_1 \times N_2}.$$

But of course these multiplications are not computed explicitly (one uses the FFT).

`Associated code: coding/test_fft2.m`

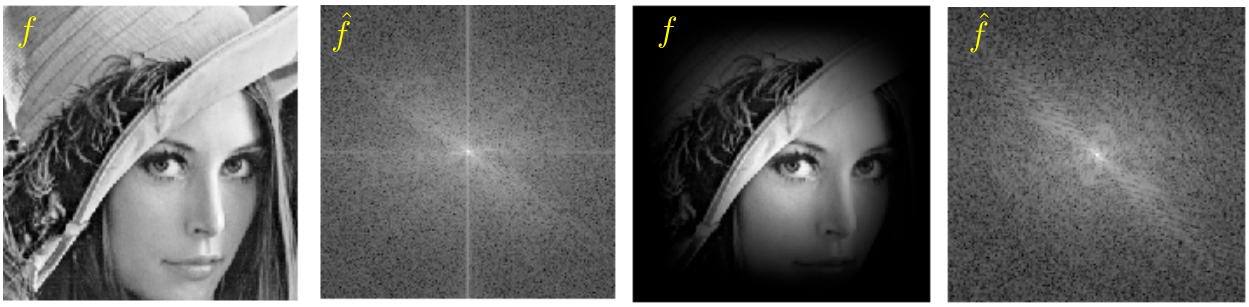


Figure 2.17: 2D Fourier analysis of a image (left), and attenuation of the periodicity artifact using masking (right).

2.5.3 Shannon sampling theorem.

The sampling Theorem 1 extends easily to $\mathbb{R}^d$ by tensorization, assuming that the sampling is on a uniform Cartesian grid. In 2-D for instance, if $\text{supp}(\hat{f}) \subset [-\pi/s_1, \pi/s_1] \times [-\pi/s_2, \pi/s_2]$ and f is decaying fast enough,

$$\forall\, x \in \mathbb{R}^2, \quad f(x) = \sum_{n \in \mathbb{Z}^2} f(n_1 s_1, n_2 s_2)\, \text{sinc}(x_1/s_1 - n_1)\, \text{sinc}(x_2/s_2 - n_2) \quad \text{where} \quad \text{sinc}(u) = \frac{\sin(\pi u)}{\pi u}.$$

2.5.4 Convolution in higher dimension.

Convolution on $\mathbb{X}^d$ with $\mathbb{X} = \mathbb{R}$ or $\mathbb{X} = \mathbb{R}/2\pi\mathbb{Z}$ are defined in the very same way as in 1-D (2.4) as

$$f \star g(x) = \int_{\mathbb{X}^d} f(t) g(x-t) \mathrm{d}t.$$

Similarly, finite discrete convolution of vectors $f \in \mathbb{R}^{N_1 \times N_2}$ extend formula (2.11) as

$$\forall\, n \in [\![0, N_1 - 1]\!] \times [\![0, N_2 - 1]\!], \quad (f \star g)_n \overset{\text{def.}}{=} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} f_k g_{n-k}$$

where additions and subtractions of vectors are performed modulo (N_1, N_2).

The Fourier-convolution theorem is still valid in all this cases, namely $\mathcal{F}(f \star g) = \hat{f} \odot \hat{g}$. In the finite case, this offers a fast $O(N \log(N))$ method to compute convolutions even if f and g do not have small support.

2.6 Application to ODEs and PDEs

2.6.1 On Continuous Domains

We here give only the intuition without formal proofs.

One $\mathbb{X} = \mathbb{R}$ or $\mathbb{T}$, one has

$$\mathcal{F}(f^{(k)})(\omega) = (\mathrm{i}\omega)^k \hat{f}(\omega).$$

Intuitively, $f^{(k)} = f \star \delta^{(k)}$ where $\delta^{(k)}$ is a distribution with Fourier transform $\hat{\delta}^{(k)}(\omega) = (\mathrm{i}\omega)^k$. Similarly on $\mathbb{X} = \mathbb{R}^d$ (see Section 2.5 for the definition of the Fourier transform in dimension d), one has

$$\mathcal{F}(\Delta f)(\omega) = -\|\omega\|^2 \hat{f}(\omega) \tag{2.15}$$

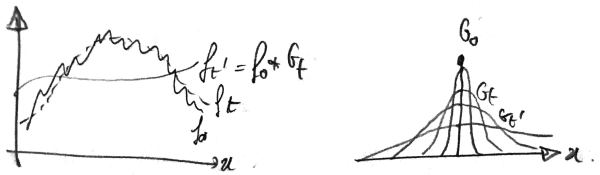


Figure 2.18: Heat diffusion as a convolution.

(and similarly on $\mathbb{T}$ replacing ω by $n \in \mathbb{Z}^d$). The Fourier transform (or Fourier coefficients) are thus powerful to study linear differential equations with constant coefficients, because they are turned into algebraic equations.

As a typical example, we consider the heat equation

$$\frac{\partial f_t}{\partial t} = \Delta f_t \quad \Longrightarrow \quad \forall\, \omega, \quad \frac{\partial \hat{f}_t(\omega)}{\partial t} = -\|\omega\|^2 \hat{f}(\omega).$$

This shows that $\hat{f}_t(\omega) = \hat{f}_0(\omega) e^{-\|\omega\|^2 t}$ and by inverse Fourier transform and the convolution theorem

$$f_t = G_t \star f_0 \quad \text{where} \quad G_t = \frac{1}{(4\pi t)^{d/2}} e^{-\frac{\|x\|^2}{4t}}$$

which is a Gaussian of standard deviation $\sqrt{2t}$.

2.6.2 Finite Domain and Discretization

On $\mathbb{R}/N\mathbb{Z}$ (i.e. discrete domains with periodic boundary conditions), one typically considers forward finite differences (first and second order)

$$D_1 f \overset{\text{def.}}{=} N(f_{n+1} - f_n)_n = f \star d_1 \quad \text{where} \quad d_1 = [1, 0, \ldots, 0, -1]^\top \in \mathbb{R}^N, \tag{2.16}$$

$$D_2 f = D_1^\top D_1 f \overset{\text{def.}}{=} N^2(f_{n+1} + f_{n-1} - 2f_n)_n = f \star d_2 \quad \text{where} \quad d_2 = d_1 \star \bar{d}_1 = [-2, 1, 0, \ldots, 0, 1]^\top \in \mathbb{R}^N. \tag{2.17}$$

Thanks to Proposition 5, one can alternatively computes

$$\mathcal{F}(D_2 f) = \hat{d}_2 \odot \hat{f} \quad \text{where} \quad (\hat{d}_2)_k = N^2(e^{\frac{2\mathrm{i}\pi}{N}} + e^{-\frac{2\mathrm{i}\pi}{N}} - 2) = -4N^2 \sin\left(\frac{\pi k}{N}\right)^2. \tag{2.18}$$

For $N \gg k$, one thus has $(\hat{d}_2)_k \sim -(2\pi k)^2$ which matches the scaling of (2.15).

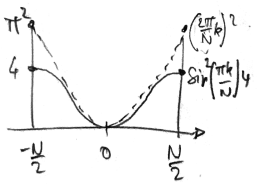


Figure 2.19: Comparison of the spectrum of Δ and D_2.

2.7 A Bit of Group Theory

The reference for this section is [20].

2.7.1 Characters

For $(G, +)$ a commutative group, a character is a group morphism $\chi : (G, +) \to (\mathbb{C}^*, \cdot)$, i.e. is satisfies

$$\forall\, (n, m) \in G, \quad \chi(n + m) = \chi(n)\chi(m).$$

The set of characters is the so-called dual $(\hat{G}, \odot)$ and is a group for the pointwise multiplication $(\chi_1 \odot \chi_2)(n) \overset{\text{def.}}{=} \chi_1(n)\chi_2(n)$. Indeed, the inverse of a character χ is $\chi^{-1}(n) = \chi(-n)$.

Note that for a finite group G with $|G| = N$, then since $N \times n = 0$ for any $n \in G$, then $\chi(n)^N = \chi(Nn) = \chi(0) = 1$, so that characters assume values in the unit circle, and more precisely

$$\chi(n) \in \left\{ e^{\frac{2\mathrm{i}\pi}{N}k} \;;\; 0 \leqslant k \leqslant N-1 \right\}. \tag{2.19}$$

So in particular $\hat{G}$ is a finite group (since there is a finite number of application between two finite sets) and $\chi^{-1} = \bar{\chi}$. In the case of a cyclic group, the dual is actually simple to describe.

Proposition 6. *For $G = \mathbb{Z}/N\mathbb{Z}$, then $\hat{G} = (\varphi_k)_{k=0}^{N-1}$ where $\varphi_k = (e^{\frac{2\mathrm{i}\pi}{N}nk})_n$ and $k \mapsto \varphi_k$ defines a (non-canonical) isomorphism $G \sim \hat{G}$.*

Proof. The φ_k are indeed characters.

Conversely, for any $\chi \in \hat{G}$, according to (2.19), $\chi(1) = e^{\frac{2\mathrm{i}\pi}{N}k}$ for some k. Then

$$\chi(n) = \chi(1)^n = {}^{\frac{2\mathrm{i}\pi}{N}kn} = \varphi_k(n).$$

Note that all these applications are different (because $\varphi_k(1)$ are all distincts) which shows that $|G| = |\hat{G}|$ so that they are isomorphic. □

This proposition thus shows that characters of cyclic groups are exactly the discrete Fourier orthonormal basis defined in (2.8).

Commutative groups. For more general commutative groups with a finite number of generators, according to the celebrated structure theorem, one can "decompose" them as a product of cyclic group (which are in some sense the basic building blocks), i.e. there is the following isomorphism of groups

$$G \sim (\mathbb{Z}/N_1\mathbb{Z}) \times \ldots \times (\mathbb{Z}/N_d\mathbb{Z}) \times \mathbb{Z}^Q. \tag{2.20}$$

If G is finite, then $Q = 0$ and $N = N_1 \times N_d$. In this case, G is simply a discrete d-dimensional "rectangle" with periodic boundary conditions.

For two finite groups (G_1, G_2) one has

$$\widehat{G_1 \times G_2} = \hat{G}_1 \otimes \hat{G}_2 = \left\{ \chi_1 \otimes \chi_2 \;;\; (\chi_1, \chi_2) \in \hat{G}_1 \times \hat{G}_2 \right\}. \tag{2.21}$$

Here $\otimes$ is the tensor product of two functions

$$\forall\, (n_1, n_2) \in G_1 \times G_2, \quad (\chi_1 \otimes \chi_2)(n_1, n_2) \overset{\text{def.}}{=} \chi_1(n_1)\chi_2(n_2).$$

Indeed, one verifies that $\chi_1 \otimes \chi_2$ is a morphism, and in fact one has the factorization $\chi = \chi(\cdot, 0) \otimes \chi(0, \cdot)$ because one decomposes $(n_1, n_2) = (n_1, 0) + (0, n_2)$.

This construction, thanks to the structure theorem, leads to a constructive proof of the isomorphism theorem.

Proposition 7. *If G is commutative and finite then $\hat{G} \sim G$.*

Proof. The structure theorem (2.20) for $Q = 0$ and the dual of a product (2.21) shows that

$$\hat{G} \sim \hat{G}_1 \otimes \ldots \otimes \hat{G}_d$$

where we denoted $G_\ell \overset{\text{def.}}{=} \mathbb{Z}/N_\ell\mathbb{Z}$. One then remark that $\hat{G}_1 \otimes \hat{G}_2 \sim \hat{G}_1 \times \hat{G}_2$. One concludes thanks to Proposition 6, since one has $\hat{G}_k \sim G_k$. □

Note that the isomorphism $\hat{G} \sim G$ is not "canonical" since it depends on the indexing of the roots of unity on the circle. Similarly to the case of duality of vector space, the isomorphism $\hat{\hat{G}} \sim G$ can be made canonical by considering the evaluation map

$$g \in G \longmapsto e_g \in \hat{\hat{G}} \quad \text{where} \quad \left(e_g : \chi \in \hat{G} \mapsto \chi(g) \in \mathbb{C}^*. \right)$$

Discrete Fourier transform from character's point of view. One can be even more constructive by remarking that characters in $\hat{G}_\ell$ are the discrete Fourier atoms (2.8), i.e. are of the form

$$(e^{\frac{2\mathrm{i}\pi}{N_\ell}k_\ell n_\ell})_{n_\ell=0}^{N_\ell-1} \quad \text{for some} \quad 0 \leqslant k_\ell < N_\ell.$$

Identifying G and $G_1 \times \ldots \times G_d$, by tensorizing these functions together, one thus obtains that the characters composing $\hat{G}$ are exactly the orthogonal multi-dimensional discrete Fourier basis (2.13).

2.7.2 More General cases

Infinite groups. For an infinite group with a finite number of generator, one has $Q > 0$, and the definition of $\hat{G}$ should impose the continuity of the characters (and also use an invariant measure on G to define inner products). In the case $G = \mathbb{Z}$, the dual are indexed by a continuous parameter,

$$\hat{\mathbb{Z}} = \left\{\varphi_\omega : n \mapsto e^{\mathrm{i}n\omega} \in \mathbb{C}^{\mathbb{Z}} \;;\; \omega \in \mathbb{R}/2\pi\mathbb{Z}\right\}$$

so that $\hat{\mathbb{Z}} \sim \mathbb{R}/2\pi\mathbb{Z}$. The case $G = \mathbb{Z}^Q$ follows by tensorization. The $(\varphi_\omega)_\omega$ are "orthogonal" in the sense that $\langle \varphi_\omega, \varphi_{\omega'} \rangle_{\mathbb{Z}} = \delta(\omega - \omega')$ can be understood as a Dirac kernel (this is similar to the Poisson formula), where $\langle u, v \rangle_{\mathbb{Z}} \overset{\text{def.}}{=} \sum_n u_n \bar{v}_n$. The "decomposition" of a sequence $(c_n)_{n\in\mathbb{Z}}$ on the set of characters is equivalent to forming a Fourier series $\sum_n c_n e^{-\mathrm{i}n\omega}$.

Similarly, for $G = \mathbb{R}/2\pi\mathbb{Z}$, one has $\hat{G} = \mathbb{Z}$, with orthonormal characters $\varphi_n = e^{\mathrm{i}\cdot n}$, so that the decomposition of functions in $L^2(G)$ is the computation of Fourier coefficients. Intuitively, an Hilbertian theory is associated to a compact group, which comes with an invariant measure.

Non-commutative groups. For non-commutative group, one observes that G is not isometric to $\hat{G}$. A typical example is the symmetric group Σ_N of N elements, where one can show that $\hat{G} = \{\mathrm{Id}, \varepsilon\}$ where $\varepsilon(\sigma) = (-1)^q$ is the signature, where q is the number of permutations involved in a decomposition of $\sigma \in \Sigma_N$.

In order to study non-commutative groups, one has to replace morphisms $\chi : G \to \mathbb{C}^*$ by morphisms $\rho : G \to \mathrm{GL}(\mathbb{C}^{n_\rho})$ for some n_ρ, which are called "representations" of the group G. For $(g, g') \in G$ (denoting now multiplicatively the operation on G), one should thus have $\rho(gg') = \rho(g) \circ \rho(g')$. When $n_\rho = 1$, identifying $\mathrm{GL}(\mathbb{C}) \sim \mathbb{C}^*$, one retrieves the definition of characters. Note that if ρ is a representation, then $\chi(g) \overset{\text{def.}}{=} \mathrm{tr}(\rho(g))$, where tr is the trace, defines a character.

If there is a subspace V stable by all the $\rho(g)$, one can build W such that $\mathbb{R}^{n_\rho} = V + W$ which is also stable, thus reducing the study of ρ to the study on a small space (matrices have a block diagonal form). It suffice to consider the inner product

$$\langle x, y \rangle \overset{\text{def.}}{=} \sum_{g\in G} \langle \rho(g)x, \rho(g)y \rangle_{\mathbb{R}^{n_\rho}}$$

and select the orthogonal $V^\perp$ for this product. Note that when using an ortho-basis of $\mathbb{R}^{n_\rho}$ for this inner product, the matrices associated to the $\rho(g)$ are unitary. In order to limit the set of such representations, one is only interested in "elementary" ones, which does not have invariant sub-spaces, and are called "irreducible" (otherwise one could create arbitrary large representation by stacking others in a block diagonal way). One also only consider these irreductible representation up to isomorphism, where two representation (ρ, ρ') are said to be isomorphic if $\rho(g) = U^{-1}\rho(g)U$ where $U \in \mathrm{GL}(\mathbb{C}^d)$ is a change of basis. The set of all these representations up to isomomorphisms is called the dual group and denoted $\hat{G}$.

For the symmetric group, there is an explicit description of the set of irreducible representations. For instance, for $G = \Sigma_3$, $|G| = 6$, and there is two representation of dimension 1 (the identity and the signature, which are the caracters) and one representation of dimension 2, which is obtained by identifying G with the isometric of the equilateral triangle in $\mathbb{R}^2$, and the dimensions indeed satisfy $6 = |G| = \sum n_\rho^2 = 1 + 1 + 2^2$.

One can show that the dimensions n_ρ of these irreducible representations $\rho \in \hat{G}$ satisfies $\sum_{\rho\in\hat{G}} n_\rho^2 = N$ and that the entries of the matrices involved in these representation define an orthogonal basis of the space

of functions $f : G \to \mathbb{C}$ (note however that this set of basis function is not canonical since it depends on a particular choice of basis for each representation up to isomorphism). The associated Fourier transform of a function $f : G \to \mathcal{C}$ is defined as

$$\forall\, \rho \in \hat{G}, \quad \hat{f}(\rho) \stackrel{\text{def.}}{=} \sum_{g \in G} f(g)\rho(g) \in \mathbb{C}^{n_\rho \times n_\rho}.$$

This corresponds to computing inner products with the aforementioned ortho-basis. It is an invertible linear transform, whose invert is given next.

Proposition 8. *One has*

$$\forall\, g \in G, \quad f(g) = \frac{1}{|G|} \sum_{\rho \in \hat{G}} n_\rho \operatorname{tr}(\hat{f}(\rho)\rho(g^{-1})).$$

Proof. The proof relies on the following formula, which states that for any $g \in G$

$$A_g \stackrel{\text{def.}}{=} \sum_{\rho \in \hat{G}} n_\rho \operatorname{tr}(\rho(g)) = \delta_g$$

where $\delta_g = 0$ for $g \neq \mathrm{Id}_G$ and $\delta_{\mathrm{Id}_G} = 1$. We will not prove this formula, and refer to the book of Diaconis p.13 for a proof, which is based on the decomposition of the character of the so-called standard representation (which corresponds to permuting by the action of G the canonical basis of $\mathbb{R}^{|G|}$) on the set of characters, which are orthogonal. One then has

$$\begin{aligned}\frac{1}{|G|} \sum_{\rho \in \hat{G}} n_\rho \operatorname{tr}(\hat{f}(\rho)\rho(g^{-1})) &= \frac{1}{|G|} \sum_{\rho \in \hat{G}} n_\rho \operatorname{tr}(\sum_u f(u)\rho(u)\rho(g^{-1})) \\ &= \sum_u f(u) \frac{1}{|G|} \sum_{\rho \in \hat{G}} n_\rho \operatorname{tr}(\rho(ug^{-1})) = \sum_u f(u) G_{ug^{-1}} = f(g).\end{aligned}$$

□

One can define the convolution of two functions $f, h : G \to \mathbb{C}$ as

$$(f \star h)(a) \stackrel{\text{def.}}{=} \sum_{bc=a} f(b)h(c)$$

(beware that it is not commutative anymore). The Fourier transform diagonalize these convolution operators, has stated next.

Proposition 9. *Denoting* $\mathcal{F}(f) = \hat{f}$

$$\mathcal{F}(f \star h)(\rho) = \hat{f}(\rho) \times \hat{h}(\rho)$$

where $\times$ *is the matrix multiplication.*

Proof.

$$\mathcal{F}(f \star h)(\rho) = \sum_x \sum_y f(y)h(y^{-1}x)\rho(yy^{-1}x) = \sum_x \sum_y f(y)\rho(y)h(y^{-1}x)\rho(y^{-1}x) = \sum_y f(y)\rho(y) \sum_z h(z)\rho(z)$$

where we made the change of variable $z = y^{-1}x$. □

For certain groups, there exists fast Fourier transforms to compute $\hat{f}$, an example being the permutation group, but the structure of these algorithm is much more involved than in the case $G = \mathbb{Z}/N\mathbb{Z}$.

This theory extends to compact group by considering a discrete but infinite set of representations. A typical example is to analyze signals defined on the rotation groups SO(3), on which one can compute explicitly the representation using the basis of sphereical harmonics detailed in Section 2.8.2 below. In this case, one has a representation of dimension $2\ell + 1$ for each frequency index ℓ.

2.8 A Bit of Spectral Theory

In order to define Fourier methods on general domains $\mathbb{X}$, one can use the aforementioned group-theoretic approach if $\mathbb{X} = G$ is a group, or also if a group acts transitively on $\mathbb{X}$. An alternative way is to describe the equivalent of Fourier basis functions as diagonalizing a specific differential operator (as we have seen in Section 2.6 that it is in some sense a way to characterise the Fourier basis). Of particular interest is the Laplacian, since it is the lowest order rotation-invariant differential operator, and that there exists natural generalization on domains such as surfaces or graphs.

2.8.1 On a Surface or a Manifold

The presentation here is very informal. One can define the Laplacian of a smooth function $f : \mathbb{X} \to \mathbb{C}$ defined on a "surface" $\mathbb{X}$ as

$$\forall\, x \in \mathbb{X}, \quad (\Delta f)(x) \overset{\text{def.}}{=} \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon(x))} \int_{B_\varepsilon(x)} f(x) \mathrm{d}\mu(x) - f(x).$$

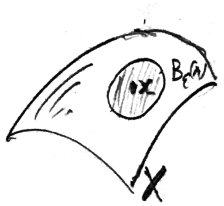


Figure 2.20: Computing Laplacian on a surface

Here $\mu(x)$ is the area measure on $\mathbb{X}$, $\mathrm{Vol}(B) \overset{\text{def.}}{=} \int_B \mathrm{d}\mu(x)$, and $B_\varepsilon(x) = \{y \,;\, d_{\mathbb{X}}(x,y) \leqslant \varepsilon\}$ is the geodesic ball of radius ε at x, where $d_{\mathbb{X}}$ is the geodesic distance on $\mathbb{X}$ (length of the shortest path).

If the surface $\mathbb{X}$ is smooth, compact and connected, then it is possible to show that Δ is itself a compact operator with a negative spectrum $0 > \lambda_1 > \lambda_2 > \ldots$ and an orthogonal set of eigenvectors $(\varphi_n)_{n \geqslant 0}$ where $\varphi_1 = 1$. Here the inner product is $\langle f, g \rangle_{\mathbb{X}} \overset{\text{def.}}{=} \int_{\mathbb{X}} f(x) g(x) \mathrm{d}\mu(x)$ on $L^2(\mathbb{X})$. In the case of a flat torus $\mathbb{X} = (\mathbb{R}/\mathbb{Z})^d$, then writing $x = (x_1, \ldots, x_d)$,

$$\Delta f = \sum_{s=1}^{d} \frac{\partial^2 f}{\partial^2 x_s}.$$

Similarly to (2.15) (which was for an unbounded domain), then one can chose for this eigen-functions φ_n the Fourier basis (2.1) and $\lambda_n = -\|n\|^2$

2.8.2 Spherical Harmonics

Of particular interest is the special case of the previous construction on the $(d-1)$-dimensional sphere $\mathbf{S}^{d-1} = \big\{x \in \mathbb{R}^d \,;\, \|x\|_{\mathbb{R}^d} = 1\big\}$. In this case, there exists a closed form expression for the eigenvectors of the Laplacian. In the 3-D case $d = 3$, they are indexed by $n = (\ell, m)$

$$\forall\, \ell \in \mathbb{N}, \quad \forall\, m = -\ell, \ldots, \ell, \quad \varphi_{\ell,m}(\theta, \varphi) = e^{\mathrm{i}m\varphi} P_\ell^m(\cos(\theta))$$

and then the eigenvalue of the Laplacian is $\lambda_{\ell,m} = -\ell(\ell+1)$. Here P_ℓ^m are associated Legendre polynomials, and we used spherical coordinates $x = (\cos(\varphi), \sin(\varphi)\sin(\theta), \sin(\varphi)\cos(\theta)) \in \mathbf{S}^3$ for $(\theta, \varphi) \in [0, \pi] \times [0, 2\pi]$. The index ℓ is analogous to the amplitude of Fourier frequencies in 2-D. For a fixed ℓ, the space $V_\ell = \mathrm{span}(\varphi_{\ell,m})$ is an eigenspace of Δ, and is also invariant under rotation.

Figure 2.21: Spherical coordinates.

2.8.3 On a Graph

We assume $\mathbb{X}$ is a graph of N vertices, simply indexed $\{1, \ldots, N\}$. Its "geometry" is described by a connectivity matrix of weights $W = (w_{i,j})_{i \sim j}$ where we denote $i \sim j$ to indicate that (i,j) is an edge of the graph for $(i,j) \in \mathbb{X}^2$. We assume that this weight matrix and the connectity is symmetric, $w_{i,j} = w_{j,i}$.

The graph Laplacian $\Delta : \mathbb{R}^N \to \mathbb{R}^N$ is computing the difference between the average of values around a point and the value at this point

$$\forall\, f \in \mathbb{R}^N, \quad (\Delta f)_i \overset{\text{def.}}{=} \sum_{j\sim i} w_{i,j} f_j - (\sum_{j\sim i} w_{i,j}) f_i \quad \Longrightarrow \quad \Delta = W - D$$

where $D \overset{\text{def.}}{=} \mathrm{diag}_i(\sum_{j\sim i} w_{i,j})$. In particular, note $\Delta \mathbb{1} = 0$

For instance, if $\mathbb{X} = \mathbb{Z}/N\mathbb{Z}$ with the graph $i \sim i-1$ and $i \sim i+1$ (modulo N), then Δ is the finite difference Laplacian operator $\Delta = D_2$ defined in (2.17). This extends to any dimension by tensorization.

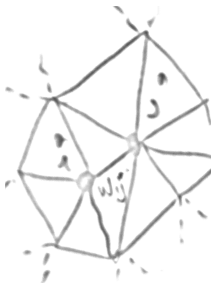


Figure 2.22: Weighted graph.

Proposition 10. *Denoting $G : f \in \mathbb{R}^N \mapsto (\sqrt{w_{i,j}}(f_i - f_j))_{i<j}$ the graph-gradient operator, one verifies that*

$$-\Delta = G^\top G \quad \Longrightarrow \quad \forall\, f \in \mathbb{R}^N, \quad \langle \Delta f,\, f\rangle_{\mathbb{R}^N} = -\langle Gf,\, Gf\rangle_{\mathbb{R}^P}.$$

where P is the number of (ordered) edges $E = \{(i,j)\ ;\ i \sim j, i < j\}$.

Proof. One has

$$\begin{aligned}
\|Gf\|^2 &= \sum_{(i,j)\in E} w_{i,j} |f_i - f_j|^2 = \sum_{i<j} w_{i,j} f_i^2 + \sum_{i<j} w_{i,j} f_j^2 - 2\sum_{i<j} w_{i,j} f_i f_j \\
&= \sum_{i<j} w_{i,j} f_i^2 + \sum_{i>j} w_{i,j} f_i^2 - \sum_{i,j} w_{i,j} f_i f_j = \sum_j f_i^2 \sum_{i,j} w_{i,j} - \sum_i f_i \sum_j w_{i,j} f_j \\
&= \langle Df,\, f\rangle - \langle Lf,\, f\rangle = -\langle Lf,\, f\rangle.
\end{aligned}$$

□

This proposition shows that Δ is a negative semi-definite operator, which thus diagonalizes in an orthobasis $(\varphi_n)_{n=1}^N$, with $\varphi_1 = 1$, with eigenvalues $0 \geqslant \lambda_1 \geqslant \lambda_N$. If $\mathbb{X}$ is connected, one can show that $\lambda_1 < 0$. In the case of a regular graph associated to a uniform grid, one retrieves the discrete Fourier basis (2.8).

More details and application of Laplacians on graphs can be found in Chapter **??**, see in particular Section **??**.

2.8.4 Other things

Multiplication of polynomials (`Code text_polymult.m`).

Fourier transform for finite-field valued functions.

Chapter 3

Wavelets

The reference for this chapter is [17].

3.1 Multi-resolution Approximation Spaces

A multiresolution approximation of $L^2(\mathbb{R})$ is a set of nested closed subspaces $(V_j)_j$

$$L^2(\mathbb{R}) \supset \ldots \supset V^{j-1} \supset V_j \supset V_{j+1} \supset \ldots \supset \{0\} \tag{3.1}$$

which must be related one from each other by dyadic scaling and must also be stable by dyadic translation

$$f \in V_j \iff f(\cdot/2) \in V_{j+1} \quad \text{and} \quad f \in V_j \iff \forall\, n \in \mathbb{Z},\ f(\cdot + n2^j) \in V_j$$

So large j corresponds to coarse approximation spaces, and 2^j is often call the "scale".

The limit on the left of (3.1) means that $\cup_j V_j$ is dense in $L^2(\mathbb{R})$, or equivalently that $P_{V_j}(f) \to f$ as $j \to -\infty$ where P_V is the orthogonal projector on V

$$P_V(f) = \operatorname*{argmin}_{f' \in V} \|f - f'\|.$$

The limit on the right of (3.1) means that $\cap_j V_j = \{0\}$, or equivalently that $P_{V_j}(f) \to 0$ as $j \to +\infty$.

The first example is piecewise constant functions on dyadic intervals

$$V_j = \left\{ f \in L^2(\mathbb{R}) \ ;\ \forall n, f \text{ is constant on } [2^j n, 2^j(n+1)[\right\}, \tag{3.2}$$

A second example is the space used for Shannon interpolation of bandlimitted signals

$$V_j = \left\{ f \ ;\ \mathrm{Supp}(\hat{f}) \subset [-2^{-j}\pi, 2^{-j}\pi] \right\} \tag{3.3}$$

which corresponds to function which can be exactly represented using samples $f(2^j n)$ (to be compared with piecewise constant signal on a grid with spacing 2^j). In this case, the orthogonal projection is the bandlimitting operator

$$P_{V_j}(f) = \mathcal{F}^{-1}(\hat{f} \odot 1_{[-2^{-j}\pi, 2^{-j}\pi]}).$$

Scaling functions. We also require that there exists a scaling function $\varphi \in L^2(\mathbb{R})$ so that

$$\{\varphi(\cdot - n)\}_n \text{ is an Hilertian orthonormal basis of } V_0.$$

By the dilation property, this implies that

$$\{\varphi_{j,n}\}_n \text{ is an Hilertian orthonormal basis of } V_j \quad \text{where} \quad \varphi_{j,n} \stackrel{\text{def.}}{=} \frac{1}{2^{j/2}} \varphi\left(\frac{\cdot - 2^j n}{2^j} \right).$$

The normalization is such to ensure $\|\varphi_{j,n}\| = 1$.

Note that one then has

$$P_{V_j}(f) = \sum_n \langle f, \varphi_{j,n}\rangle \varphi_{j,n}.$$

Figure 3.1 illustrates the translation and scaling effect.

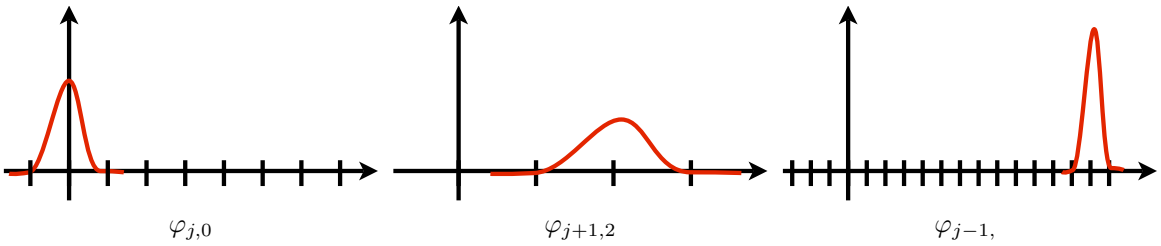


Figure 3.1: Translation and scaling to generate approximation spaces.

For the case of piecewise constant signals (3.2), one can use

$$\varphi = 1_{[0,1[} \quad \text{and} \quad \varphi_{j,n} = 2^{-j/2} 1_{[2^j n, 2^j(n+1)[}.$$

For the case of Shannon multiresolution (3.3), one can use $\varphi(t) = \sin(\pi t)/(\pi t)$ and one verifies

$$\langle f, \varphi_{j,n}\rangle = f(2^j n) \quad \text{and} \quad f = \sum_n f(2^j n)\varphi_{j,n}.$$

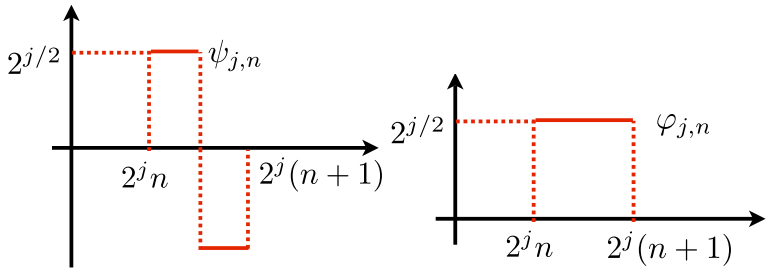


Figure 3.2: **BUG: the elevation should be $2^{-j/2}$ and not $2^{j/2}$** Haar scaling (left) and wavelet (right) functions.

Spectral orthogonalization. In many case of practical interest, the space V_j is describe by a translation-invariant basis which is not-orthogonal, $V_j = \mathrm{Span}(\theta(\cdot - n))_{n\in\mathbb{Z}}$. The following proposition shows how to orthogonalize it using the Fourier transform. Figure 3.3 shows how it leads to cardinal spline orthogonal functions.

Proposition 11. *For $\theta \in L^2(\mathbb{R})$ (assumed regular and with fast enough decay), $\{\theta(\cdot - n)\}_{n\in\mathbb{Z}}$ is orthonormal if and only if*

$$\forall \omega, \quad A(\omega) \stackrel{\text{def.}}{=} \sum_k |\hat\theta(\omega - 2\pi k)|^2 = 1.$$

If there exists $0 < a \leqslant b < +\infty$ such that $a \leqslant A(\omega) \leqslant b$, then φ defined by

$$\hat\varphi(\omega) = \frac{\hat\theta(\omega)}{\sqrt{A(\omega)}}$$

is such that $\{\varphi(\cdot - n)\}_{n\in\mathbb{Z}}$ is an Hilbertian basis of $\mathrm{Span}\{\theta(\cdot - n)\}_{n\in\mathbb{Z}}$.

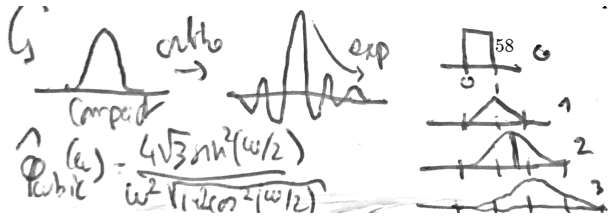


Figure 3.3: Orthogonalization of b-spline to defines cardinal orthogonal spline functions.

Proof. One has that $\{\theta(\cdot - n)\}_{n\in\mathbb{Z}}$ is orthonormal if and only if

$$\langle \theta, \theta(\cdot - n)\rangle = (\theta \star \bar{\theta})(n) = \delta_{0,n} \overset{\text{def.}}{=} \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{otherwise.} \end{cases}$$

where $\bar{\theta} = \theta(-\cdot)$ and δ_0 is the discrete Dirac vector. Recall the Poisson summation formula (1.8) for any function h

$$\sum_n h(\omega - 2n\pi) = \sum_n h(n)e^{-\mathrm{i}n\omega}$$

which here reads

$$\sum_n \mathcal{F}(\theta \star \bar{\theta})(\omega - 2\pi n) = \sum_n \delta_{0,n} e^{-\mathrm{i}n\omega} = 1.$$

We conclude with the Fourier-convolution formula (2.6) shows that $\mathcal{F}(\theta\star\bar{\theta})(\omega) = \hat{\theta}(\omega)\hat{\theta}(\omega)^* = |\hat{\theta}(\omega)|^2$ which leads to the desired formula, and it is if and only if. Normalizing by $1/\sqrt{A(\omega)}$, which is a bounded function, shows that $\hat{\varphi}$ satsifies $\sum_k |\hat{\varphi}(\omega - 2\pi k)|^2 = 1$. □

A typical example of application is spline (e.g. cubic ones) interpolations, which are generated by the box-spline function θ which is a piecewise polynomial with a compact support.

3.2 Multi-resolution Details Spaces

The details spaces are defined as orthogonal complement of $V_j \subset V_{j-1}$, which is legit because these are closed subspaces

$$\forall\, j, \quad W_j \text{ is such that } V_{j-1} = V_j \oplus^{\perp} W_j.$$

This leads to the following sequence of embedded spaces

$$\mathrm{L}^2(\mathbb{R}) \longrightarrow \cdots \searrow V_{j-1} \longrightarrow V_j \searrow V_{j+1} \longrightarrow \cdots \searrow \longrightarrow \{0\}$$
$$W_{j-1} \qquad W_j \qquad W_{j+1}$$

Once again, we suppose that W_0 has an Hilbertian ortho-basis of the form $\{\psi(\cdot - n)\}_n$, so that

$$\{\psi_{j,n}\}_n \text{ is an Hilertian orthonormal basis of } W_j \quad \text{where} \quad \psi_{j,n} \overset{\text{def.}}{=} \frac{1}{2^{j/2}}\psi\left(\frac{\cdot - 2^j n}{2^j}\right).$$

Due to the orthogonal complementarity property, one has

$$L^2(\mathbb{R}) = \bigoplus_{j=-\infty}^{j=+\infty} W_j = V_{j_0} \bigoplus_{j\leqslant j_0}^{j} V_j.$$

This means that for all $f \in L^2(\mathbb{R})$, one has the following convergence in $L^2(\mathbb{R})$,

$$f = \lim_{(j_-,j_+)\to(-\infty,+\infty)} \sum_{j=j_-}^{j_+} P_{W_j} f = \lim_{j_+\to+\infty} P_{V_{j_0}} f + \sum_{j=j_0}^{j_+} P_{W_j} f.$$

This decomposition shows that

$$\left\{\psi_{j,n} \;;\; (j,n) \in \mathbb{Z}^2\right\}$$

is an Hilbertian orthogonal basis of $L^2(\mathbb{R})$, which is called a wavelet basis. One also have a "truncated" ortho-basis

$$\{\psi_{j,n} \;;\; j \leqslant j_0, n \in \mathbb{Z}\} \cup \{\varphi_{j_0,n} \;;\; n \in \mathbb{Z}\}.$$

A (forward) Wavelet transform corresponds to the computation of all the inner products of some function f with the elements of these basis.

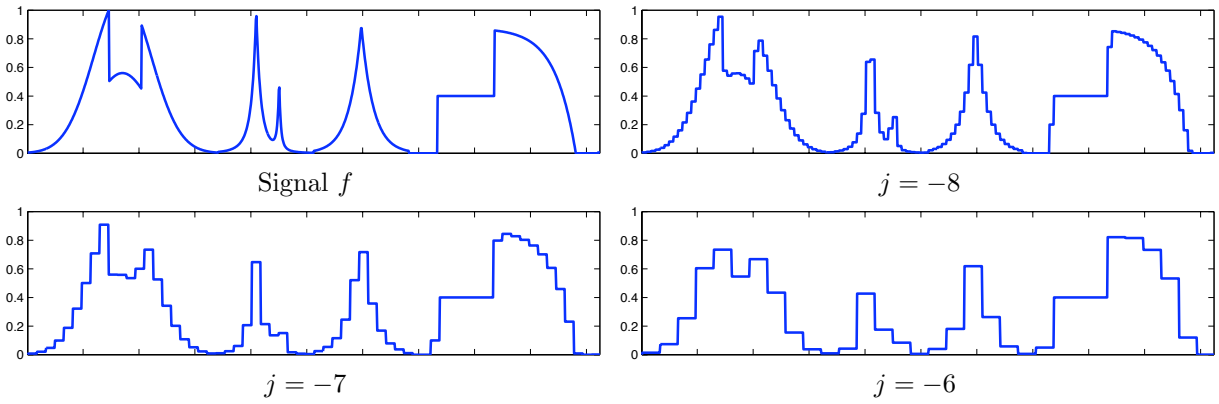


Figure 3.4: 1-D Haar multiresolution projection $P_{V_j} f$ of a function f.

Haar wavelets. For the Haar multiresolution (3.2), one has

$$W_j = \left\{ f \;;\; \forall n \in \mathbb{Z},\; f \text{ constant on } [2^{j+1}n, 2^{j+1}(n+1)) \quad \text{and} \quad \int_{n2^j}^{(n+1)2^j} f = 0 \right\}. \tag{3.4}$$

A possible choice for a mother wavelet function is

$$\psi(t) = \frac{1}{\sqrt{2}} \begin{cases} 1 & \text{for} \quad 0 \leqslant t < 1/2, \\ -1 & \text{for} \quad 1/2 \leqslant t < 1, \\ 0 & \text{otherwise,} \end{cases}$$

as shown on Figure 3.2, right.

Figure 3.5 shows examples of projections on details spaces, and how they can be derived from projection on approximation spaces.

Shannon and splines. Figure 3.6 shows that Shannon and splines corresponds to a hard and a soft segmentation of the frequency domain (Shannon being in some sense the high degree limit of splines).

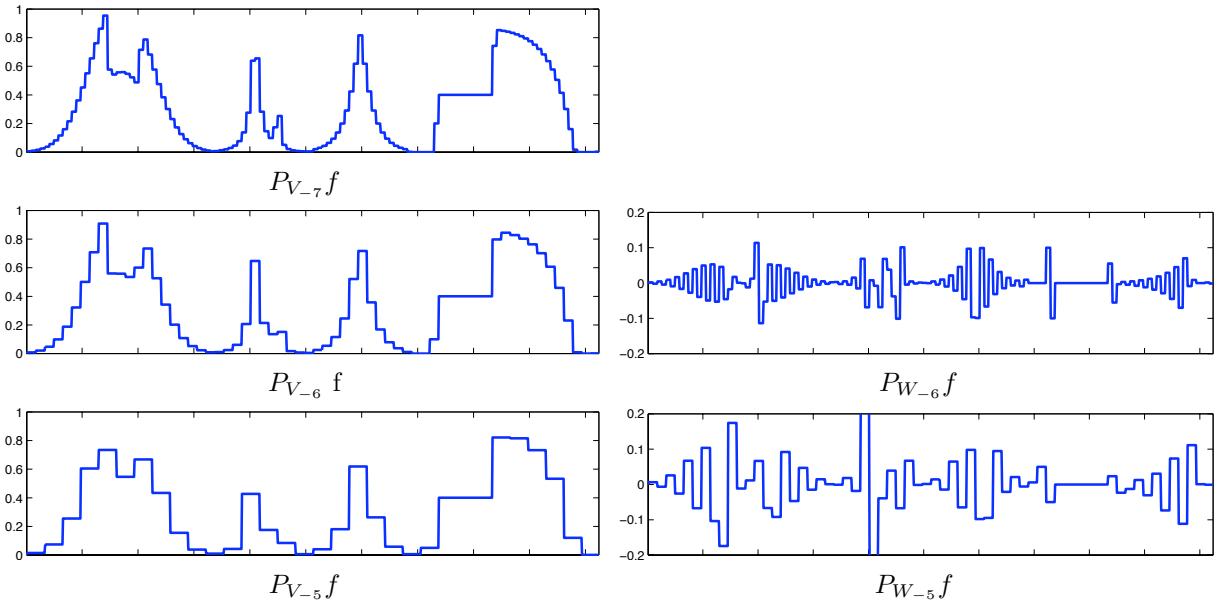


Figure 3.5: Projection on Haar approximation spaces (left) and detail spaces (right).

3.3 On Bounded Domains

On a periodic bounded domain $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ (note that we use here 1-periodicity, in contrast to the convention we used for Fourier series of 2π-periodicity), one obtains an orthogonal wavelet basis of $L^2(\mathbb{T})$ by periodizing the original wavelets, and also restricting the translation points $2^j n$ to be in $[0,1]$, i.e. $0 \leqslant n < 2^{-j}$. Similarly to (1.6), the periodization of a function $f \in L^1(\mathbb{R})$ is the function

$$f^P = \sum_{n \in \mathbb{Z}} f(\cdot - n) \in L^1(\mathbb{T}).$$

The wavelet basis is thus define as

$$\left\{ \psi^P_{j,n} \;;\; j \leqslant j_0, 0 \leqslant n < 2^{-j} \right\} \cup \left\{ \varphi^P_{j_0,n} \;;\; 0 \leqslant n < 2^{-j_0} \right\}.$$

and one verifies that it defines an Hilbertian ortho-basis of $L^2(\mathbb{T})$, see Figure 3.7. It is possible to define wavelet basis using Neumann (mirror) boundary conditions, but this is more involved.

3.4 Fast Wavelet Transform

3.4.1 Discretization

We now work over $\mathbb{R}/\mathbb{Z}$. The modeling hypothesis is that one has access to a discrete signal $a_J \in \mathbb{R}^N$ with $N = 2^{-J}$ at some fixed scale 2^J, and that this signal exactly matches the inner-product with the scaling functions, i.e.

$$\forall\, n, \in \{0, \ldots, N-1\}, \quad a_{J,n} = \langle f, \varphi^P_{J,n} \rangle \approx f(2^J n), \tag{3.5}$$

for some function of interest f we are sampling. This is equivalent to saying that the discretization process have exactly access to $P_{V_J} f$. This hypothesis is questionnable, and similar to the Shannon bandlimit assumption. In practice, the scaling functions $(\varphi_{J,n})_n$ are often quite close to the point-spread function of the acquisition device, so it is acceptable. One can however improves this by correcting the acquired values by the device to be closer to asumption (3.5).

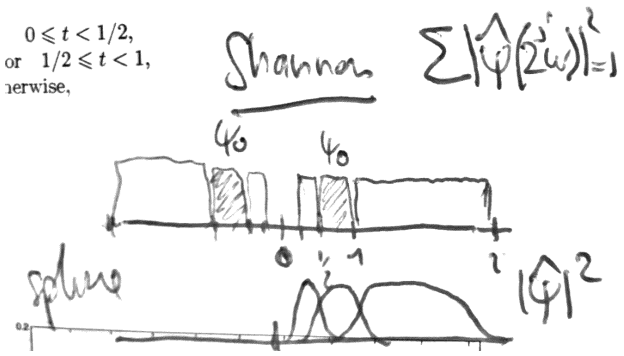


Figure 3.6: Spline and Shannon wavelet segmentation of the frequency axis.

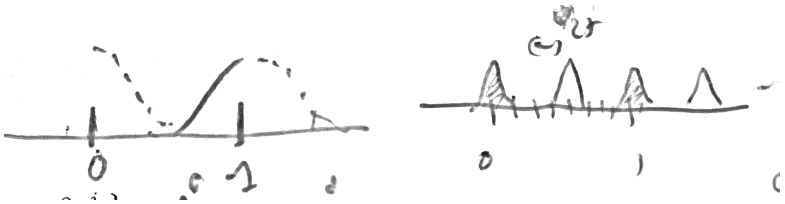


Figure 3.7: Periodizing basis functions. This leads to considering only $0 \leqslant n2^{-j}$, so that in practice one choose $j_0 = 0$.

The discrete wavelet transform then computes, from this input a_J, all the coefficients

$$\forall\, j \in \{J+1, J+2, \ldots, 0\}, \quad \forall\, n \in [\![0, 2^{-j}-1]\!], \quad a_{j,n} \stackrel{\text{def.}}{=} \langle f, \varphi^P_{j,n}\rangle, \quad \text{and} \quad d_{j,n} \stackrel{\text{def.}}{=} \langle f, \psi^P_{j,n}\rangle$$

in this order (increasing values of j). During the algorithm, the previously computed vector a_j can be discarded, and only the d_j are kept.

The forward discrete wavelet transform on a bounded domain is thus the orthogonal finite dimensional map

$$a_J \in \mathbb{R}^N \longmapsto \big\{d_{j,n}\ ;\ 0 \leqslant j < J, 0 \leqslant n < 2^{-j}\big\} \cup \{a_0 \in \mathbb{R}\}.$$

The inverse transform, which is thus the adjoint due to orthogonality, is the inverse map.

Figure 3.8 shows examples of wavelet coefficients. For each scale 2^j, there are 2^{-j} coefficients.

3.4.2 Forward Fast Wavelet Transform (FWT)

The algorithm proceeds by computing a series of simple operators

$$\forall\, j = J+1, \ldots, 0, \quad (a_j, d_j) = \mathcal{W}_j(a_{j-1}) \quad \text{where} \quad \mathcal{W}_j : \mathbb{R}^{2^{-j+1}} \to \mathbb{R}^{2^{-j}} \times \mathbb{R}^{2^{-j}} \tag{3.6}$$

The number of such steps is thus $|J| = \log_2(N)$. Each $\mathcal{W}_j$ is orthogonal since it corresponds to linear maps between coefficients in orthogonal bases.

In order to describe the algorithm that computes $\mathcal{W}_j$, we introduce the filters "filter" coefficients $f, g \in \mathbb{R}^{\mathbb{Z}}$

$$h_n \stackrel{\text{def.}}{=} \frac{1}{\sqrt{2}}\langle \varphi(\cdot/2), \varphi(\cdot - n)\rangle \quad \text{and} \quad g_n \stackrel{\text{def.}}{=} \frac{1}{\sqrt{2}}\langle \psi(\cdot/2), \varphi(\cdot - n)\rangle. \tag{3.7}$$

Figure 3.9 illustrates the computation of these weights for the Haar system.

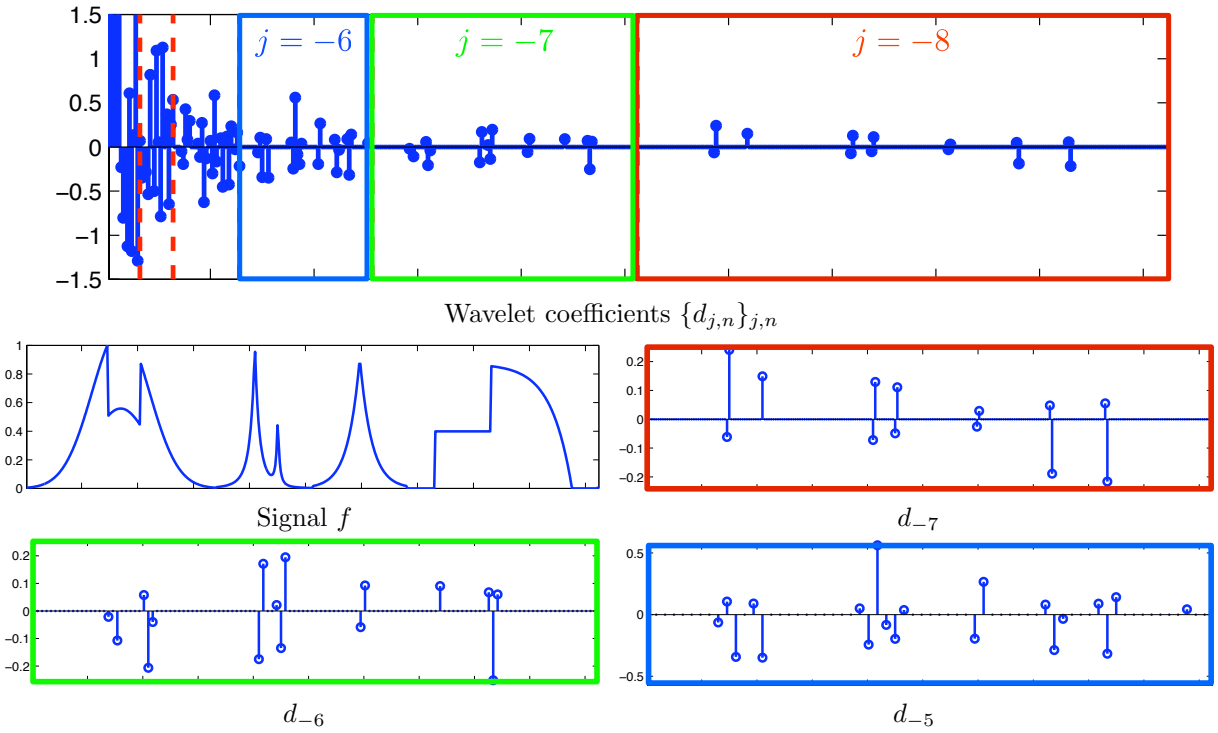


Figure 3.8: Wavelet coefficients. Top row: all the coefficients. Bottoms rows: zoom on the different scales

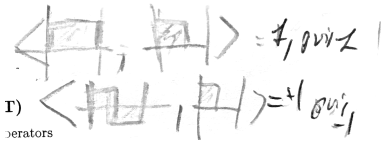


Figure 3.9: Haar wavelets weights as inner products.

We denote as $\downarrow_2 : \mathbb{R}^K \to \mathbb{R}^{K/2}$ the subsampling operator by a factor of 2, i.e.

$$u \downarrow_2 \stackrel{\text{def.}}{=} (u_0, u_2, u_4, \ldots, u_{K-2}, u_K).$$

In the following, we assume that these filters are decaying fast enough.

Proposition 12. *One has*

$$\mathcal{W}_j(a_j) = ((\bar{h} \star a_{j-1}) \downarrow_2, (\bar{g} \star a_{j-1}) \downarrow_2), \tag{3.8}$$

where $\star$ denotes periodic convolutions on $\mathbb{R}^{-j+1}$.

Proof. We note that $\varphi(\cdot/2)$ and $\psi(\cdot/2)$ are in $\mathcal{W}_j$, so one has the decompositions

$$\frac{1}{\sqrt{2}}\varphi(t/2) = \sum_n h_n \varphi(t-n) \quad \text{and} \quad \frac{1}{\sqrt{2}}\psi(t/2) = \sum_n g_n \varphi(t-n) \tag{3.9}$$

Doing the change of variable $t \mapsto \frac{t-2^j p}{2^{j-1}}$ in (3.9), one obtains

$$\frac{1}{\sqrt{2}}\varphi\left(\frac{t-2^j p}{2^j}\right) = \sum_n h_n \varphi\left(\frac{t}{2^{j-1}} - (n+2p)\right)$$

(similarly for ψ) and then doing the change $n \mapsto n-2p$, one obtains

$$\varphi_{j,p} = \sum_{n\in\mathbb{Z}} h_{n-2p}\varphi_{j-1,n} \quad \text{and} \quad \psi_{j,p} = \sum_{n\in\mathbb{Z}} g_{n-2p}\psi_{j-1,n}.$$

When working with periodized function $(\varphi^P_{j,n}, \psi^P_{j,p})$, this formula is still valid, but the summation over $n \in \mathbb{Z}$ should be done modulo 2^{-j+1}. Taking inner product of both size with respect to f (which is legit if h, g are decaying fast enough), one obtains the fundamental recursion fromula

$$a_{j,p} = \sum_{n\in\mathbb{Z}} h_{n-2p}a_{j-1,n} = (\bar{h} \star a_{j-1})_{2p} \quad \text{and} \quad d_{j,p} = \sum_{n\in\mathbb{Z}} g_{n-2p}a_{j-1,n} = (\bar{g} \star a_{j-1})_{2p} \tag{3.10}$$

where $\bar{u}_n \overset{\text{def.}}{=} u_{-n}$. One can show that this formula is still valid when working over a bounded interval $\mathbb{T} = \mathbb{R}/\mathbb{Z}$, but then $\star$ denotes the periodic convolution over $\mathbb{Z}/2^{-j+1}\mathbb{Z}$. □

Figure 3.10 shows two steps of application of these refinement relationships.

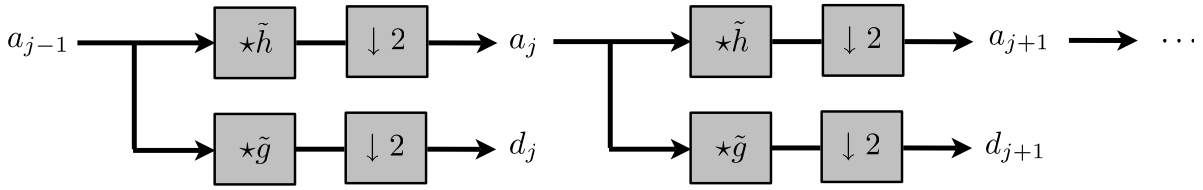


Figure 3.10: Forward filter bank decomposition.

The FWT thus operates as follow:

– Input: signal $f \in \mathbb{C}^N$.

– Initialization: $a_J = f$.

– For $j = J, \ldots, j_0 - 1$.

$$a_{j+1} = (a_j \star \tilde{h}) \downarrow 2 \quad \text{and} \quad d_{j+1} = (a_j \star \tilde{g}) \downarrow 2$$

– Output: the coefficients $\{d_j\}_{j_0\leqslant j<J} \cup \{a_{j_0}\}$.

If $|h|, |g| \leqslant C$ so that both filter are compactly supported, then computing each $\mathcal{W}_j$ is $(2C)2^{-j}$ operation, so that the complexity of the whole wavelet transform is

$$\sum_{j=J}^{1} (2C)2^{-j} = (2C)2^{-J} = 2CN.$$

This shows that the fast wavelet transform is a linear time algorithm. Figure 3.11 shows the process of extracting iteratively the wavelet coefficients. Figure 3.12 shows an example of computation, where at each iteration, the coefficients of a_j and d_j are added to the left of the output vector.

Fast Haar transform. For the Haar wavelets, one has

$$\varphi_{j,n} = \frac{1}{\sqrt{2}}(\varphi_{j-1,2n} + \varphi_{j-1,2n+1}),$$

$$\psi_{j,n} = \frac{1}{\sqrt{2}}(\varphi_{j-1,2n} - \varphi_{j-1,2n+1}).$$

This corresponds to the filters

$$h = [\ldots,\ 0,\ h[0] = \frac{1}{\sqrt{2}},\ \frac{1}{\sqrt{2}},\ 0, \ldots],$$

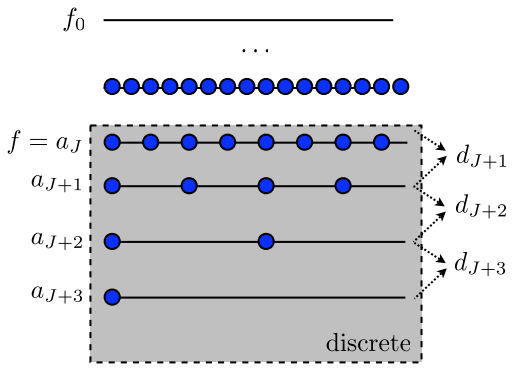


Figure 3.11: Pyramid computation of the coefficients.

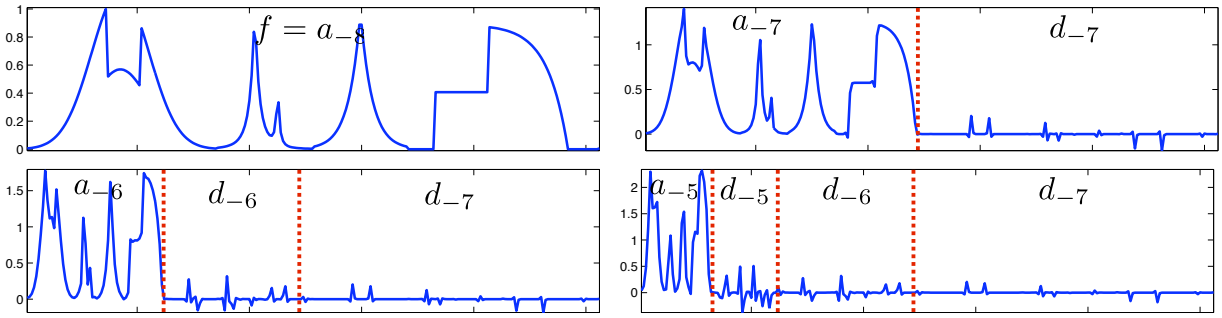


Figure 3.12: Wavelet decomposition algorithm.

$$g = [\ldots,\ 0,\ h[0] = \frac{1}{\sqrt{2}},\ -\frac{1}{\sqrt{2}},\ 0, \ldots].$$

The Haar wavelet transform algorithm thus processes by iterating averaging and differences:
– **Input:** signal $f \in \mathbb{C}^N$.
– **Initialization:** $a_J = f$.
– **For** $j = J, \ldots, j_0 - 1$.

$$a_{j+1,n} = \frac{1}{\sqrt{2}}(a_{j-1,2n} + a_{j-1,2n+1}) \quad \text{and} \quad d_{j+1,n} = \frac{1}{\sqrt{2}}(a_{j-1,2n} - a_{j-1,2n+1}).$$

– **Output:** the coefficients $\{d_j\}_{j_0 \leqslant j < J} \cup \{a_{j_0}\}$.

3.4.3 Inverse Fast Transform (iFWT)

The inverse algorithm proceeds by inverting each step (3.11)

$$\forall\, j = 0, \ldots, J+1, \quad a_{j-1} = \mathcal{W}_j^{-1}(a_j, d_j) = \mathcal{W}_j^*(a_j, d_j), \tag{3.11}$$

where $\mathcal{W}_j^*$ is the adjoint for the canonical inner product on $\mathbb{R}^{2^{-j+1}}$, i.e. when viewed as a matrix, the transpose.

We denote $\uparrow_2 : \mathbb{R}^{K/2} \to \mathbb{R}^K$ the up-sampling operator

$$a \uparrow_2 = (a_0, 0, a_1, 0, \ldots, 0, a_{K/2}, 0) \in \mathbb{R}^K.$$

Proposition 13. *One has*

$$\mathcal{W}_j^{-1}(a_j, d_j) = (a_j \uparrow_2) \star h + (d_j \uparrow_2) \star g.$$

Proof. Since $\mathcal{W}_j$ is orthogonal, $\mathcal{W}_j^{-1} = \mathcal{W}_j^*$. We write the whole transform as

$$\mathcal{W}_j = S_2 \circ C_{\bar{h},\bar{g}} \circ \mathcal{D} \quad \text{where} \quad \begin{cases} \mathcal{D}(a) = (a, a), \\ C_{\bar{h},\bar{g}}(a, b) = (\bar{h} \star a, \bar{g} \star a), \\ S_2(a, b) = (a \downarrow_2, b \downarrow_2). \end{cases}$$

One has the following adjoint operator

$$\mathcal{D}^*(a, b) = a + b, \quad C_{\bar{h},\bar{g}}^*(a, b) = (h \star a, g \star b), \quad \text{and} \quad S_2(a, b) = (a \uparrow_2, b \uparrow_2).$$

Indeed, let us check this for the convolution, assuming involved sequences are in ℓ_1 (they are actually finite sums when considering periodic signals),

$$\langle f \star \bar{h},\, g \rangle = \sum_n (f \star \bar{h})_n g_n = \sum_n \sum_k f_k h_{k-n} g_n = \sum_k f_k \sum_n h_{k-n} g_n = \langle f,\, h \star g \rangle,$$

for the copying

$$\langle \mathcal{D}(a),\, (u, v) \rangle = \langle a,\, u \rangle + \langle a,\, v \rangle = \langle a,\, u + v \rangle = \langle a,\, \mathcal{D}^*(u, v) \rangle,$$

and for the down-sampling

$$\langle f \downarrow_2,\, g \rangle = \sum_n f_{2n} g_n = \sum_n (f_{2n} g_n + f_{2n+1} 0) = \langle f,\, g \uparrow_2 \rangle.$$

This is shown using matrix notations in Figure 3.13. Putting everything together gives the desired formula. □

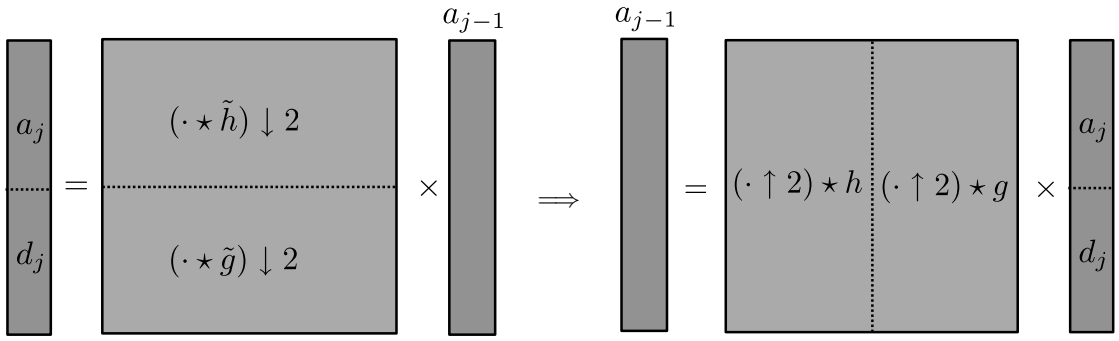


Figure 3.13: Wavelet inversion in matrix format.

The inverse Fast wavelet transform iteratively applies this elementary step

– Input: $\{d_j\}_{j_0 \leqslant j < J} \cup \{a_{j_0}\}$.

– For $j = j_0, \ldots, J + 1$.

$$a_{j-1} = (a_j \uparrow 2) \star h + (d_j \uparrow 2) \star g.$$

– Output: $f = a_J$.

This process is shown using a block diagram in Figure 3.14, which is the inverse of the block diagram 3.10.

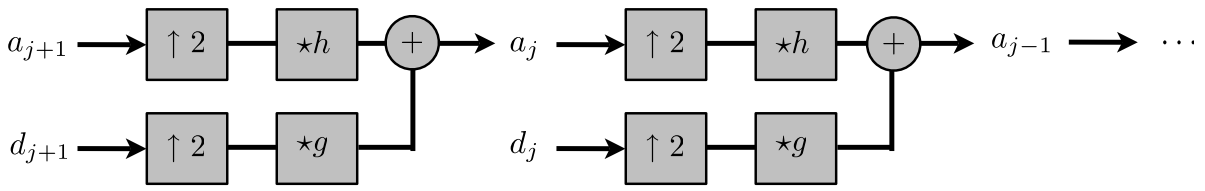


Figure 3.14: Backward filterbank recomposition algorithm.

3.5 2-D Wavelets

3.5.1 Anisotropic Wavelets

2-D anisotropic wavelets are defined using tensor product of Wavelet basis functions (2.12). The basis over $\mathbb{T}^2$ is thus of the form

$$\left\{\psi_{(j_1,j_2),(n_1,n_2)}\ ;\ (j_1,j_2)\in\mathbb{Z}^2, 0\leqslant n_1<2^{-j_1}, 0\leqslant n_2<2^{-j_2}\right\}$$

$$\text{where}\quad \psi_{(j_1,j_2),(n_1,n_2)}(x_1,x_2)=\psi_{j_1,n_1}(x_1)\psi_{j_2,n_2}(x_2). \tag{3.12}$$

The computation of the fast anisotropic Wavelet transform in 2-D is similar to the 2-D FFT detailed in Section 2.5.2. Viewing the input image $a_J\in\mathbb{R}^{2^{-J}\times 2^{-J}}$ as a matrix, one first apply the 1-D FWT to each row, and then to each column, resulting in a linear time $O(N)$ algorithm, where $N=2^{-2J}$.

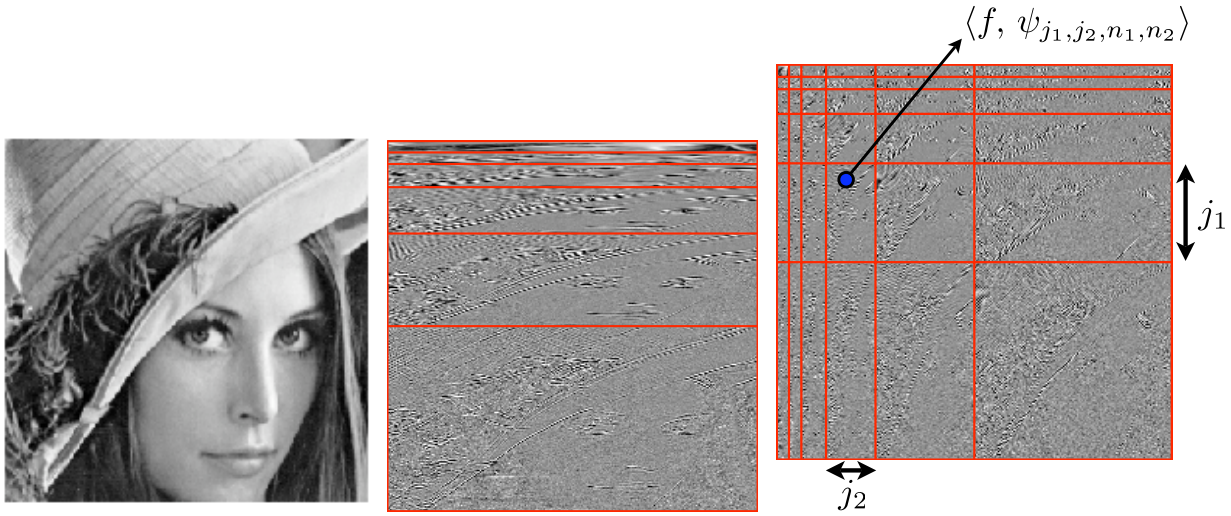


Image f Row transform Column transform.

Figure 3.15: Steps of the anisotropic wavelet transform.

3.5.2 Isotropic Wavelets

A major issue with these anisotropic wavelet (3.12) is that a function $\psi_{(j_1,j_2),(n_1,n_2)}$ is scaled independently in each direction, leads to functions concentrated along an axis-oriented rectangle of size $2^{-j_1}\times 2^{j_2}$. This is not a very good features (since natural images usually do not exhibit such an anisotropy) and typically leads to visually unpleasant artifacts when used for processing (denoising or compression).

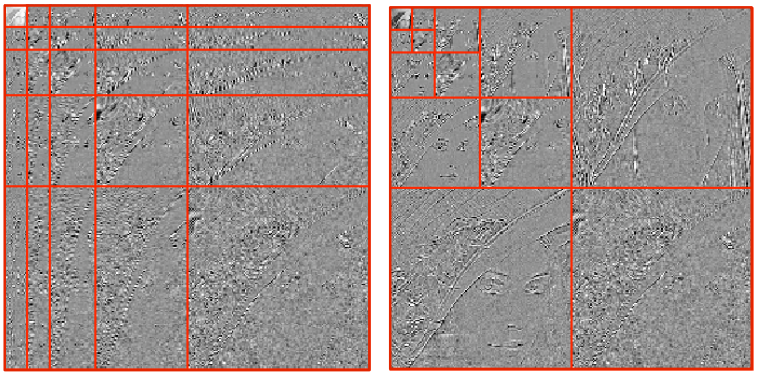


Figure 3.16: Anisotropic (left) versus isotropic (right) wavelet coefficients.

One rather use isotropic wavelet obtained by considering 2-D multi-resolution, obtained by tensor products of the 1-D approximation spaces

$$L^2(\mathbb{R}^2) \supset \ldots \supset V^{j-1} \otimes V^{j-1} \supset V_j \otimes V_j \supset V_{j+1} \otimes V_{j+1} \supset \ldots \supset \{0\}.$$

In the following, we denote

$$V_j^O \stackrel{\text{def.}}{=} V_j \otimes V_j$$

this isotropic 2-D multiresolution space.

Recall that the tensor product of two space $(V_1, V_2) \in L^2(\mathbb{R})^2$ is

$$V_1 \otimes V_2 = \text{Closure}\left(\text{Span}\left\{f_1(x_1)f_2(x_2) \in L^2(\mathbb{R}^2)\ ;\ f_1 \in V_1, f_2 \in V_2\right\}\right).$$

If $(\varphi_k^s)_k$ are Hilbertian bases for V_s, then one can show that $(\varphi_k^1(x_1)\varphi_k^2(x_2))_k$ is an Hilbertian basis for $V_1 \otimes V_2$.

One easily verify that one has the distributivity

$$(V_j \oplus^\perp W_J) \otimes (V_j \oplus^\perp W_J) = V_j^O \oplus^\perp W_j^V \oplus^\perp W_j^V \oplus^\perp W_j^V \quad \text{where} \quad \left\{ \begin{array}{l} W_j^V \stackrel{\text{def.}}{=} (V_j \otimes W_j), \\ W_j^H \stackrel{\text{def.}}{=} (W_j \otimes V_j), \\ W_j^D \stackrel{\text{def.}}{=} (W_j \otimes W_j). \end{array} \right.$$

Here the letters $\{V, H, D\}$ stands for *Vertical, Horizontal, Diagonal* detail spaces. This leads to the following diagram of embedded spaces

$$\begin{array}{ccccccccccc} \mathrm{L}^2(\mathbb{R}^2) & \rightarrow & \cdots & \rightarrow & V_{j-1} \otimes V_{j-1} & \rightarrow & V_j \otimes V_j & \rightarrow & V_{j+1} \otimes V_{j+1} & \rightarrow & \cdots \rightarrow \{0\} \\ & & & \searrow & W_{j-1}^{(2)} & \searrow & W_j^{(2)} & \searrow & W_{j+1}^{(2)} & \searrow & \end{array}$$

For $j \in \mathbb{Z}$, each of the three wavelet spaces is spanned with a wavelet, where basis elements are indexed by $n = (n_1, n_2) \in \mathbb{Z}$ (or in $\{0, \ldots, 2^{-j} - 1\}^2$ on the interval $\mathbb{T}$),

$$\forall\, \omega \in \{V, H, D\}, \quad W_j^\omega = \text{Span}\{\psi_{j,n_1,n_2}^\omega\}_{n_1,n_2}$$

where

$$\forall\, \omega \in \{V, H, D\}, \quad \psi_{j,n_1,n_2}^\omega(x) = \frac{1}{2^j}\psi^\omega\left(\frac{x_1 - 2^j n_1}{2^j}, \frac{x_2 - 2^j n_2}{2^j}\right)$$

and where the three mother wavelets are

$$\psi^H(x) = \psi(x_1)\varphi(x_2), \quad \psi^V(x) = \varphi(x_1)\psi(x_2), \quad \text{and} \quad \psi^D(x) = \psi(x_1)\psi(x_2).$$

Figure 3.17 displays an examples of these wavelets.

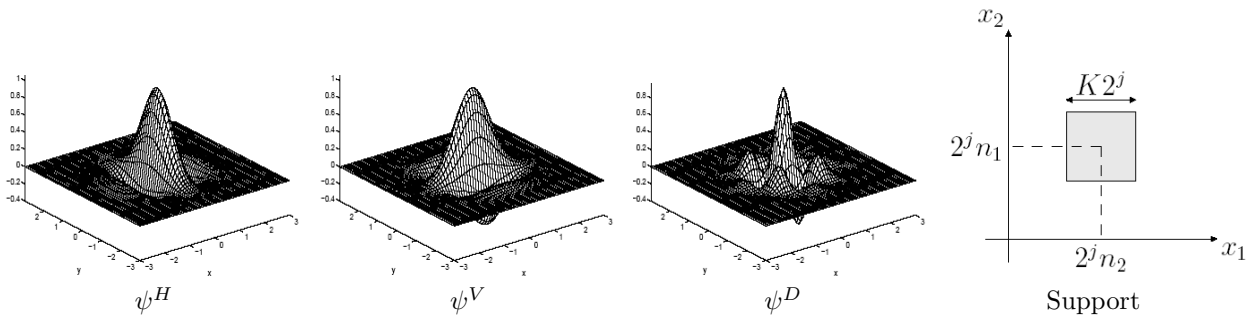


Figure 3.17: 2-D wavelets and their approximative support (right).

Haar 2-D multiresolution. For the Haar multiresolution, one obtains 2-D piecewise-constant Haar approximation. A function of $V_j \otimes V_j$ is constant on squares of size $2^j \times 2^j$. Figure 3.18 shows an example of projection of an image onto these 2-D Haar approximation spaces.

Figure 3.18: 2-D Haar approximation $P_{V_j^O} f$ for increasing j.

Discrete 2-D wavelet coefficients. Similarly to (3.5), we suppose that the sampling mechanism gives us access to inner product of the analog (continuous) signal f with the scaling function at scale $N = 2^{-J}$

$$\forall\, n, \in \{0, \ldots, N-1\}^2, \quad a_{J,n} = \langle f,\, \varphi^P_{J,n} \rangle$$

Discrete wavelet coefficients are defined as

$$\forall\, \omega \in \{V, H, D\},\ \forall\, J < j \leqslant 0,\ \forall\, 0 \leqslant n_1, n_2 < 2^{-j}, \quad d^\omega_{j,n} = \langle f,\, \psi^\omega_{j,n} \rangle.$$

(we use here periodized wavelets). Approximation coefficients are defined as

$$a_{j,n} = \langle f_0,\, \varphi^O_{j,n} \rangle.$$

Figure 3.19 shows examples of wavelet coefficients, that are packed in an image of N pixels. Figure 3.20 shows other examples of wavelet decompositions.

Forward 2-D wavelet transform basic step. A basic step of the computation of the 2-D wavelet transform computes detail coefficients and a low pass residual from the fine scale coefficients

$$a_{j-1} \longmapsto (a_j, d^H_j, d^V_j, d^D_j).$$

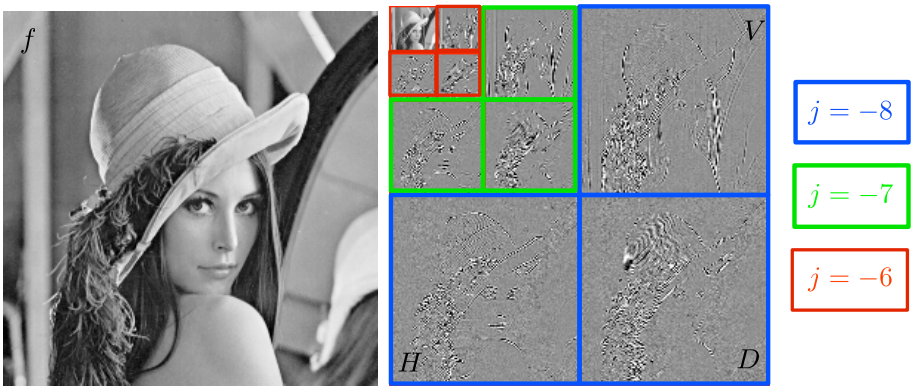


Figure 3.19: 2-D wavelet coefficients.

Similarly to the 1-D setting, this mapping is orthogonal, and is computed using the 1-D filtering and sub-sampling formula (3.8).

One first applies 1-D horizontal filtering and sub-sampling

$$\tilde{a}_j = (a_{j-1} \star^H \tilde{h}) \downarrow^H 2$$
$$\tilde{d}_j = (a_{j-1} \star^H \tilde{h}) \downarrow^H 2,$$

where $\star^H$ is the horizontal convolution, that applies the 1-D convolution to each column of a matrix

$$a \star^H b_{n_1,n_2} = \sum_{m_1=0}^{P-1} a_{n_1-m_1,n_2} b_{m_1}$$

where $a \in \mathbb{C}^{P \times P}$ and $b \in \mathbb{C}^P$ are matrix and vectors. The notation $\downarrow^H 2$ accounts for sub-sampling in the horizontal direction

$$(a \downarrow^H 2)_{n_1,n_2} = a_{2n_1,n_2}.$$

One then applies 1-D vertical filtering and sub-sampling to $\tilde{a}_j$ and $\tilde{d}_j$ to obtain

$$\begin{aligned} a_j &= (\tilde{a}_j \star^V \tilde{h}) \downarrow^V 2, & d_j^H &= (\tilde{d}_j \star^V \tilde{h}) \downarrow^V 2, \\ d_j^V &= (\tilde{a}_j \star^V \tilde{g}) \downarrow^V 2, & d_j^D &= (\tilde{d}_j \star^V \tilde{g}) \downarrow^V 2, \end{aligned}$$

where the vertical operators are defined similarly to horizontal operators but operating on rows.

These two forward steps are shown in block diagram in Figure 3.21. These steps can be applied in place, so that the coefficients are stored in an image of N pixels, as shown in Figure 3.22. This gives the traditional display of wavelet coefficients used in Figure 3.20.

Fast 2-D wavelet transform. The 2-D FWT algorithm iterates these steps through the scales:

– Input: signal $f \in \mathbb{C}^N$.

– Initialization: $a_J = f$.

– For $j = J, \ldots, j_0 - 1$.

$$\begin{aligned} \tilde{a}_j &= (a_{j-1} \star^H \tilde{h}) \downarrow^H 2, & d_j^V &= (\tilde{a}_j \star^V \tilde{g}) \downarrow^V 2, \\ \tilde{d}_j &= (a_{j-1} \star^H \tilde{h}) \downarrow^H 2, & d_j^H &= (\tilde{d}_j \star^V \tilde{h}) \downarrow^V 2, \\ a_j &= (\tilde{a}_j \star^V \tilde{h}) \downarrow^V 2, & d_j^D &= (\tilde{d}_j \star^V \tilde{g}) \downarrow^V 2. \end{aligned}$$

– Output: the coefficients $\{d_j^\omega\}_{j_0 \leqslant j < J, \omega} \cup \{a_{j_0}\}$.

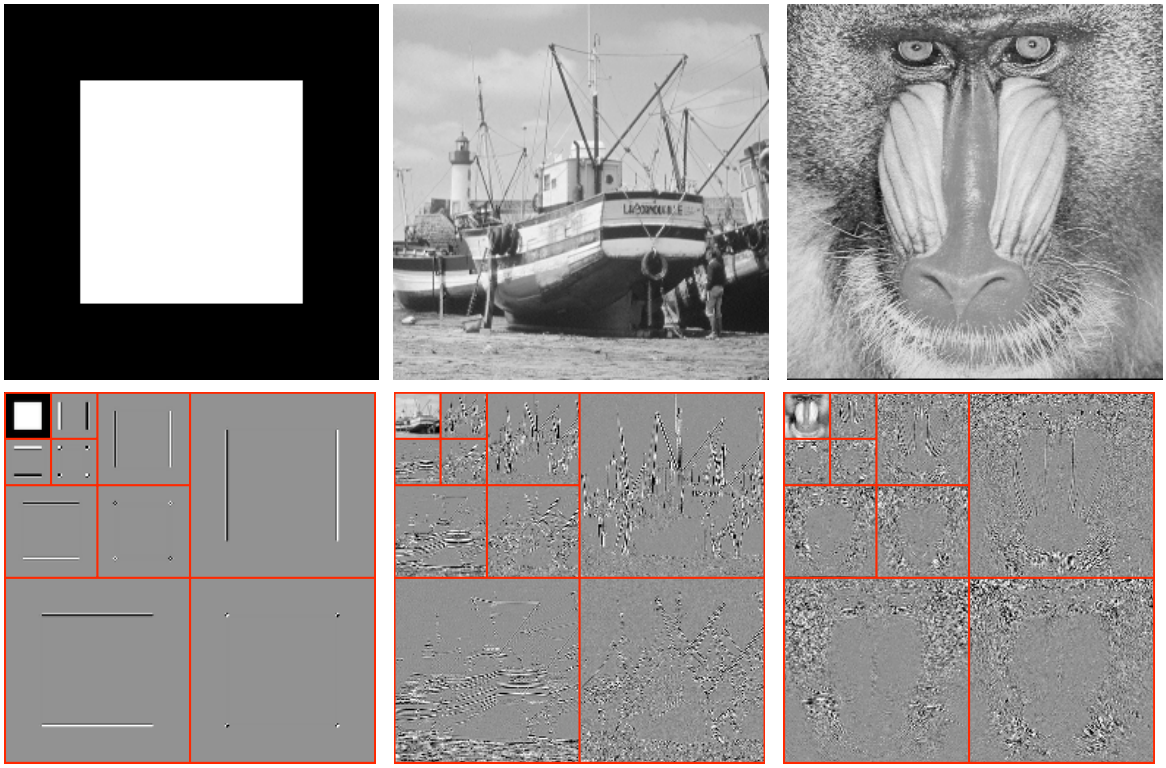


Figure 3.20: Examples of images (top row) and the corresponding wavelet coefficients (bottom row) .

Fast 2-D inverse wavelet transform. The inverse transform undo the horizontal and vertical filtering steps. The first step computes

$$\tilde{a}_j = (a_j \star^V h) \uparrow^V 2 + (d_j^V \star^V g) \uparrow^V 2,$$
$$\tilde{d}_j = (d_j^H \star^V h) \uparrow^V 2 + (d_j^D \star^V g) \uparrow^V 2,$$

where the vertical up-sampling is

$$(a \uparrow^V 2)_{n_1,n_2} = \begin{cases} a_{k,n_2} & \text{if} \quad n_1 = 2k, \\ 0 \quad \text{if} & n = 2k+1. \end{cases}$$

The second inverse step computes

$$a_{j-1} = (\tilde{a}_j \star^H h) \uparrow^H 2 + (\tilde{d}_j \star^H g) \uparrow^H 2.$$

Figure 3.23 shows in block diagram this inverse filter banks, that is the inverse of the diagram 3.21.

The inverse Fast wavelet transform iteratively applies these elementary steps

– Input: $\{d_j^\omega\}_{j_0 \leqslant j < J,\omega} \cup \{a_{j_0}\}$.

– For $j = j_0, \ldots, J+1$.

$$\tilde{a}_j = (a_j \star^V h) \uparrow^V 2 + (d_j^V \star^V g) \uparrow^V 2,$$
$$\tilde{d}_j = (d_j^H \star^V h) \uparrow^V 2 + (d_j^D \star^V g) \uparrow^V 2,$$
$$a_{j-1} = (\tilde{a}_j \star^H h) \uparrow^V 2 + (\tilde{d}_j \star^H g) \uparrow^V 2.$$

– Output: $f = a_J$.

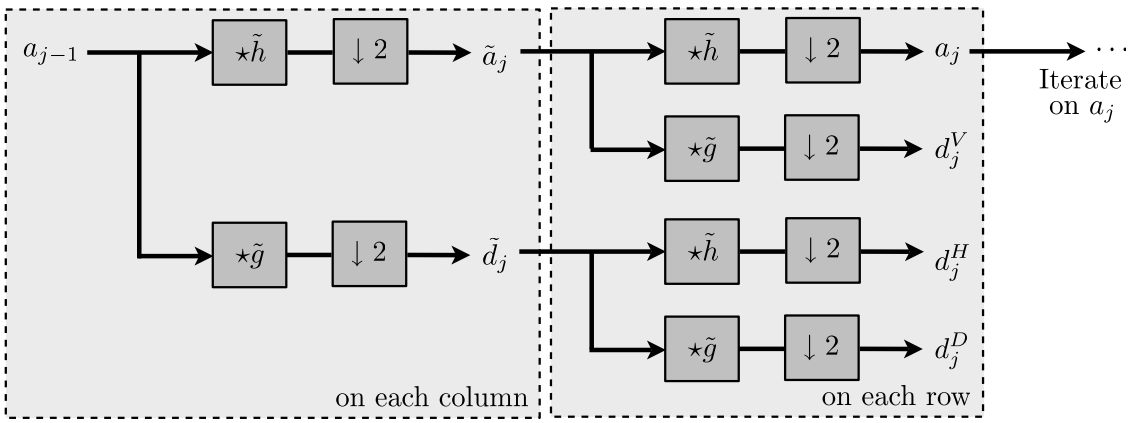


Figure 3.21: Forward 2-D filterbank step.

Figure 3.22: One step of the 2-D wavelet transform algorithm.

3.6 Wavelet Design

To be able to compute the wavelet coefficients using the FWT algorithm, it remains to know how to compute the scaling and wavelet functions. The FWT only makes use of the filters h and g, so instead of explicitly knowing the functions φ and ψ, one can only know these filters. Indeed, most of the known wavelets do not have explicit formula, and are implicitly defined through the cascade of the FWT algorithm.

This section shows what are the constraints h and g should satisfy, and gives practical examples. Furthermore, it shows that the knowledge of h determines g under the constraint of having quadrature filters, which is the most usual choice for wavelet analysis.

3.6.1 Low-pass Filter Constraints

We introduce the following three conditions on a filter h

$$\hat{h}(0) = \sqrt{2} \tag{C_1}$$

$$|\hat{h}(\omega)|^2 + |\hat{h}(\omega+\pi)|^2 = 2, \tag{C_2}$$

$$\inf_{\omega\in[-\pi/2,\pi/2]} |\hat{h}(\omega)| > 0. \tag{C^*}$$

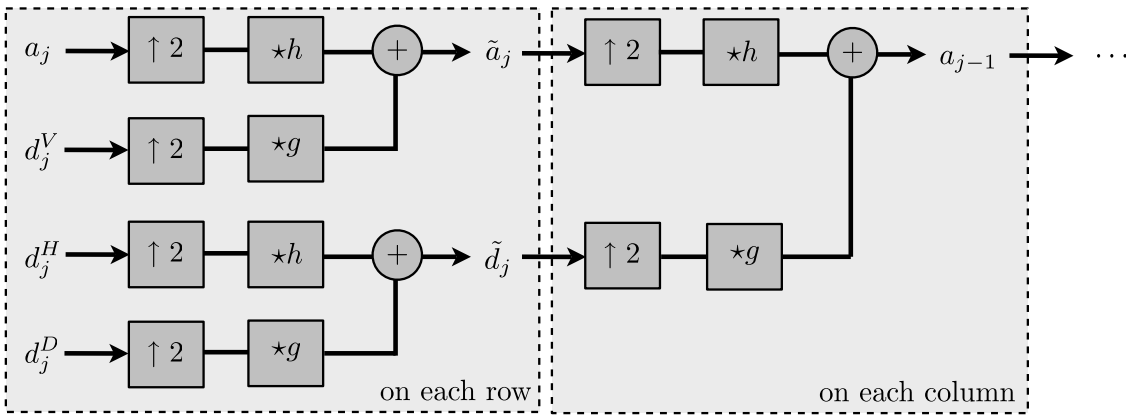


Figure 3.23: Backward 2-D filterbank step.

Here we are using the Fourier series associated to a filter $h \in \mathbb{R}^{\mathbb{Z}}$

$$\forall \omega \in \mathbb{R}/2\pi\mathbb{Z}, \quad \hat{h}(\omega) \stackrel{\text{def.}}{=} \sum_{n\in\mathbb{Z}} h_n e^{-\mathrm{i}n\omega}.$$

If $h \in \ell^1(\mathbb{Z})$, this defines a continuous periodic fonction $\hat{h} \in \mathcal{C}^0(\mathbb{R}/2\pi\mathbb{Z})$, and this definition can be extended to $h \in \ell^2(\mathbb{Z})$ and defines $\hat{h} \in L^2(\mathbb{R}/2\pi\mathbb{Z})$.

Theorem 3. *If φ defines multi-resolution approximation spaces, then (C_1) and (C_2) holds for h defined in (3.7). Conversely, if (C_1), (C_2) and (C^*) holds, then there exists a φ defining multi-resolution approximation spaces so that associated filter is h as defined in (3.7).*

Proof. We only prove the first statement of the theorem. The converse statement is much more difficult to prove.

We now prove condition (C_1). The refinement equation reads like a discrete-continuous convolution (or equivalently a convolution with a distribution)

$$\frac{1}{\sqrt{2}}\varphi\left(\frac{t}{2}\right) = \sum_{n\in\mathbb{Z}} h_n \varphi(t-n). \tag{3.13}$$

Denoting $h \star \varphi$ such a convolution, assuming $h \in \ell_1(\mathbb{Z})$ and $\varphi \in L^1(\mathbb{R})$, one check that one can apply Fubini and that $h \star \varphi \in L^1(\mathbb{R})$ and then

$$\begin{aligned}\mathcal{F}(\sum_{n\in\mathbb{Z}} h_n\varphi(t-n))(\omega) &= \int_{\mathbb{R}} \sum_{n\in\mathbb{Z}} h_n\varphi(t-n)e^{-\mathrm{i}\omega t}\mathrm{d}t = \sum_{n\in\mathbb{Z}} h_n \int_{\mathbb{R}} \varphi(t-n)e^{-\mathrm{i}\omega t}\mathrm{d}t \\ &= \sum_{n\in\mathbb{Z}} h_n e^{-\mathrm{i}n\omega} \int_{\mathbb{R}} \varphi(x)e^{-\mathrm{i}\omega x}\mathrm{d}x = \hat{\varphi}(\omega)\hat{h}(\omega)\end{aligned}$$

where we made the change of variable $x = t-n$. Note that here, $\hat{h}(\omega)$ is the 2π-periodic Fourier transform (i.e. Fourier series) of infinite filters defined in (3.6.1), whereas $\hat{\varphi}(\omega)$ is the Fourier transform of function. This is thus a product of a 2π-periodic function $\hat{h}$ and a non-periodic function $\hat{\varphi}$. We recall that $\mathcal{F}(f(\cdot/s)) = s\hat{f}(s\cdot)$. Over the Fourier domain, equation (3.13) thus reads

$$\hat{\varphi}(2\omega) = \frac{1}{\sqrt{2}}\hat{h}(\omega)\hat{\varphi}(\omega). \tag{3.14}$$

One can show that $\hat{\varphi}(0) \neq 0$ (actually, $|\hat{\varphi}(0)| = 1$), so that this relation implies the first condition (C_1).

We now prove condition (C_2). The orthogonality of $\varphi(\cdot - n)\}_n$ is rewritten using a continuous convolution as (see also Proposition 11)

$$\forall\, n \in \mathbb{Z}, \quad \varphi \star \bar{\varphi}(n) = \delta_0$$

where $\bar{\varphi}(x) = \varphi(-x)$, and thus over the Fourier domain, using (3.14) which shows $\hat{\varphi}(\omega) = \frac{1}{\sqrt{2}}\hat{h}(\omega/2)\hat{\varphi}(\omega/2)$

$$1 = \sum_k |\hat{\varphi}(\omega + 2k\pi)|^2 = \frac{1}{2}\sum_k |\hat{h}(\omega/2 + k\pi)|^2 |\hat{\varphi}(\omega/2 + k\pi)|^2.$$

Since $\hat{h}$ is 2π-periodic, one can split even and odd k and obtain

$$2 = |\hat{h}(\omega/2)|^2 \sum_k |\hat{\varphi}(\omega/2 + 2k\pi)|^2 + |\hat{h}(\omega/2 + \pi)|^2 \sum_k |\hat{\varphi}(\omega/2 + 2k\pi + \pi)|^2$$

This leads to condition (C_2). Re-using the fact that $\sum_k |\hat{\varphi}(\omega + 2k\pi)|^2 = 1$ for $\omega' = \omega/2$ in place of ω, one thus has

$$|\hat{h}(\omega')|^2 + |\hat{h}(\omega' + \pi)|^2 = 2.$$

We do not prove the converse statement, which requires to "create" a function φ from the filter h. The intuition is that iterating (3.14) leads informally to

$$\hat{\varphi}(\omega) = \prod_{k<0} \frac{\hat{h}(\omega/2^k)}{\sqrt{2}}. \tag{3.15}$$

Condition (C^*) can be shown to imply that this infinite product converge, and define a (non-periodic) function in $L^2(\mathbb{R})$. □

Note that for the converse statement of this theorem to holds, condition (C^*) imposes a control on the behavior of $\hat{h}$ near 0.

3.6.2 High-pass Filter Constraints

We now introduce the following two conditions on a pair of filter (g, h)

$$|\hat{g}(\omega)|^2 + |\hat{g}(\omega + \pi)|^2 = 2 \tag{C_3}$$

$$\hat{g}(\omega)\hat{h}(\omega)^* + \hat{g}(\omega + \pi)\hat{h}(\omega + \pi)^* = 0. \tag{C_4}$$

Figure 3.24 illustrates this constraint.

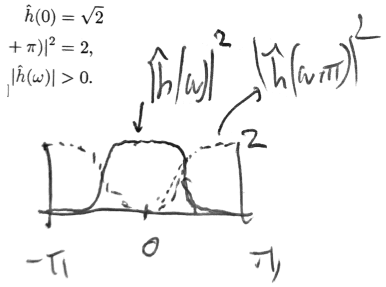


Figure 3.24: Low/high pass filter constraint.

Theorem 4. *If (φ, ψ) defines a multi-resolution analysis, then (C_3) and (C_4) holds for (h, g) defined in (3.7). Conversely, if (C_1) to (C_4) hold, then there exists a (φ, ψ) defining multi-resolution analysis so that associated filters are (h, g) as defined in (3.7). Furthermore,*

$$\hat{\psi}(\omega) = \frac{1}{\sqrt{2}} \hat{g}(\omega/2)\hat{\varphi}(\omega/2). \tag{3.16}$$

Proof. We prove condition (C_3). The refinement equation for the wavelet reads

$$\frac{1}{\sqrt{2}}\psi\left(\frac{t}{2}\right) = \sum_{n\in\mathbb{Z}} g_n \varphi(t-n)$$

and thus over the Fourier domain

$$\hat{\psi}(2\omega) = \frac{1}{\sqrt{2}} \hat{g}(\omega)\hat{\varphi}(\omega). \tag{3.17}$$

The orthogonality of $\{\psi(\cdot - n)\}_n$ is re-written

$$\forall\, n \in \mathbb{Z}, \quad \psi \star \bar{\psi}(n) = \delta_0$$

and thus over the Fourier domain (using Poisson formula, see also Proposition 11)

$$\sum_k |\hat{\psi}(\omega + 2k\pi)|^2 = 1.$$

Using the Fourier domain refinement equation (3.17), similarely to the proof of Theorem 3 for (C_1), this is equivalent to condition (C_3). Figure 3.25 shows the Fourier transform of two filters that satisfy this complementary condition.

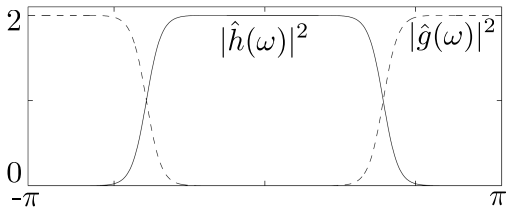


Figure 3.25: Complementarity between a low pass and a high pass wavelet filters h and g that satisfy condition (C_3).

We now prove condition (C_4). The orthogonality between $\{\psi(\cdot - n)\}_n$ and $\{\varphi(\cdot - n)\}_n$ is written as

$$\forall\, n \in \mathbb{Z}, \quad \psi \star \bar{\varphi}(n) = 0$$

and hence over the Fourier domain (using Poisson formula, similarly to Proposition 11)

$$\sum_k \hat{\psi}(\omega + 2k\pi)\hat{\varphi}^*(\omega + 2k\pi) = 0.$$

Using the Fourier domain refinement equations (3.14) and (3.17), this is equivalent to condition (C_4). □

Quadrature mirror filters. Quadrature mirror filters (QMF) defines g as a function of h so that the conditions of Theorem 4 are automatically satisfy. This choice is the natural choice to build wavelet filters, and is implicitly assumed in most constructions (other choices leading to the same wavelet function anyway, since it safisfies (3.16)).

Proposition 14. *For a filter $h \in \ell^2(\mathbb{Z})$ satisfying (C_1), defining $g \in \ell^2(\mathbb{Z})$ as*

$$\forall\, n \in \mathbb{Z}, \quad g_n = (-1)^{1-n} h_{1-n} \tag{3.18}$$

satisfies conditions (C_3) and (C_4).

Proof. One indeed has that

$$\hat{g}(\omega) = e^{-i\omega} \hat{h}(\omega + \pi)^*, \tag{3.19}$$

so that

$$|\hat{g}(\omega)|^2 + |\hat{g}(\omega+\pi)|^2 = |e^{-i\omega}\hat{h}(\omega+\pi)^*|^2 + |e^{-i(\omega+\pi)}\hat{h}(\omega+2\pi)^*|^2 = |\hat{h}(\omega+\pi)|^2 + |\hat{h}(\omega+2\pi)|^2 = 2.$$

where we used the fact that $\hat{h}$ is 2π-periodic, and also

$$\begin{aligned}\hat{g}(\omega)\hat{h}(\omega)^* + \hat{g}(\omega+\pi)\hat{h}(\omega+\pi)^* &= e^{-i\omega}\hat{h}(\omega+\pi)^*\hat{h}(\omega)^* + e^{-i(\omega+\pi)}\hat{h}(\omega+2\pi)^*\hat{h}(\omega+\pi)^* \\ &= (e^{-i\omega} + e^{-i(\omega+\pi)})\hat{h}(\omega+\pi)^*\hat{h}(\omega)^* = 0.\end{aligned}$$

□

3.6.3 Wavelet Design Constraints

According to the previous sections, the construction of a multi-resolution analysis (i.e. of functions (φ, ψ) is obtained by designing a filter h satisfying conditions (C_1) and (C_2). The function φ is obtained by an infinite cascade of filtering, or equivalently in the Fourier domain by (3.15), there is in general (put aside special case such as the Haar multiresolution) no closed form expression for φ. Once φ is defined, ψ is automatically defined by the relation (3.16) (and g can be defined as (3.19)).

There exists only one Fourier transform, but there is a large choice of different mother wavelet functions ψ. They are characterized by

– Size of the support.
– Number of oscillations (the so called number p of vanishing moments).
– Symmetry (only possible for non-orthogonal bases).
– Smoothness (number of derivatives).

We now detail how these constraints are integrated together with conditions (C_1)- (C_4).

Vanishing moments. A wavelet ψ has p vanishing moments if

$$\forall\, k \leqslant p-1, \quad \int_{\mathbb{R}} \psi(x)\, x^k \mathrm{d}x = 0. \tag{3.20}$$

This ensures that $\langle f, \psi_{j,n}\rangle$ is small if f is C^α, $\alpha < p$ on $\mathrm{Supp}(\psi_{j,n})$, which can be seen by doing a Taylor expansion of f around the point $2^j n$ on the support of the wavelet. This condition is equivalently expressed over Fourier as followed (see Fig. 3.26).

Proposition 15. *Assuming enough regularity of ψ, and using the QMF construction (3.19), it has p vanishing moments if and only if*

$$\forall\, k \leqslant p-1, \quad \frac{\mathrm{d}^k \hat{h}}{\mathrm{d}\omega^k}(\pi) = \frac{\mathrm{d}^k \hat{g}}{\mathrm{d}\omega^k}(0) = 0. \tag{3.21}$$

Proof. Since ψ is regular, one has that $\hat{\psi}^{(k)} = \mathcal{F}((-\mathrm{i}\cdot)^k \psi(\cdot))$, so that

$$(-\mathrm{i})^k \int_{\mathbb{R}} x^k \psi(x) \mathrm{d}x = \hat{\psi}^{(k)}(0).$$

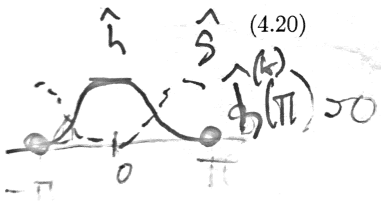


Figure 3.26: Wanishing moment condition as zero derivative conditions.

Relation (3.16) and (3.19) implies

$$\hat{\psi}(2\omega) = \hat{h}(\omega+\pi)^*\rho(\omega) \quad \text{where} \quad \rho(\omega) \stackrel{\text{def.}}{=} \frac{1}{\sqrt{2}} e^{-i\omega}\hat{\varphi}(\omega).$$

and differentiating this relation shows

$$2\hat{\psi}^{(1)}(2\omega) = \hat{h}(\omega+\pi)^*\rho^{(1)}(\omega) + \hat{h}^{(1)}(\omega+\pi)^*\rho(\omega)$$

which shows that, since $\rho(0) = \hat{\varphi}(0) \neq 0$, $\psi^{(1)}(0) = 0 \Leftrightarrow \hat{h}^{(1)}(\pi)^* = 0$. Recursing this argument and iterating the derrivative, one obtains that $\psi^{(k)}(0) = 0 \Leftrightarrow \hat{h}^{(k)}(\pi)^* = 0$ (assuming this hold for previous derivatives). □

Note that conditions (C_1) and (C_1) implies that $\hat{h}(\pi) = 0$, so that an admissible wavelet necessarily has 1 vanishing moment, i.e. $\int \psi = 0$. Condition (3.21) shows that having more vanishing moment is equivalent to having a Fourier transform $\hat{h}$ which is "flatter" arround $\omega = \pi$.

Support. Figure 3.27 shows the wavelet coefficients of a piecewise smooth signal. Coefficients of large magnitude are clustered near the singularities, because the wavelet ψ has enough vanishing moments.

To avoid that many wavelets create large coefficients near singularities, one should choose ψ with a small support. One can show that the size of the support of ψ is proportional to the size of the support of h. This requirement is however contradictory with the vanishing moment property (3.20). Indeed, one can prove that for an orthogonal wavelet basis with p vanishing moments

$$|\text{Supp}(\psi)| \geqslant 2p-1,$$

where $\sup(a)$ is the largest closed interval outside of which the function f is zero.

Chapter 4 studies in details the tradeoff of support size and vanishing moment to perform non-linear approximation of piecewise smooth signals.

Smoothness. In compression or denoising applications, an approximate signals is recovered from a partial set I_M of coefficients,

$$f_M = \sum_{(j,n)\in I_M} \langle f, \psi_{j,n}\rangle \psi_{j,n}.$$

This approximation f_M has the same smoothness as ψ.

To avoid visually unpleasant artifacts, one should thus choose a smooth wavelet function ψ. This is only for cosmetic reasons, since increasing smoothness does not leads to a better approximation. However, for most wavelet family, increasing the number of vanishing moments also increases the smoothness of the wavelets. This is for instance the case of the Daubechies family exposed in the next section.

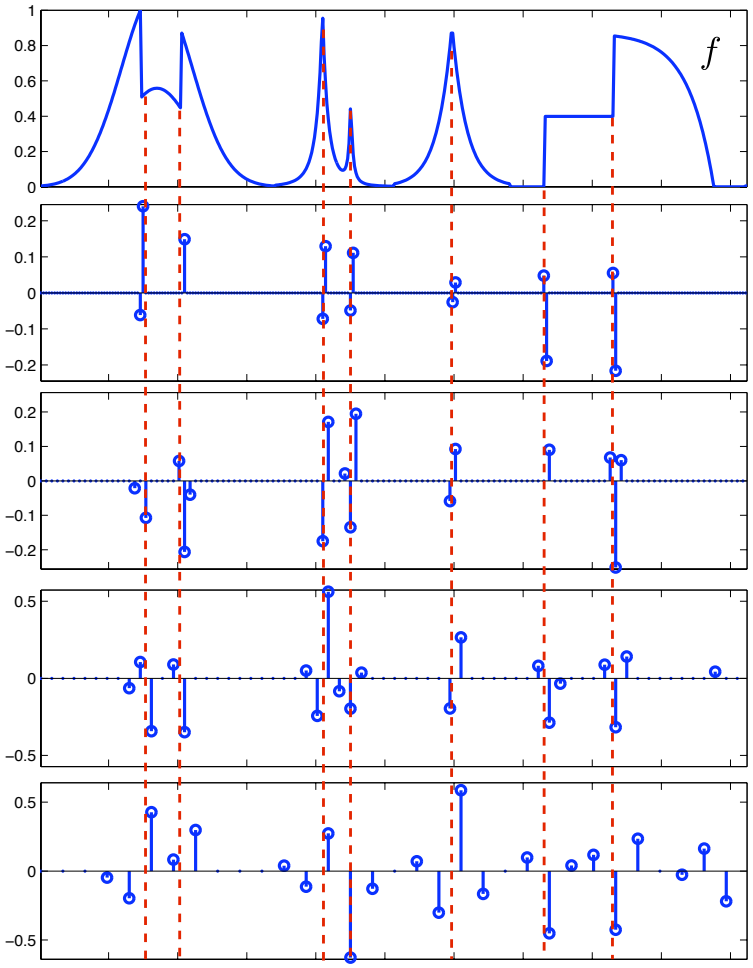


Figure 3.27: Location of large wavelet coefficients.

3.6.4 Daubechies Wavelets

To build a wavelet ψ with a fixed number p of vanishing moments, one designs the filter h, and use the quadrature mirror filter relation (3.19) to compute g. One thus look for h such that

$$|\hat{h}(\omega)|^2 + |\hat{h}(\omega+\pi)|^2 = 2, \quad \hat{h}(0) = \sqrt{2}, \quad \text{and} \quad \forall\, k < p, \ \frac{\mathrm{d}^k \hat{h}}{\mathrm{d}\omega^k}(\pi) = 0.$$

This corresponds to algebraic relationships between the coefficients of h, and it turns out that they can be solved explicitly using the Euclidean division algorithm for polynomials.

This leads to Daubechies wavelets with p vanishing moments, which are orthogonal wavelets with a minimum support length of $2p-1$.

For $p = 1$, it leads to the Haar wavelet, with

$$h = [h_0 = 0.7071; 0.7071].$$

For $p = 2$, one obtains the celebrated Daubechies 4 filter

$$h = [0.4830; h_0 = 0.8365; 0.2241; -0.1294],$$

and for $p = 3$,

$$h = [0; 0.3327; 0.8069; h_0 = 0.4599; -0.1350; -0.0854; 0.0352].$$

Wavelet display. Figure 3.28 shows examples of Daubechies mother wavelet functions with an increasing number of vanishing moments. These displays are obtained by computing in fact a discrete wavelet $\bar{\psi}_{j,n}$ defined in (**??**) for a very large number of samples N. This discrete wavelet is computed by applying the inverse wavelet transform to the coefficients $d_{j',n'} = \delta_{j-j'}\delta_{n-n'}$.

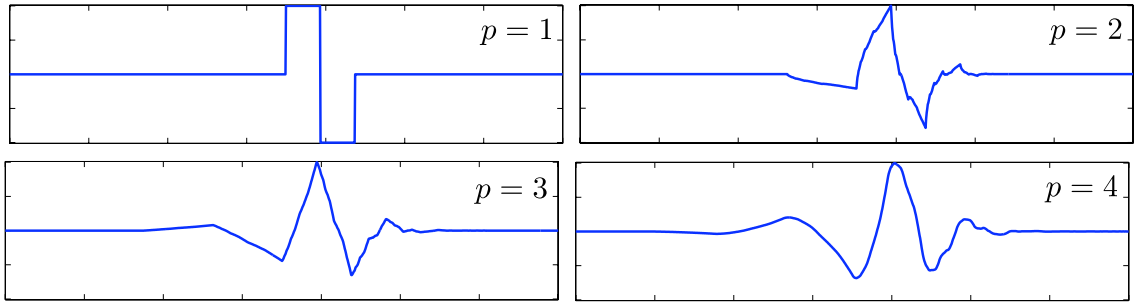


Figure 3.28: Examples of Daubechies mother wavelets ψ with an increasing number p of vanishing moments.

Chapter 4

Linear and Non-linear Approximation

This chapter studies the theory of signal and image approximation, and gives an application to lossy compression. This theoritical analysis is performed for continuous functions $f \in \mathrm{L}^2([0,1]^d)$ for $d = 1, 2$. This analysis is important to studies the performance of compression, denoising, and super-resolution applications.

4.1 Approximation

4.1.1 Approximation in an Ortho-basis

We consider an orthogonal basis $\mathcal{B} = \{\psi_m\}_m$ of $\mathrm{L}^2([0,1]^d)$, with for instance $d = 1$ (signals) or $d = 2$ (images). We recall that the decomposition of a signal in an orthonormal basis

$$f = \sum_{m \in \mathbb{Z}} \langle f, \psi_m \rangle \psi_m$$

gives back the original signal and thus produces no error. Processing algorithms modify the coefficients $\langle f, \psi_m \rangle$ and introduce some error.

The simplest processing computes an approximation by considering only a sub-set $I_M \subset \mathbb{Z}$ of M coefficients and performing the reconstruction from this subset

$$f_M \stackrel{\text{def.}}{=} \sum_{m \in I_M} \langle f, \psi_m \rangle \psi_m, \quad \text{where} \quad M = |I_M|.$$

The reconstructed signal f_M is the orthogonal projection of f onto the space

$$V_M \stackrel{\text{def.}}{=} \mathrm{Span}\{\psi_m \ ; \ m \in I_M\}.$$

Since V_M might depend on f, this projection $f \mapsto f_M$ might be non-linear.

Since the basis is orthogonal, the approximation error is

$$\|f - f_M\|^2 = \sum_{m \notin I_M} |\langle f, \psi_m \rangle|^2.$$

The important question is now to choose the set I_M, which might depend on the signal f itself.

4.1.2 Linear Approximation

Linear approximation is obtained by fixing once for all I_M, and thus using the same set of coefficients for all f. The mapping $f \mapsto f_M$ is a linear orthogonal projection on V_M, and it satisfies

$$(f + g)_M = f_M + g_M$$

For the Fourier basis, one usually selects the low-frequency atoms

$$I_M = \{-M/2+1, \ldots, M/2\}.$$

For a 1-D wavelet basis, one usually selects the coarse wavelets

$$I_M = \{m = (j,m) \; ; \; j \geqslant j_0\}$$

where j_0 is selected such that $|I_M| = M$.

Original f | Linear f_M^ℓ | Non-linear f_M^n

Figure 4.1: Linear versus non-linear wavelet approximation.

Figure 4.1, center, shows an example of such a linear approximation with wavelets. Linear approximation tends to performs poorly near singularities, because they introduce some blurring.

4.1.3 Non-linear Approximation

A non-linear approximation is obtained by choosing I_M depending on f. In particular, one would like to choose I_M to minimize the approximation error $\|f - f_M\|$. Since the basis is orthogonal, this is achieved by selecting the M largest coefficients in magnitude

$$I_M = \{M \text{ largest coefficients } |\langle f, \psi_m \rangle|\}.$$

This can be equivalently obtained using a thresholding

$$I_M = \{m \; ; \; |\langle f, \psi_m \rangle| > T\}$$

where T depends on the number of coefficients M,

$$M = \#\{m \; ; \; |\langle f, \psi_m \rangle| > T\}.$$

Computation of the threshold. There is a bijective 1:1 mapping between T and M obtained by ordering the coefficient magnitudes $|\langle f, \psi_m \rangle|$ by decaying order,

$$T = d_M \quad \text{where} \quad \{d_m\}_{m=0}^{N-1} = \{|\langle f, \psi_m \rangle|\}_0^{N-1} \quad \text{and} \quad d_m \geqslant d_{m+1}. \tag{4.1}$$

Figure 4.2 shows this mapping between M and T.

The following proposition show that the decay of the ordered coefficients is linked to the non-linear approximation decay.

Proposition 16. *One has*

$$d_m = O(m^{-\frac{\alpha+1}{2}}) \quad \Longleftrightarrow \quad \|f - f_M\|^2 = O(M^{-\alpha}). \tag{4.2}$$

Proof. One has

$$\|f - f_M\|^2 = \sum_{m>M} d_m^2$$

and

$$d_M^2 \leqslant \frac{2}{M} \sum_{m=M/2+1}^{M} d_m^2 \leqslant \frac{2}{M} \sum_{m>M/2} d_m^2 = \frac{2}{M} \|f - f_{M/2}\|^2.$$

□

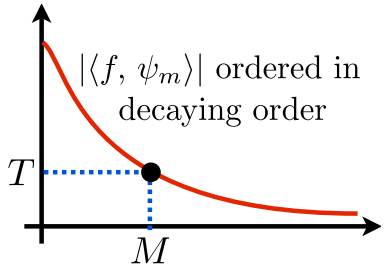


Figure 4.2: Decay of the ordered coefficients and determination of the threshold for non-linear approxiation.

Hard thresholding. The non-linear approximation is re-written as

$$f_M = \sum_{|\langle f, \psi_m \rangle| > T} \langle f, \psi_m \rangle \psi_m = \sum_m S_T(\langle f, \psi_m \rangle) \psi_m, \tag{4.3}$$

where

$$S_T(x) = \begin{cases} x & \text{if} \quad |x| > T \\ 0 & \text{if} \quad |x| \leqslant T \end{cases} \tag{4.4}$$

is the hard thresholding, that is displayed in Figure 4.3.

4.2 Signal and Image Modeling

A signal model is a constraint $f \in \Theta$, where $\Theta \subset \mathrm{L}^2([0,1]^d)$ is a set of signals one is interested in. Figure 4.4 shows different class of models for images, that we describe in the following paragraph.

4.2.1 Uniformly Smooth Signals and Images

Signals with derivatives. The simplest model is made of uniformly smooth signals, that have bounded derivatives

$$\Theta = \left\{ f \in \mathrm{L}^2([0,1]^d) \; ; \quad \|f\|_{\mathrm{C}^\alpha} \leqslant C \right\}, \tag{4.5}$$

where $C > 0$ is a fixed constant, and where in 1-D

$$\|f\|_{\mathrm{C}^\alpha} \overset{\text{def.}}{=} \max_{k \leqslant \alpha} \left\| \frac{\mathrm{d}^k f}{\mathrm{d}t^k} \right\|_\infty.$$

This extends to higher dimensional signals by considering partial derivatives along each direction.

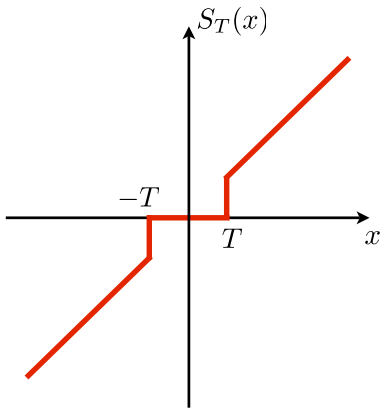
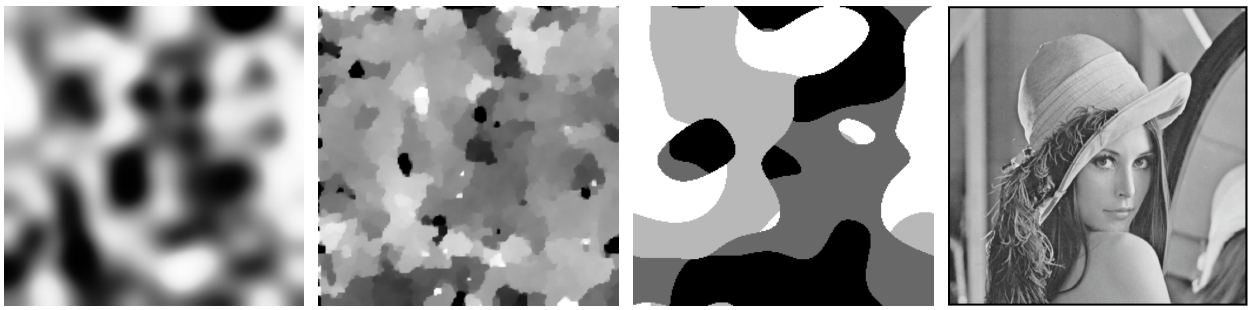


Figure 4.3: Hard thresholding.



Regular Bounded variations Cartoon Natural

Figure 4.4: Examples of different kinds of image models.

Sobolev smooth signals and images. A smooth C^α signals in (4.5) has derivatives with bounded energy, so that

$$\frac{\mathrm{d}^\alpha f}{\mathrm{d}t^\alpha}(t) = f^{(\alpha)}(t) \in \mathrm{L}^2([0,1]).$$

Using the fact that

$$\hat{f}_m^{(\alpha)} = (2i\pi m)^\alpha \hat{f}_m$$

where $\hat{f}$ is the Fourier coefficient defined in (2.2), except we are here on $\mathbb{R}/\mathbb{Z}$ in place of $\mathbb{R}/2\pi\mathbb{Z}$,

$$\hat{f}_n \overset{\text{def.}}{=} \int_0^1 e^{-2\mathrm{i}\pi n x} f(x)\mathrm{d}x,$$

one defines a so-called Sobolev functional

$$\|f\|_{\mathrm{Sob}(\alpha)}^2 = \sum_{m\in\mathbb{Z}} |2\pi m|^{2\alpha} |\langle f,\, e_m\rangle|^2, \tag{4.6}$$

that satisfies $\|f\|_{\mathrm{Sob}(\alpha)} = \|f^{(\alpha)}\|$ for smooth functions. This Sobolev functional is extended to signals that have derivatives in the sense of distribution in $\mathrm{L}^2([0,1])$.

This definition extends to distributions and signals $f \in \mathrm{L}^2([0,1]^d)$ of arbitrary dimension d as

$$\|f\|_{\mathrm{Sob}(\alpha)}^2 = \sum_{m\in\mathbb{Z}^d} (2\pi\|m\|)^{2\alpha} |\langle f,\, e_m\rangle|^2, \tag{4.7}$$

The C^α-Sobolev model

$$\Theta = \left\{ f \in \ell^2([0,1]^d) \; ; \; \max_{k \leqslant \alpha} \|f\|^2_{\mathrm{Sob}(k)} \leqslant C \right\} \tag{4.8}$$

generalizes the C^α smooth image model (4.5).

Figure 4.5 shows images with an increasing Sobolev norm for $\alpha = 2$.

Figure 4.5: Images with increasing Sobolev norm.

4.2.2 Piecewise Regular Signals and Images

Piecewise smooth signals. Piecewise smooth signals in 1-D are functions $f \in \mathrm{L}^2([0,1])$ that are C^α smooth outside a set of less than K pointwise discontinuities

$$\Theta = \left\{ f \in \mathrm{L}^2([0,1]) \; ; \; \exists\, (t_i)_{i=0}^{K-1}, \; \|f_{(t_i,t_{i+1})}\|_{\mathrm{C}^\alpha} \leqslant C \right\} \tag{4.9}$$

where $f_{(t_i,t_{i+1})}$ is the restriction of f to the open interval (t_i, t_{i+1}).

Piecewise smooth images. Piecewise smooth images are 2-D functions $f \in \mathrm{L}^2([0,1]^2)$ that are C^α regular outside a set of less than K curves that have a finite perimeter

$$\Theta = \left\{ f \in \mathrm{L}^2([0,1]^2) \; ; \; \exists\, \Gamma = (\gamma_i)_{i=0}^{K-1}, \; \|f\|_{\mathrm{C}^\alpha(\Gamma^c)} \leqslant C_1 \quad \text{and} \quad |\gamma_i| \leqslant C_2 \right\} \tag{4.10}$$

where $|\gamma_i|$ is the length of the curve γ_i and where $\|f\|_{\mathrm{C}^\alpha(\Gamma^c)}$ is the maximum norm of the derivatives of f outside the set of curves Γ.

Segmentation methods such as the one proposed by Mumford and Shah [18] implicitly assume such a piecewise smooth image modem.

4.2.3 Bounded Variation Signals and Images

Signals with edges are obtained by considering functions with bounded variations

$$\Theta = \left\{ f \in \mathrm{L}^2(\mathbb{R}^d) \; ; \; \|f\|_\infty \leqslant C_1 \quad \text{and} \quad \|f\|_{\mathrm{TV}} \leqslant C_2 \right\}. \tag{4.11}$$

For $d = 1$ and $d = 2$, this model generalizes the model of piecewise smooth signals (4.9) and images (4.10).

The total variation of a smooth function is

$$\int \|\nabla f(x)\| \mathrm{d}x$$

where

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_i} \right)_{i=0}^{d-1} \in \mathbb{R}^d$$

is the gradient vector at x. The total variation is extended to discontinuous images, that might for instance exhibit jumps across singular curves (the edges of the image). In particular, the total variation of a piecewise smooth image is the sum of the lengths of its level sets

$$\|f\|_{\mathrm{TV}} = \int_{-\infty}^{\infty} |\mathcal{L}_t(f)| \mathrm{d}t < +\infty \quad \text{where} \quad \mathcal{L}_t(f) = \{x \,;\, f(x) = t\}\,, \tag{4.12}$$

and where $|\mathcal{L}_t(f)|$ is the length of $\mathcal{L}_t(f)$. For a set $\Omega \subset \mathbb{R}^2$ with finite perimeter $|\partial\Omega|$, then

$$\|1_\Omega\|_{\mathrm{TV}} = |\partial\Omega|.$$

The model of bounded variation was introduced in image processing by Rudin, Osher and Fatemi [22].

4.2.4 Cartoon Images

The bounded variation image model (4.11) does not constrain the smoothness of the level set curves $\mathcal{L}_t(f)$. Geometrical images have smooth contour curves, which should be taken into account to improve the result of processing methods.

The model of C^α cartoon images is composed of 2-D functions that are C^α regular outside a set of less that K regular edge curves γ_i

$$\Theta = \left\{ f \in \mathrm{L}^2([0,1]^2) \,;\, \exists\, \Gamma = (\gamma_i)_{i=0}^{K-1},\ \|f\|_{\mathrm{C}^\alpha(\Gamma^c)} \leqslant C_1 \quad \text{and} \quad \|\gamma_i\|_{\mathrm{C}^\alpha} \leqslant C_2 \right\} \tag{4.13}$$

where each γ_i is a arc-length parameterization of the curve $\gamma_i : [0, A] \mapsto [0,1]^2$. Figure 4.6 shows cartoon images with increasing total variation $\|f\|_{\mathrm{TV}}$.

Figure 4.6: Cartoon image with increasing total variation.

Typical images might also be slightly blurred by optical diffraction, so that one might consider a blurred cartoon image model

$$\tilde{\Theta} = \left\{ \tilde{f} = f \star h \in \mathrm{L}^2([0,1]^2) \,;\, f \in \Theta \quad \text{and} \quad h \in \mathcal{H} \right\} \tag{4.14}$$

where Θ is the model of sharp (unblurred) images (4.13) and $\mathcal{H}$ is a set of constraints on the blurring kernel, for instance $h \geqslant 0$ should be smooth, localized in space and frequency. This unknown blurring makes difficult brute force approaches that detects the edges location Γ and then process the regular parts in $[0,1]^2 \backslash \Gamma$.

Figure 4.7 shows examples of images in Θ and $\tilde{\Theta}$.

4.3 Efficient approximation

4.3.1 Decay of Approximation Error

To perform an efficient processing of signals or images in Θ, the goal is to design an orthogonal basis such that the non-linear approximation error $\|f - f_M\|$ decays as fast as possible to 0 when M increases.

Figure 4.7: Examples of cartoon images: sharp discontinuities (left and center) and smooth discontinuities (right).

Polynomial error decay. This error decay measured using a power law

$$\forall\, f \in \Theta,\ \forall\, M, \quad \|f - f_M\|^2 \leqslant C_f M^{-\alpha} \tag{4.15}$$

where α is independent of f and should be as large as possible. The parameter α depends on the basis and on Θ. It is a class-complexity parameter that describes the overall complexity of signals in Θ with respect to the orthogonal basis one considers for approximation. The parameter C_f depends on f and describes the complexity of the signal f within its class Θ.

Relevance for compression, denoising and inverse problems. Monitoring the decay of approximation error is not only interesting from a mathematical point of view. Section 5.1 shows that the compression error is close to the non-linear approximation error. Bases that are efficient for approximation are thus also efficient for compression.

Chapter **??** shows that a similar conclusion holds for non-linear denoising with thresholding. Efficient denoisers are obtained by performing a non-linear approximation of the noisy image in a well chosen basis. The average denoising error with respect to a random noise is closely related to the approximation error.

Chapter **??** shows that ill-posed inverse problems such as super-resolution of missing information can be solved by taking advantage of the compressibility of the signal or the image in a well chosen basis. A basis that is efficient for approximation of the high resolution signal is needed to recover efficiently missing information. The performance of these schemes is difficult to analyze, and the basis atoms must also be far enough from the kernel of the operator that removes information.

Comparison of signals. For a fixed basis (for instance wavelets), the decay of $\|f - f_M\|$ allows one to compare the complexity of different images. Figure 4.9 shows that natural images with complicated geometric structures and textures are more difficult to approximate using wavelets.

Since the approximation error often decays in a power-low fashion (4.15), the curves are displayed in a log-log plot, so that

$$\log(\|f - f_M\|^2) = \mathrm{cst} - \alpha \log(M)$$

and hence one should expect an affine curve with slope $-\alpha$. Due to discretization issue, this is only the case for value of $M \ll N$, since the error quickly drops to zero for $M \approx N$.

4.3.2 Comparison of bases.

For a given image f, one can compare different ortho-bases using the decay of $\|f - f_M\|$. Figure 4.11 shows the efficiency of several bases to approximate a fixed natural image with contours and textures. The Fourier basis described in Section **??** is highly innefficient because of periodic boundary artifact and the global

Smooth Cartoon Natural #1 Natural #2

Figure 4.8: Several different test images.

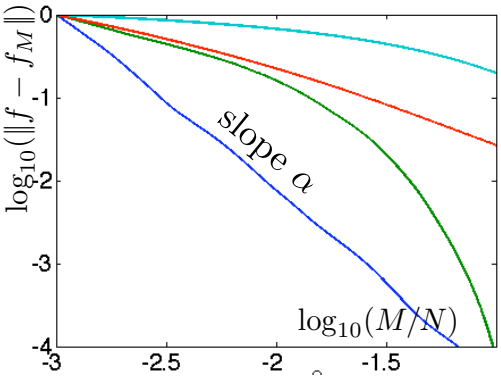


Figure 4.9: Comparison of approximation error decay in wavelets for different images shown in Figure 4.8.

support of its atoms that fail to capture contours. The cosine basis uses symmetric boundary conditions and thus removes the boundary artifacts, but it still not able to resolve efficiently localized features. The local DCT basis corresponds to the union of local cosine bases defined on small square patches. It is more efficient since its atoms have a local support. However, it gives bad approximation for a small number M of kept coefficients, because of blocking artifacts. The isotropic wavelet basis detailed in Section **??** gives the best approximation results because its is both composed of localized atoms and does not have a block structure but rather a multiresolution structure.

Figure 4.12 summarizes the different type of approximation decays for various class of data, which are detailed in the following section.

4.4 Fourier Linear Approximation of Smooth Functions

The smooth signal and image model (4.5) assumed that the analog function have bounded α continuous derivatives. A function f with a large α has more smoothness, and is thus simpler to approximate. Figure 4.5 shows images with increasing smoothness.

4.4.1 1-D Fourier Approximation

A 1-D signal $f \in \mathrm{L}^2([0,1])$ is associated to a 1-periodic function $f(t+1) = f(t)$ defined for $t \in \mathbb{R}$.

Fourier
SNR=17.1dB

Cosine
SNR=17.5dB

Local DCT
SNR=18.4dB

Wavelets
SNR=19.3dB

Figure 4.10: Comparison of approximation errors for different bases using the same number $M = N/50$ of coefficients.

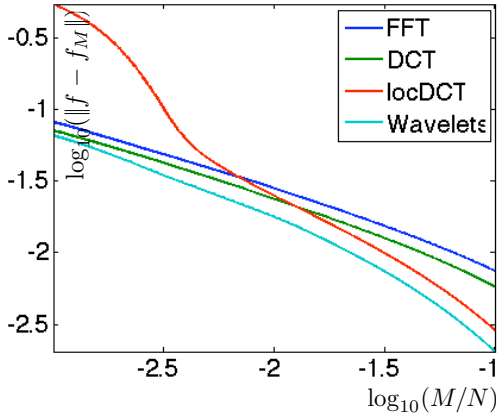


Figure 4.11: Comparison of approximation error decay for different bases.

Low pass approximation. We consider a linear Fourier approximation, that only retains low frequencies

$$f_M^{\text{lin}} = \sum_{m=-M/2}^{M/2} \langle f, e_m \rangle e_m$$

where we use the 1-D Fourier atoms

$$\forall\, n \in \mathbb{Z}, \quad e_m(t) \stackrel{\text{def.}}{=} e^{2i\pi m t}.$$

We note that f_M actually requires $M+1$ Fourier atoms.

Figure 4.13 shows examples of such linear approximation for an increasing value of M. Since the original function f is singular (no derivative), this produces a large error and one observe ringing artifacts near singularities.

This low pass approximation corresponds to a filtering, since

$$f_M = \sum_{m=-M/2}^{M/2} \langle f, e_m \rangle e_m = f \star h_M \quad \text{where} \quad \hat{h}_M \stackrel{\text{def.}}{=} 1_{[-M/2, M/2]}.$$

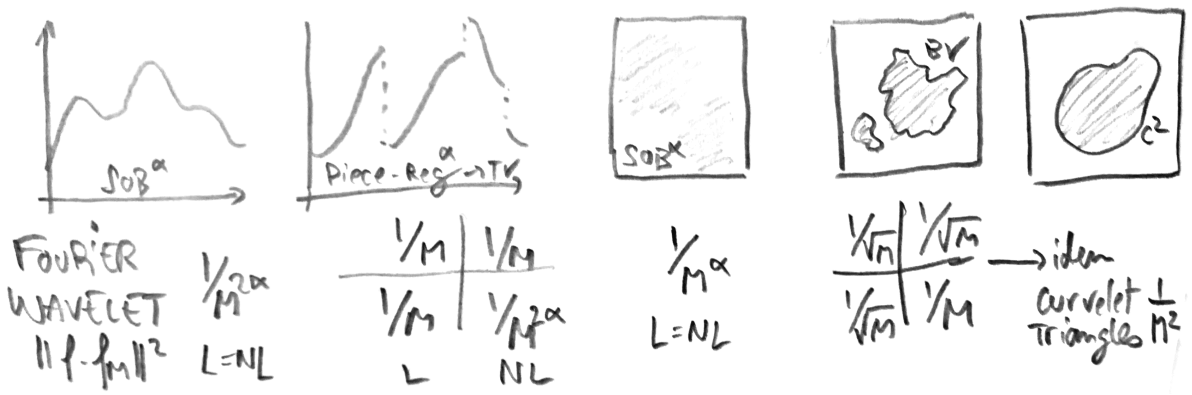


Figure 4.12: Summary of linear and non-linear approximation rates $\|f - f_M\|^2$ for different classes of 1-D signals and images.

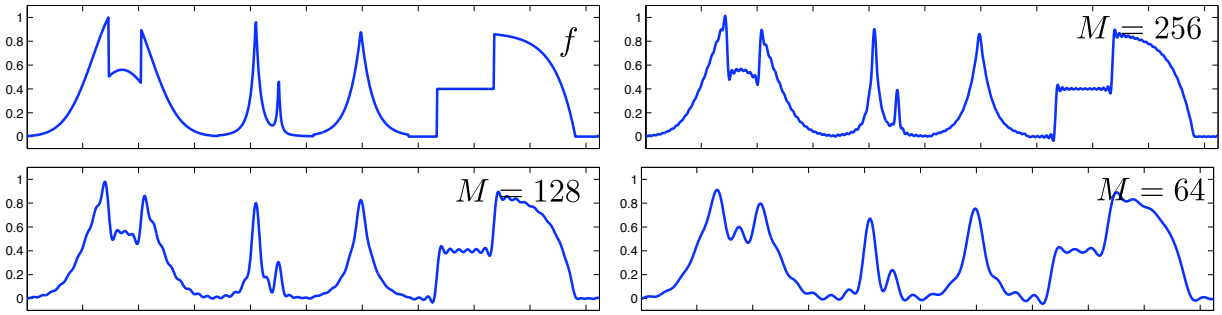


Figure 4.13: Fourier approximation of a signal.

Here, h_M is the so-called Dirichlet kernel (see also the chapter on Fourier).

The following proposition shows that this approximation decays fast for $\mathcal{C}^\alpha$ signals.

Proposition 17. *One has for $f \in \mathcal{C}^\alpha(\mathbb{R}/\mathbb{Z})$*

$$\|f - f_M^{lin}\|^2 = O(M^{-2\alpha+1}).$$

Proof. Using integration by part, one shows that for a C^α function f in the smooth signal model (4.5), $\mathcal{F}(f^{(\ell)})_m = (2\mathrm{i}m\pi)^\ell \hat{f}_m$, so that one has using Cauchy-Schwartz

$$|\langle f, e_m\rangle| \leqslant |2\pi m|^{-\alpha} \|f^{(\alpha)}\|_2.$$

This implies

$$\|f - f_M^{\mathrm{lin}}\|^2 = \sum_{|m|>M/2} |\langle f, e_m\rangle|^2 \leqslant \|f^{(\alpha)}\|_2^2 \sum_{|m|>M/2} |2\pi m|^{-2\alpha} = O(M^{-2\alpha+1}).$$

□

We show next that a slightly modified proof gives a better decay (assuming $f^{(\alpha)}$ is in $L^2(\mathbb{R}/\mathbb{Z})$) and that this conclusion is valid for a larger class of Sobolev functions.

Proposition 18. *For signal f in the Sobolev model (4.8), i.e. $f^{(\alpha)} \in L^2(\mathbb{R})$, one has*

$$\|f - f_M^{lin}\|^2 \leqslant C\|f^{(\alpha)}\|^2 M^{-2\alpha}. \tag{4.16}$$

Proof. One has

$$\|f^{(\alpha)}\|^2 = \sum_m |2\pi m|^{2\alpha} |\langle f, e_m\rangle|^2 \geqslant \sum_{|m|>M/2} |2\pi m|^{2\alpha} |\langle f, e_m\rangle|^2 \tag{4.17}$$

$$\geqslant (\pi M)^{2\alpha} \sum_{m>M/2} |\langle f, e_m\rangle|^2 \geqslant (\pi M)^{2\alpha} \|f - f_M^{\mathrm{lin}}\|^2. \tag{4.18}$$

□

One can also shows that this asymptotic error decay is optimal, and that the non-linear approximation error in Fourier also decays like $O(M^{-2\alpha})$.

For a signal in the piecewise smooth model (4.9), such as the one shows in Figure 4.13, one only has a slow decay of the linear and non-linear approximation error

$$\|f - f_M\|^2 \leqslant C_f M^{-1} \tag{4.19}$$

and Fourier atoms are not anymore optimal for approximation. A typical example is $f = 1_{[}a, b]$ the indicator of a step, for which $|\hat{f}_m| \sim 1/m$, and thus $\|f - f_M\|^2 \sim \sum_{|m|>M} 1/|m|^2 \sim 1/M$ for both linear and non-linear approximations.

4.4.2 Sobolev Images

This analysis caries over to images and higher dimensional datasets by considering a Sobolev functional (4.7) for $d > 1$.

The linear and non-linear approximation of an α-regular Sobolev image then satisfy

$$\|f - f_M\|^2 \leqslant C\|f^{\alpha}\|^2 M^{-\alpha}.$$

For d-dimensional data $f : [0,1]^d \to \mathbb{R}$, one would have an error decay of $O(M^{-2\alpha/d})$.

For an image in the piecewise smooth model (4.10), the linear and non-linear error decays are slow,

$$\|f - f_M\|^2 \leqslant C_f M^{-1/2}, \tag{4.20}$$

and Fourier atoms are not anymore optimal for approximation.

4.5 Wavelet Approximation of Piecewise Smooth Functions

Wavelet approximation improve significantly over Fourier approximation to capture singularities. This is due to the localized support of wavelets.

4.5.1 Decay of Wavelet Coefficients

To efficiently approximate regular parts of signals and images, one uses wavelet with a large enough number p of vanishing moments

$$\forall\, k < p, \quad \int \psi(x) x^k \mathrm{d}x = 0.$$

This number p should be larger than the regularity α of the signal outside singularities (for instance jumps or kinks).

To quantify the approximation error decay for piecewise smooth signals (4.9) and images (4.10), one needs to treat differently wavelets that are in regular and singular areas. Figure 4.15 shows for a signal and an image the localization of singular and regular parts.

Figure 4.14: Linear (top row) and non-linear (bottom row) Fourier approximation.

Proposition 19. *If f is C^α on $\mathrm{supp}(\psi_{j,n})$, with $p \geqslant \alpha$, one has*

$$|\langle f, \psi_{j,n}\rangle| \leqslant C_f \|\psi\|_1 2^{j(\alpha+d/2)}. \tag{4.21}$$

In general, one always has for bounded f

$$|\langle f, \psi_{j,n}\rangle| \leqslant \|f\|_\infty \|\psi\|_1 2^{j\frac{d}{2}}.$$

Proof. If f is C^α on $\mathrm{supp}(\psi_{j,n})$, with $p \geqslant \alpha$, then one can perform a Taylor expansion of f around the point $2^j n$

$$f(x) = P(x - 2^j n) + R(x - 2^j n) = P(2^j t) + R(2^j t)$$

where $\deg(P) < \alpha$ and

$$|R(x)| \leqslant C_f \|x\|^\alpha.$$

One then bounds the wavelet coefficient

$$\langle f, \psi_{j,n}\rangle = \frac{1}{2^{j\frac{d}{2}}} \int f(x)\psi\left(\frac{x - 2^j n}{2^j}\right) \mathrm{d}x = 2^{j\frac{d}{2}} \int R(2^j t)\psi(t)\mathrm{d}t$$

where we have performed the change of variable $t = \frac{x-2^j n}{2^j}$. This shows that (4.21) holds. Property (19) is straightforward. □

4.5.2 1-D Piecewise Smooth Approximation

For 1-D signal in the piecewise regular model (4.9), large wavelets coefficients $\langle f, \psi_{j,n}\rangle$ are clustered around the singularities of the signal. We call $\mathcal{S} \subset [0, 1]$ the finite set of singular points.

Theorem 5. *f is in the piecewise smooth signal model (4.9), the non-linear approximation error in wavelet obeys*

$$\|f - f_M\|^2 = O(M^{-2\alpha}). \tag{4.22}$$

Proof. The proof is split in several parts.

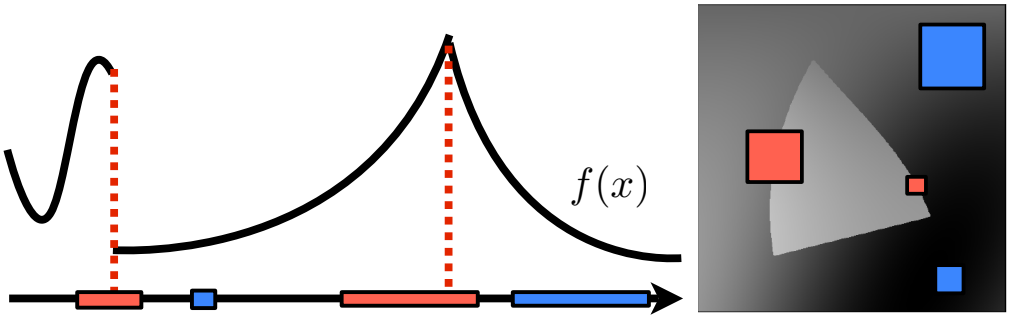


Figure 4.15: Singular par of signals (left) and image (right).

Step 1. Coefficient segmentation. The singular support at scale 2^j is the set of coefficients corresponding to wavelets that are crossing a singularity

$$\mathcal{C}_j = \{n \,;\, \mathrm{supp}(\psi_{j,n}) \cap \mathcal{S} \neq \emptyset\} \tag{4.23}$$

It has a constant size because of the dyadic translation of wavelets

$$|\mathcal{C}_j| \leqslant K|\mathcal{S}| = \text{constant}.$$

Using (4.21) for $d = 1$, the decay of regular coefficients is bounded as

$$\forall\, n \in \mathcal{C}_j^c, \quad |\langle f,\, \psi_{j,n}\rangle| \leqslant C 2^{j(\alpha+1/2)}.$$

Using (19) for $d = 1$, the decay of singular coefficients is bounded as

$$\forall\, n \in \mathcal{C}_j, \quad |\langle f,\, \psi_{j,n}\rangle| \leqslant C 2^{j/2}.$$

Once a fixed threshold T is fixed to compute the non-linear approximation, one defines cut-off scales for regular and singular coefficients that depend on T

$$2^{j_1} = (T/C)^{\frac{1}{\alpha+1/2}} \quad \text{and} \quad 2^{j_2} = (T/C)^2.$$

Figure 4.16 shows a schematic segmentation of the set of wavelet coefficients into regular and singular parts, and also using the cut-off scales.

Step 2. Counting the error. These cut-off scales allow us to define a hand-crafted approximation signal

$$\tilde{f}_M = \sum_{j \geqslant j_2} \sum_{n \in \mathcal{C}_j} \langle f,\, \psi_{j,n}\rangle \psi_{j,n} + \sum_{j \geqslant j_1} \sum_{n \in \mathcal{C}_j^c} \langle f,\, \psi_{j,n}\rangle \psi_{j,n}. \tag{4.24}$$

The approximation error generated by this M-term approximation $\tilde{f}_M$ is larger than the best M-term approximation error, and hence

$$\|f - f_M\|^2 \leqslant \|f - \tilde{f}_M\|^2 \leqslant \sum_{j<j_2, n \in \mathcal{C}_j} |\langle f,\, \psi_{j,n}\rangle|^2 + \sum_{j<j_1, n \in \mathcal{C}_j^c} |\langle f,\, \psi_{j,n}\rangle|^2 \tag{4.25}$$

$$\leqslant \sum_{j<j_2} (K|\mathcal{S}|) \times C^2 2^j + \sum_{j<j_1} 2^{-j} \times C^2 2^{j(2\alpha+1)} \tag{4.26}$$

$$= O(2^{j_2} + 2^{2\alpha j_1}) = O(T^2 + T^{\frac{2\alpha}{\alpha+1/2}}) = O(T^{\frac{2\alpha}{\alpha+1/2}}). \tag{4.27}$$

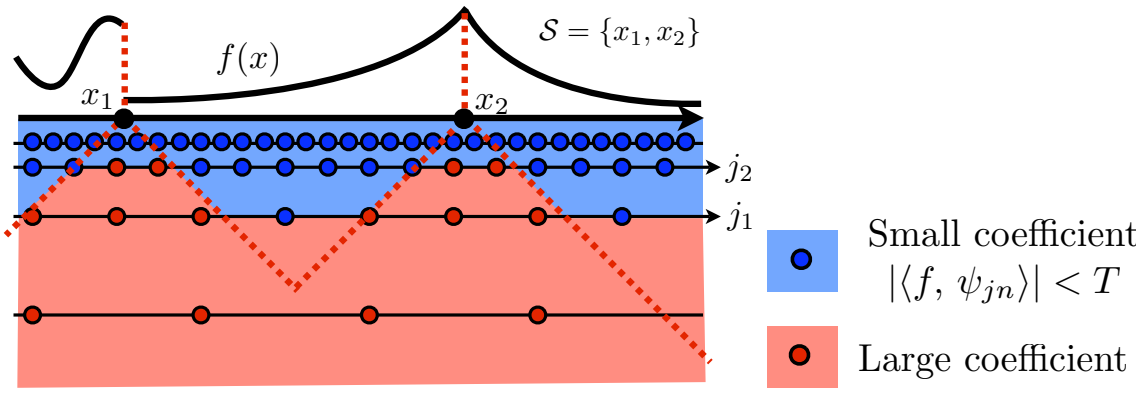


Figure 4.16: Segmentation of the wavelet coefficients into regular and singular parts.

Step 3. Counting the number of measurements. The number of coefficients needed to build the approximating signal $\tilde{f}_M$ is

$$M \leqslant \sum_{j \geqslant j_2} |\mathcal{C}_j| + \sum_{j \geqslant j_1} |\mathcal{C}_j^c| \leqslant \sum_{j \geqslant j_2} K|\mathcal{S}| + \sum_{j \geqslant j_1} 2^{-j} \tag{4.28}$$

$$= O(|\log(T)| + T^{\frac{-1}{\alpha+1/2}}) = O(T^{\frac{-1}{\alpha+1/2}}). \tag{4.29}$$

Step 3. Putting everything together. Putting equations (4.25) and (4.28) together gives the desired result. □

This theorem improves significantly over the $O(M^{-1})$ decay of Fourier approximation (4.19). Furthermore, this decay is the same as the error decay of uniformly smooth signal (4.16). In 1-D, wavelet approximations do not "see" the singularities. The error decay (4.22) can be shown to be asymptotically optimal.

Figure 4.17 shows examples of wavelet approximation of singular signals.

4.5.3 2-D Piecewise Smooth Approximation

We now give the equivalent of Theorem 5 but for 2-D functions.

Theorem 6. *f is in the piecewise smooth signal model (4.10), the non-linear approximation error in wavelet obeys*

$$\|f - f_M\|^2 = O(M^{-1}). \tag{4.30}$$

Proof. For an image in the piecewise smooth model (4.10), we define the singular support $\mathcal{C}_j$ as in (4.23). The major difference with the 1-D setting, is that for 2-D images, the size of the singular support grows when the scale 2^j goes to zero

$$|\mathcal{C}_j^\omega| \leqslant 2^{-j} K|\mathcal{S}|,$$

where $|\mathcal{S}|$ is the perimeter of the singular curves $\mathcal{S}$, and $\omega \in \{V, H, D\}$ is the wavelet orientation. Using (4.21) for $d = 2$, the decay of regular coefficients is bounded as

$$\forall\, n \in (\mathcal{C}_j^\omega)^c, \quad \left|\langle f, \psi_{j,n}^\omega\rangle\right| \leqslant C 2^{j(\alpha+1)}.$$

Using (19) for $d = 2$, the decay of singular coefficients is bounded as

$$\forall\, n \in \mathcal{C}_j^\omega, \quad \left|\langle f, \psi_{j,n}^\omega\rangle\right| \leqslant C 2^j.$$

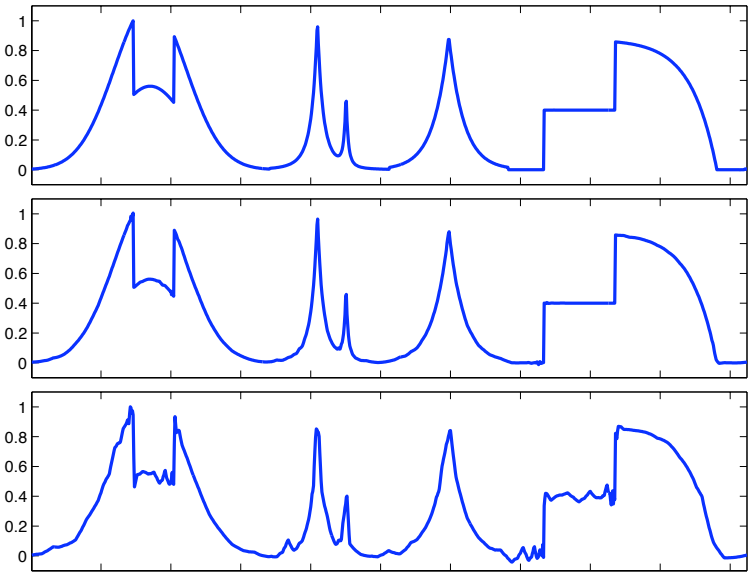


Figure 4.17: 1-D wavelet approximation.

After fixing T, the cut-off scales are defined as

$$2^{j_1} = (T/C)^{\frac{1}{\alpha+1}} \quad \text{and} \quad 2^{j_2} = T/C.$$

We define similarly to (4.24) a hand-made approximation. Similarly to (4.25), we bound the approximation error as

$$\|f - f_M\|^2 \leqslant \|f - \tilde{f}_M\|^2 = O(2^{j_2} + 2^{2\alpha j_1}) = O(T + T^{\frac{2\alpha}{\alpha+1}}) = O(T)$$

and the number of coefficients as

$$M = O(T^{-1} + T^{\frac{-1}{\alpha+1}}) = O(T^{-1}).$$

This leads to the announced decay of the non-linear wavelet approximation error. □

This improves significantly over the $O(M^{-1/2})$ decay of Fourier approximation (4.20). This result is however deceiving, since it does not take advantage of the C^α regularity of the image outside the edge curves.

This error decay is still valid for the more general model of images with bounded variations (4.11). One can shows that wavelets are asymptotically optimal to approximate images with bounded variations.

Figure 4.18 shows wavelet approximations of a bounded variation image.

4.6 Cartoon Images Approximation

The square support of wavelet makes them inefficient to approximate geometric images (4.13), whose edges are more regular than the level set of bounded variation images (4.11), which are only assumed to be of finite length.

4.6.1 Wavelet Approximation of Cartoon Images

Result (4.30) shows that wavelet approximation of images in the cartoon models (4.13) decays at least like $O(M^{-1})$. One can show that simple cartoon images like $f = 1_\Omega$ where Ω is a disk reach this low decay

Figure 4.18: 2-D wavelet approximation.

speed. This is because the square support of wavelets forbid them to take advantage of the regularity of edge curves. The approximation error for the smoothed cartoon model (4.14) is also slow if the width of the blurring kernel is small with respect to the number M of coefficients.

Figure 4.19 shows that many large coefficients are located near edge curves, and retaining only a small number leads to a bad approximation with visually unpleasant artifacts.

Figure 4.19: Wavelet approximation of a cartoon image.

4.6.2 Finite Element Approximation

To improve over the wavelet approximation, one can design an adapted triangulation that is highly anisotropic near edges. Figure 4.20 shows an example of such a triangulation.

A triangulation is obtain by sampling M points over the image domain $[0,1]^2$ and then connecting them using triangles. One then defines a piecewise linear interpolation $\tilde{f}_M$ over these triangles.

As shown in Figure 4.21, an efficient approximation of a C^2-cartoon image (4.13) for $\alpha = 2$ is obtained by seeding $\approx M/2$ approximately equilateral triangles of width $\approx M^{-1/2}$ in the areas where the image is regular. Near the edges, using the C^2 regularity of the singular curve, one can seed $\approx M/2$ anisotropic triangles of length M^{-1} and width $\approx M^{-1/2}$. One can show that such an adaptive triangulation leads to an

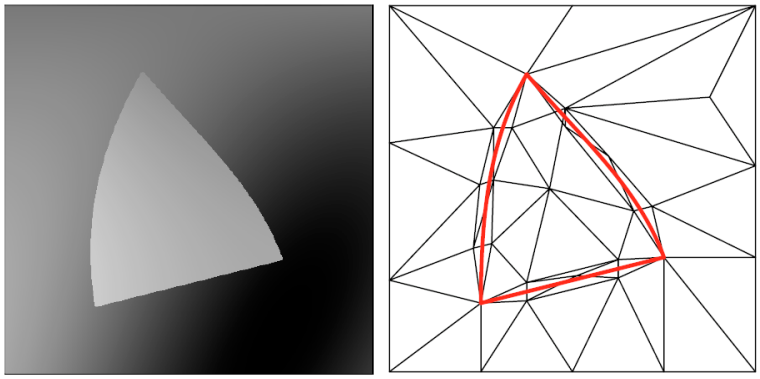


Figure 4.20: Left: cartoon image, right: adaptive triangulation.

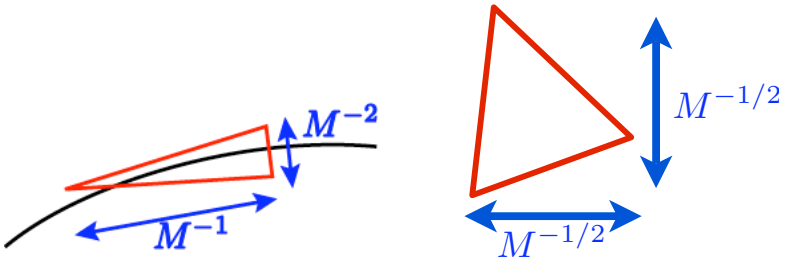


Figure 4.21: Aspect ratio of triangle away from edges (left) and near an edge (right).

approximation error

$$\|f - f_M\|^2 = O(M^{-2}), \tag{4.31}$$

which improves over the wavelet approximation error decay (4.30).

This scheme is however difficult to implement in practice, since the edge curves are not known and difficult to find. This is in particular the case for smooth cartoon images when the smoothing kernel h is unknown.

There is currrently no known algorithm that can automatically produces the error decay (4.31). One thus has to use heuristics and greedy algorithm to find the location of the sampling points and computes the triangles. Figure (4.22) shows an example of compression using such a greedy seeding algorithm, that works well in practice.

4.6.3 Curvelets Approximation

Instead of using an adaptive approximation scheme such as finite elements, one can replace the wavelet basis by a set of oriented anisotropic atoms. The curvelet frame was proposed by Candès and Donoho for this purpose [4].

Curvelets. The curvelet construction starts from a curvelet function c that is oriented along the horizontal direction, and perform stretching

$$c_{2^j}(x_1, x_2) \approx 2^{-3j/4} c(2^{-j/2} x_1, 2^{-j} x_2),$$

Adpated triangulation $\tilde{f}_M$ JPEG-2000

Figure 4.22: Comparison of adaptive triangulation and JPEG-2000, with the same number of bits.

translation and rotation

$$c^{\theta}_{2^j,u}(x_1,x_2) = c_{2^j}(R_\theta(x-u))$$

where R_θ is the rotation of angle θ.

The atoms $c^{\theta}_{2^j,u}$ is located near u, with an orientation θ, and has an aspect ratio "width $\approx$ length^2", which is the same aspect used to build an adaptive finite element approximation. This aspect ratio is essential to capture the anisotropic regularity near edges for images in the cartoon model (4.13) for $\alpha = 2$.

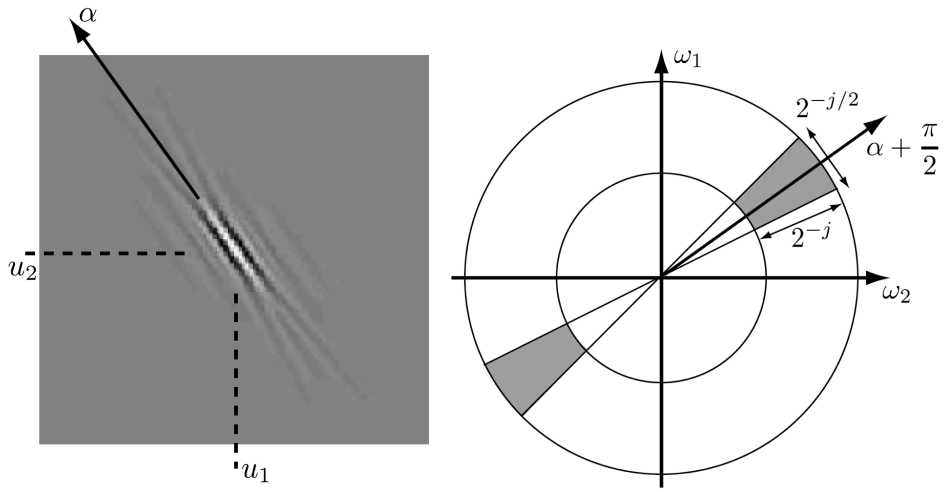


Figure 4.23: Left: a curvelet c_m, right: its Fourier transform localization.

Figure 4.24 shows the spacial and frequency localization of curvelets.

Parameter discretization. To build an image representation, one need to sample the u and θ parameter. To maintain a stable representation, the sub-sampling of the angles depends on the scale

$$\forall\, 0 \leqslant k < 2^{-\lceil j/2\rceil+2}, \quad \theta_k^{(j)} = k\pi 2^{\lceil j/2\rceil-1}$$

and the spacial grid depends on the scale and on the angles

$$\forall\, m = (m_1, m_2) \in \mathbb{Z}^2, \quad u_m^{(j,\theta)} = R_\theta(2^{j/2} m_1, 2^j m_2).$$

Figure 4.24 shows this sampling pattern.

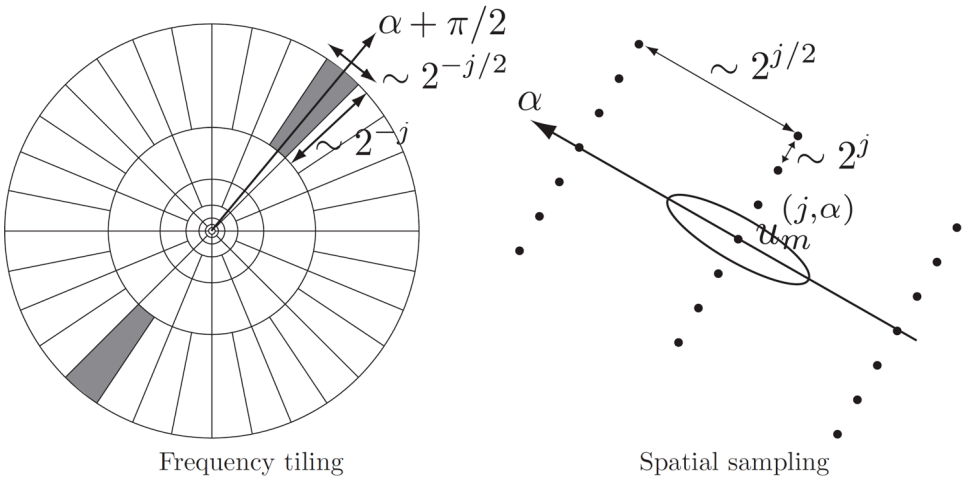


Figure 4.24: Sampling pattern for the curvelet positions.

Curvelet tight frame. This sampling leads to a stable redundant family

$$C_{j,m,k}(x) = c^{\theta}_{2^j,u}(x) \quad \text{where} \quad \theta = \theta_k^{(j)} \quad \text{and} \quad u = u_m^{(j,\theta)},$$

that obeys a conservation of energy

$$\|f\|^2 = \sum_{j\in\mathbb{Z}} \sum_{k=0}^{2^{-\lceil j/2\rceil+2}} \sum_{m\in\mathbb{Z}^2} |\langle f,\, C_{j,m,k}\rangle|^2$$

and a reconstruction formula

$$f = \sum_{j\in\mathbb{Z}} \sum_{k=0}^{2^{-\lceil j/2\rceil+2}} \sum_{m\in\mathbb{Z}^2} \langle f,\, C_{j,m,k}\rangle C_{j,m,k}$$

that extends the properties of orthogonal basis (tight frame), although the representation is redundant (the atoms are not orthogonal).

A numerical implementation of this tight frame also defines a discrete tight frame for image of N pixels, that is made of $\approx 5N$ atoms [5].

Curvelet approximation. A non-linear M-term approximation in curvelets is defined as

$$f_M = \sum_{|\langle f,\, C_{j,m,k}\rangle| > T} \langle f,\, C_{j,m,k}\rangle C_{j,m,k}$$

where T is a threshold that depends on M. One should note that f_M is not necessarily the best M-term curvelet approximation since the curvelet frame is not orthogonal.

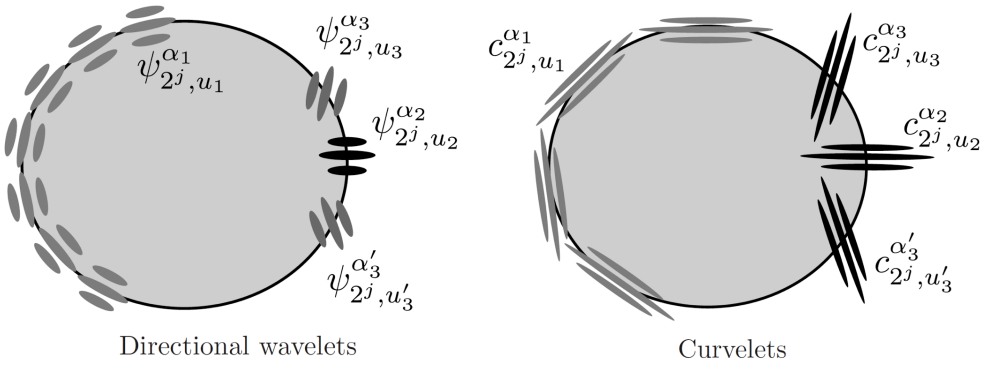


Figure 4.25: Comparison of the principle of wavelets (left) and curvelet (right) approximations of a cartoon image.

For position $u_m^{(j,\theta)}$ that are far away from an edges, the vanishing moments of the curvelets create a small coefficient $\langle f, C_{j,m,k}\rangle$. If $u_m^{(j,\theta)}$ is close to an edge curve whose tangent has direction $\tilde{\theta}$, then the coefficient $\langle f, C_{j,m,k}\rangle$ decays very fast to zero when $|\theta - \tilde{\theta}|$ increases. Figure 4.25 shows the principle of this curvelet approximation, and compares it with directional wavelets that have a square support.

Using these two properties together with the sparse sampling of the curvelet in space and orientation leads to the following approximation error decay

$$\|f - f_M\|^2 = O(\log^3(M) M^{-2})$$

for image in the cartoon model (4.13) for $\alpha = 2$. This is close to the decay of adaptive triangulations (4.31), but this time one computes f_M with a fast $O(N \log(N))$ algorithm for an image of N pixels.

In practice, the redundancy of the curvelet frame makes it not suitable for image compression. Its efficiency is however useful for denoising purpose, where it can improve over wavelet to denoise geometric images and textures, see Figure 4.26. The result is obtained by using a thresholding denoiser as detailed in Section 6.3.1.

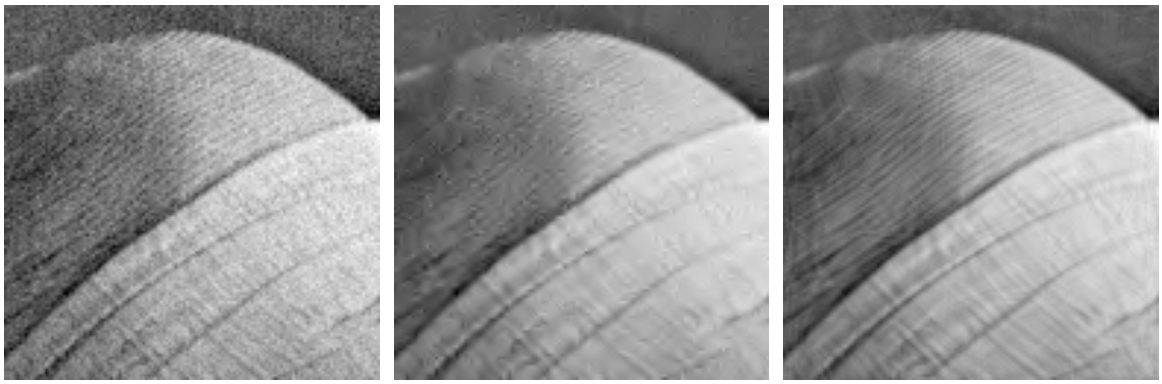


Figure 4.26: Comparison of wavelets (translation invariant) and curvelet denoising.

Chapter 5

Compression

5.1 Transform Coding

5.1.1 Coding

State of the art compression schemes correspond to transform coders, that code quantized coefficients in an ortho-basis. They first computes the coefficients of the decomposition of the signal into an well-chosen basis (for instance wavelets)

$$a_m = \langle f, \psi_m \rangle \in \mathbb{R}.$$

Quantization corresponds to rounding the coefficients to an integer using a step size $T > 0$

$$q_m = Q_T(a_m) \in \mathbb{Z} \quad \text{where} \quad Q_T(x) = \text{sign}(x) \left\lfloor \frac{|x|}{T} \right\rfloor.$$

We note that this quantizer has a twice larger zero bin, so that coefficients in $[-T, T]$ are set to zero.

This quantizer nonlinearity should be compared to the hard thresholding nonlinearity (9.4) used to perform non-linear approximation. The quantizer not only set to zero small coefficients that are smaller than T in magnitude, it also modifies larger coefficients by rounding, see Figure 5.1.

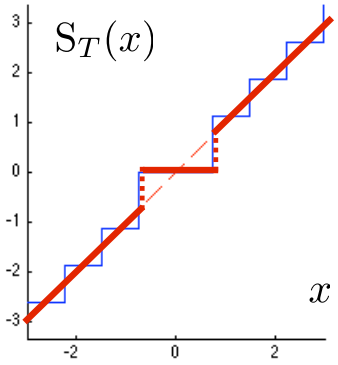


Figure 5.1: Thresholding and quantization non-linearity mappings.

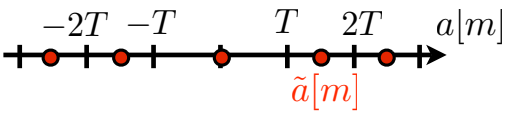
The resulting integer values $(q_m)_m$ are stored into a binary file of length R, which corresponds to a number of bits. Sections 5.1.3 and 5.2 detail two different approach to perform this transformation from integer to bits. The goal is to reduce as much as possible the number R of bits.

5.1.2 De-coding

The decoder retrieves the quantized coefficients q_m from the binary file, and dequantizes the coefficients using

$$\tilde{a}_m = \mathrm{sign}(q_m)\left(|q_m| + \frac{1}{2}\right)T. \tag{5.1}$$

This corresponds to retrieving the value from quantization at the center of quantization bins:



The compressed-decompressed image is then reconstructed as

$$\mathcal{Q}_T(f) \stackrel{\text{def.}}{=} \sum_{m \in I_T} \tilde{a}_m \psi_m = \sum_{m \in I_T} Q_T(\langle f, \psi_m \rangle)\psi_m,$$

thus producing a decompression error $\|f - \mathcal{Q}_T(f)\|$.

This decompression reconstruction (5.1.2) should be compared with the non-linear approximation formula (4.3). One necessarily has $\|f - f_M\| \leqslant \|f - \mathcal{Q}_T(f)\|$, but in practice these two errors have comparable magnitudes.

Proposition 20. *One has*

$$\|f - \mathcal{Q}_T(f)\|^2 \leqslant \|f - f_M\|^2 + MT^2/4 \quad \textit{where} \quad M = \#\left\{m \,;\, \tilde{a}_m \neq 0\right\}. \tag{5.2}$$

Proof. Indeed, the de-quantization formula (5.1) implies that for $|a_m| > T$,

$$|a_m - \tilde{a}_m| \leqslant \frac{T}{2}.$$

One thus has

$$\|f - \mathcal{Q}_T(f)\|^2 = \sum_m (a_m - \tilde{a}_m)^2 \leqslant \sum_{|a_m| < T} |a_m|^2 + \sum_{|a_m| \geqslant T} \left(\frac{T}{2}\right)^2,$$

which leads to the desired bound. □

5.1.3 Support Coding

To measure how large is the additional error term $MT^2/4$ in (5.2), one needs to choose a method to store the quantized coefficients q_m into a file.

For aggressive compression scenario, where R and M are small with respect to the size N of the image, the support

$$I_M = \{m \,;\, \tilde{a}_m \neq 0\}$$

is highly sparse. It thus make sense to code first this support and then to code the actual value $q_m \neq 0$ for $m \in I_M$.

The remaining of this section proves the following theorem.

Theorem 7. *We assume $\|f - f_M\|^2 \sim M^{-\alpha}$ where f_M is the M-term non-linear approximation of f in $\{\psi_m\}_m$. We also assume that the required number of discrete samples N used can be bounded polynomially with N (see (5.7) below for more details). Then for all T there exists a coding strategy of $\mathcal{Q}_T(f)$ using $R = R(T)$ bits such that*

$$\|f - \mathcal{Q}_T(f)\|^2 = O(\log^\alpha(R) R^{-\alpha}).$$

Proof. This proof is split in several parts.

Signals constraints. First, let us notice that, thanks to Proposition 16 the error decay hypothesis $\|f - f_M\|^2 \sim M^{-\alpha}$ is equivalent to imposing a fast decay of the ordered coefficients d_m defined in (4.1) satisfies

$$d_m \sim m^{-\frac{\alpha+1}{2}} \quad \Longrightarrow \quad T \sim M^{-\frac{\alpha+1}{2}}. \tag{5.3}$$

Thanks to Proposition 20, one thus has

$$\|f - f_M\|^2 \leqslant \|f - \mathcal{Q}_T(f)\|^2 \leqslant \|f - f_M\|^2 + MT^2/4 \sim M^{-\alpha} \sim \|f - f_M\|^2, \tag{5.4}$$

which shows that the compression error is comparable with the approximation error.

Discrete computation and scaling of N. For the compression from the discrete signals to be the same as a compression of a continuous signal, we impose that N is large enough so that

$$\forall\, m \geqslant N, \quad |\langle f, \psi_m \rangle| < T \tag{5.5}$$

so that the coefficients not quantized to 0 are contained within the set $\{\langle f, \psi_m \rangle\}_{0 \leqslant m < N}$ of the N computed coefficients. For instance, if f is bounded and one considers a wavelet basis, (19) ensures that $|\langle f, \psi_m \rangle| = O(2^{-jd/2})$, and thus choosing $T \sim 2^{-j_{\max} d/2}$ ensures (5.5) where $N = 2^{-j_{\max} d}$, i.e.

$$N = O(1/T^2). \tag{5.6}$$

The other hypothesis beyond (5.3) of Theorem 7 is that the number N of required discrete samples is not too large, and in particular, that there is a polynomial grows of N with respect to the number M of coefficients to code

$$N = O(M^\beta) \tag{5.7}$$

for some $\beta > 0$. For instance, under the decay condition (5.3) and the worse case scaling (5.6) for bounded function in a wavelet basis,

$$N = O(1/T^2) \quad \text{and} \quad T \sim M^{-\frac{\alpha+1}{2}} \quad \Longrightarrow \quad N = O(M^{\alpha+1}).$$

Support coding. One can simply encode the M indexes of each element $m \in I_M \subset \{1, \ldots, N\}$ using $\log_2(N)$ bits, so that the total number of bits for the support I_M is

$$R_{\text{ind}} = M \log_2(N) = O(M \log_2(M)) \tag{5.8}$$

where we used (5.3).

Values coding. The quantized values satisfy $q_m \in \{-A, \ldots, A\}$, with

$$A \leqslant \frac{1}{T} \max_m |\langle f, \psi_m \rangle| = O(T^{-1}),$$

so one can code them using a number of bits

$$R_{\text{val}} = O(M |\log_2(T)|) = O(M \log_2(M)) \tag{5.9}$$

where we have used hypothesis (5.3) that implies $|\log_2(T)| \sim \log_2(M)$.

Total number of bits. Putting (5.8) and (5.9) together, the total number of bits for this support coding approach is thus

$$R = R_{\text{ind}} + R_{\text{val}} = O(M \log_2(M)). \tag{5.10}$$

The function $\varphi(m) = m \log_2(m)$ is strictly increasing, one can invert it, and we now show that $\varphi^{-1}(r) = r/\log(r) + o(1)$ for large r. Indeed, writing $r = \varphi(m)$

$$m = \frac{r}{\log_2(m)} = \frac{r}{\log_2(r) - \log_2 \log_2(m)} \sim \frac{r}{\log_2(r)}$$

where we used the fact that since $m \leqslant r$, $\log_2 \log_2(m) = o(\log_2(r))$. Inverting this relationship (5.10) thus proves that

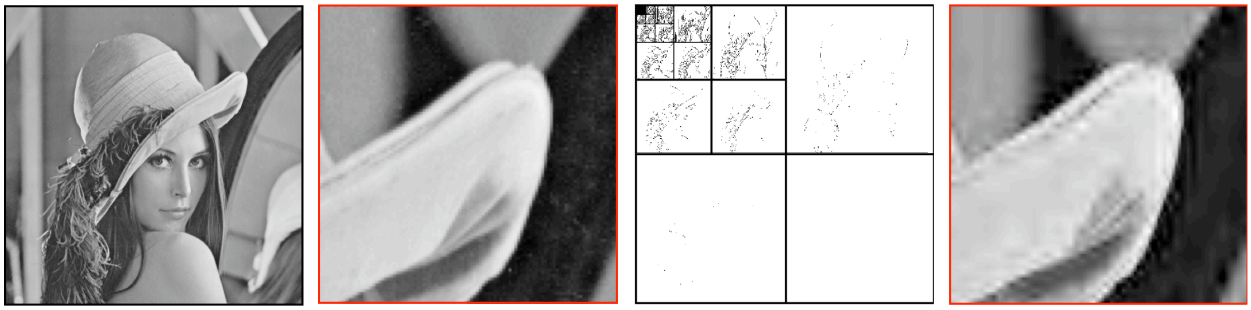
$$M \geqslant C \frac{R}{\log_2(R)} \tag{5.11}$$

for some constant C. Using (5.4) and (5.11), one thus finally arrives to

$$\|f - \mathcal{Q}_T(f)\|^2 \sim M^{-\alpha} = O(\log^{\alpha}(R) R^{-\alpha}).$$

□

This theorem shows the importance of the study of non-linear approximation, and in particular the design of bases that are efficient for approximation of a given signal model Θ. Let us also insists on the fact that a practical compression algorithm is only capable of dealing with discrete signals of size N. We thus considers that the algorithm has access to N inner products $\{\langle f, \psi_m \rangle\}_{1 \leqslant m < N}$ that are computed using a decomposition algorithm from a discretized signal or image of size N. For instance, Section **??** details a discrete wavelet transform, and introduces a compatibility condition (**??**) on the sampling operator for this inner product computation to be possible from discrete data.



Original f | Zoom | Wavelet support | Compressed $\mathcal{Q}_T(f)$

Figure 5.2: Image compression using wavelet support codding.

5.2 Entropic Coding

To further reduce the file size R (in bits), one can use an entropic coder to transform the integer values q_m into bits. Such coding scheme makes use of the statistical redundancy of the quantized values, that have a large number of zero entries and very few large entries. The theory of coding was formalized by Shannon [24].

We refer to Section 1.3 for the theoretical foundation associated to what we now describe.

Probabilistic modeling. The quantized coefficients $q_m \in \{-A, \ldots, A\}$ are assumed to take values in an alphabet of $Q = 2A + 1$ elements. A coding scheme performs the transformation

$$\{q_m\}_m \longmapsto \{0, 1, 1, \ldots, 0, 1\} \in \{0, 1\}^R.$$

To reduce the average value of R, one makes use of a statistical model, which assumes that the q_m are drawn independently at random from a known probability distribution

$$\mathbb{P}(q_m = i) = p_i \in [0, 1].$$

Huffman code. A Huffman code is a code with variable length, since it perform a mapping from symbols to binary strings

$$q_m = i \in \{-A, \ldots, A\} \longmapsto c_i \in \{0, 1\}^{|c_i|}$$

where $|c_i|$ is the length of the binary code word c_i, that should be larger if p_i is small. A Huffman tree algorithm is able to build a code such that

$$|c_i| \leqslant \lceil \log_2(p_i) \rceil$$

so that

$$R \leqslant (\mathcal{E}(p) + 1)N$$

where $\mathcal{E}$ is the entropy of the distribution, defined as

$$\mathcal{E}(p) = -\sum_i p_i \log_2(p_i).$$

Figure 5.3 shows different probability distribution. The entropy is small for highly sparse distribution. Wavelet coefficients of natural images tend to have a low entropy because many coefficients are small.

The Huffman scheme codes symbols independently, leading to a sub-optimal code if some of the p_i are large, which is usually the case for wavelet coefficients. One usually prefers arithmetic coding schemes, that codes groups of symbols, and are able to get close to the entropy bound $R \approx \mathcal{E}(p)N$ for large N.

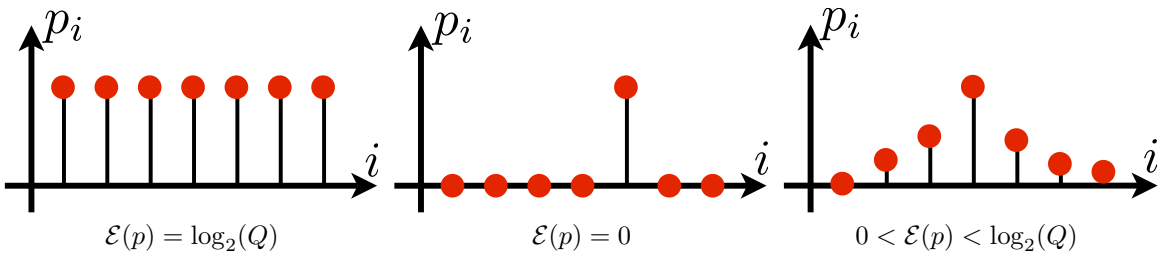


Figure 5.3: Three different probability distributions.

5.3 JPEG-2000

JPEG-2000 is the latest still image compression standard. It corresponds to a wavelet transform coder that performs a clever adaptive entropy coding that makes use of the statistical redundancy of wavelet coefficients of natural images. The wavelet transform is not orthogonal, it is a symmetric 7/9 biorthogonal, with symmetric boundary condition and a lifting implementation. This transform is however close to orthogonality, so that the previous discussion about orthogonal approximation and coding is still relevant.

Figure 5.4 shows an overview of JPEG-2000 architecture. Figure 5.5 shows a comparison between JPEG and JPEG-2000 compressors. JPEG is based on a local DCT transform, and suffers from blocking artifacts at low bit rates, which is not the case of JPEG-2000. This new standard also comes with several important features, such as regions of interest, which allows to refine the coding in some specific parts of the image.

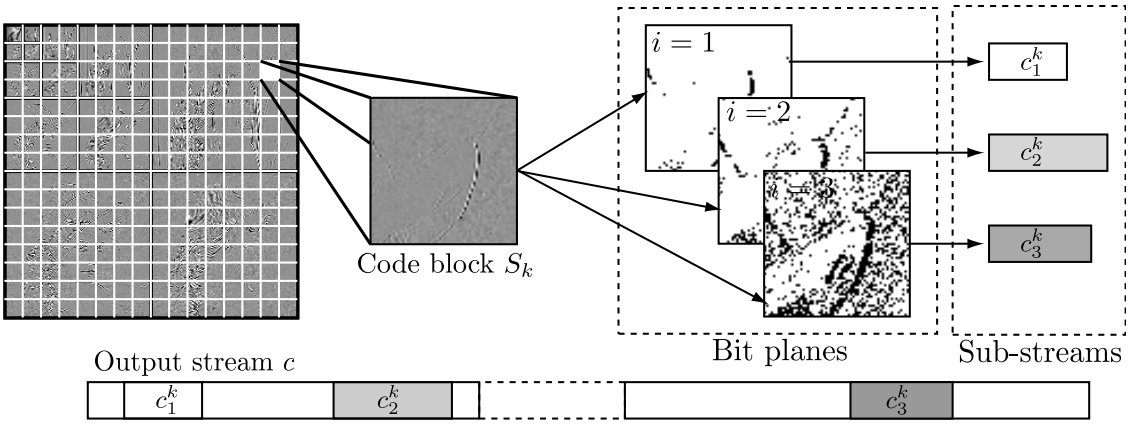


Figure 5.4: JPEG-2000 coding architecture.

Dyadic quantization. The wavelet coefficients are quantized with a varying quantization step $T_i = 2^{-i}T_0$. This allows one to progressively increase the precision of the coded coefficients. For each i, a bit plane coding pass produces new bits to refine the value of the coefficients when i increases.

Steam packing. The bits obtained using the bit plane pass with quantizer $T_i = 2^{-i}T_0$ are entropy coded using a contextual coder. This coder processes square blocks S_k of coefficients. This local coding enhances the parallelization of the method. This local block coding produces a bit stream c_i^k, and these streams are optimally packed into the final coded file to reduce the distortion $\|f - \mathcal{Q}_T(f)\|$ for almost every possible number R of bits. This stream packing ensures scalability of the output bit stream. It means that one can receive only the R first bits of a large coded file and get a low resolution decoded image $\mathcal{Q}_T(f)$ that has an almost minimal distortion $\|f - \mathcal{Q}_T(f)\|$.

Bit plane coding pass. For each threshold T_i, for each scale and orientation $(j, \omega \in \{V, H, D\})$, for each coefficient location $n \in S_k$, JPEG-2000 coder encodes several bit reflecting the value of the wavelet coefficient $d_j^\omega[n]$. In the following we drop the dependancy on (j, ω) for simplicity.

– If $d_j^\omega[n] < T_{i-1}$, the coefficient was not significant at bit-plane $i-1$. It thus encodes a significance bit $b_i^1[n]$ to tell wether $d_j^\omega[n] \geqslant T_i$ or not.

– If $b_i^1[n] = 1$, meaning that the coefficient has became significant, it codes it sign as a bit $b_i^2[n]$.

– For every position n that was previously significant, meaning $d_j^\omega[n] \geqslant T_{i-1}$, it codes a value refinement bit $b_i^3[n]$ to tell wether $d_j^\omega[n] \geqslant T_i$ or not.

Contextual coder. The final bits streams c_i^k are computed from the produced bits $\{b_i^s[n]\}_{s=1}^3$ for $n \in S_k$ using a contextual coder. The contextual coding makes use of spacial redundancies in wavelet coefficients, especially near edges and geometric singularities that create clusters of large coefficients. The coefficients $n \in S_k$ are traversed in zig-zag order as shown on Figure 5.6.

For each coefficient location $n \in S_k$, the context value $v_i^s[n]$ of the bit $b_i^s[n]$ to code at position x is an integer computed over a 3×3 window

$$w_n = \{(n_1 + \varepsilon_1, n_2 + \varepsilon_2)\}_{\varepsilon_i = \pm 1}.$$

This local context $v_i^s[n]$ integrates in a complicated way the previous bit plane values $\{b_{i-1}^s[\tilde{n}]\}_{\tilde{n} \in w_n}$, and neighboring bits at plane $\{b_i^s[\tilde{n}]\}_{\tilde{n} \in w_n, \tilde{n} \text{ coded}}$ that have already been coded.

The bit value $b_i^s[n]$ is then coded with an arithmetic coding by making use of the conditional probability distribution $\mathbb{P}(b_i^s[n] | v_i^s[n])$. The choice made for the computation $v_i^s[n]$ allows to reduce significantly the

Figure 5.5: Comparison of JPEG (left) and JPEG-2000 (right) coding.

entropy of this conditional probability condition with respect to the original distribution $\mathbb{P}(b_i^s[n])$, thus reducing the overall number of bits.

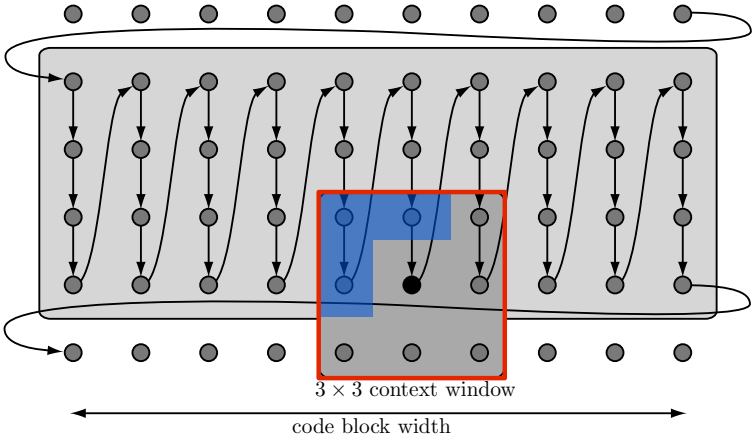


Figure 5.6: Coding order and context for JPEG-2000 coding.

Chapter 6

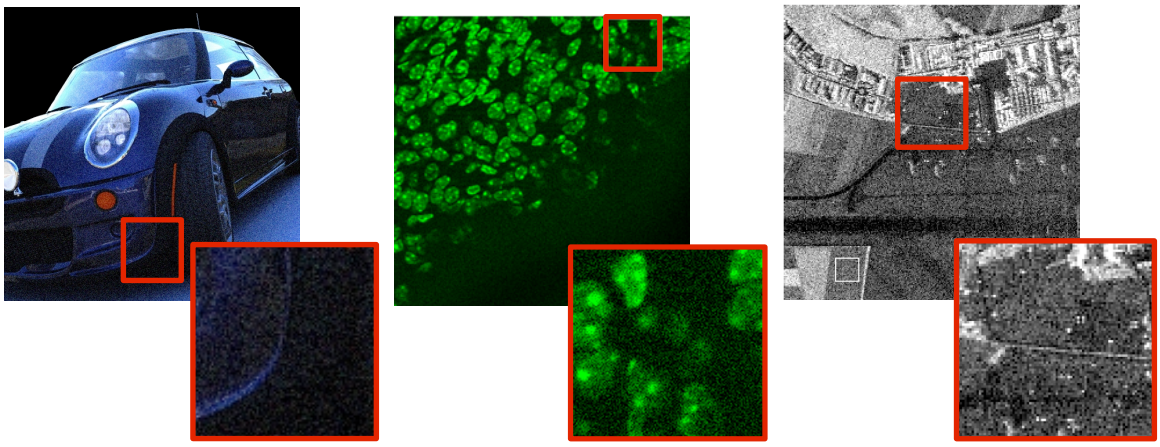
Denoising

Together with compression, denoising is the most important processing application, that is pervasive in almost any signal or image processing pipeline. Indeed, data acquisition always comes with some kind of noise, so modeling this noise and removing it efficiently is crucial.

6.1 Noise Modeling

6.1.1 Noise in Images

Image acquisition devices always produce some noise. Figure 6.1 shows images produced by different hardware, where the regularity of the underlying signal and the statistics of the noise is very different.



Digital camera Confocal imaging SAR imaging

Figure 6.1: Example of noise in different imaging device.

One should thus model both the acquisition process and the statistics of the noise to fit the imaging process. Then one should also model the regularity and geometry of the clean signal to choose a basis adapted to its representation. This chapter describes how thresholding methods can be used to perform denoising in some specific situations where the noise statistics are close to being Gaussian and the mixing operator is a sum or can be approximated by a sum.

Since noise perturbs discrete measurements acquired by some hardware, in the following, we consider only finite dimensional signal $f \in \mathbb{C}^N$.

6.1.2 Image Formation

Figure 6.2 shows an idealized view of the image formation process, that mixes a clean image f_0 with a noise w to obtain noisy observations $f = f_0 \oplus w$, where $\oplus$ might for instance be a sum or a multiplication.

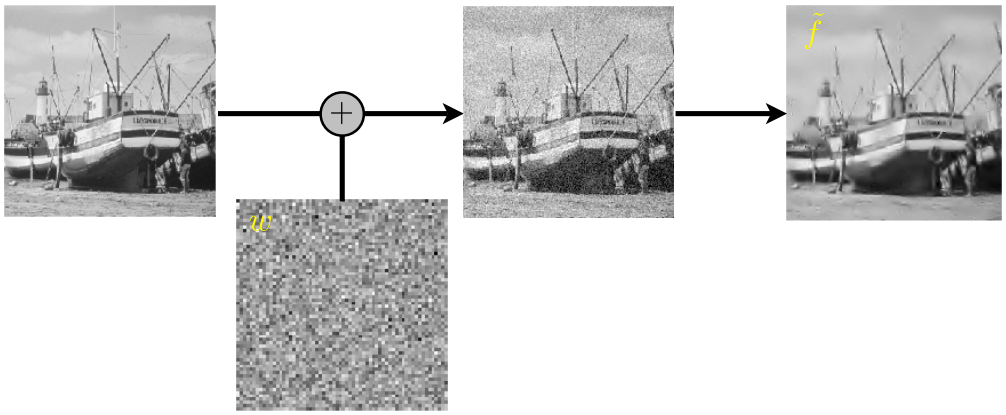


Figure 6.2: Image formation with noise modeling and denoising pipepline.

Statistical modeling considers w as a random vector with known distribution, while numerical computation are usually done on a single realization of this random vector, still denoted as w.

Additive Noise. The simplest model for such image formation consists in assuming that it is an additive perturbation of a clean signal f_0

$$f = f_0 + w$$

where w is the noise residual. Statistical noise modeling assume that w is a random vector, and in practice one only observes a realization of this vector. This modeling thus implies that the image f to be processed is also a random vector. Figure 6.3 and 6.4 show examples of noise addition to a clean signal and a clean image.

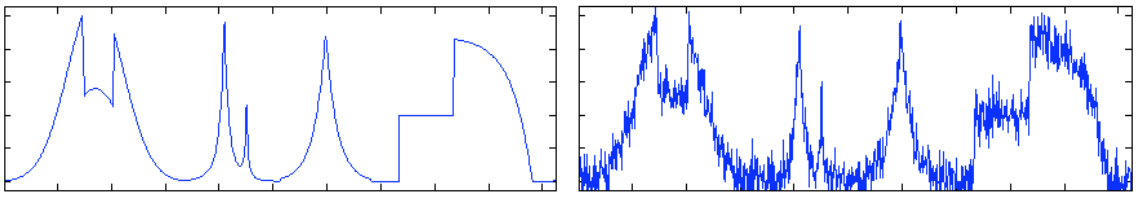


Figure 6.3: 1-D additive noise example.

The simplest noise model assumes that each entry w_n of the noise is a Gaussian random variable of variance σ^2, and that the w_n are independent, i.e. $w \sim \mathcal{N}(0, \mathrm{Id}_N)$. This is the white noise model.

Depending on the image acquisition device, one should consider different noise distributions, such as for instance uniform noise $w_n \in [-a, a]$ or Impulse noise

$$\mathbb{P}(w_n = x) \propto e^{-|x/\sigma|^\alpha} \quad \text{where} \quad \alpha < 2$$

In many situations, the noise perturbation is not additive, and for instance its intensity might depend on the intensity of the signal. This is the case with Poisson and multiplicative noises considered in Section 6.4.

f_0 f

Figure 6.4: 2-D additive noise example.

6.1.3 Denoiser

A denoiser (also called estimator) is an estimation $\tilde{f}$ of f_0 computed from the observation f alone. It is thus also a random vector that depends on the noise w. Since f is a random vector of mean f_0, the numerical denoising process corresponds to the estimation of the mean of a random vector from a single realization. Figure 6.5 shows an example of denoising.

The quality of a denoiser is measured using the average mean square risk $\mathbb{E}_w(\|f_0 - \tilde{f}\|^2)$. where $\mathbb{E}_w$ is the esperance (averaging) with respect to the noise w. Since f_0 is unknown, this corresponds to a theoretical measure of performance, that is bounded using a mathematical analysis. In the numerical experiments, one observes a single realization $f^r \sim f_0 + w$, and the performance is estimated from this single denoising using the SNR

$$\mathrm{SNR}(\tilde{f}^r, f_0) = -20 \log_{10}(\|\tilde{f}^r - f_0\|/\|f_0\|),$$

where $\tilde{f}^r$ should be computed from the single realization f^r of f. In the following, with an abuse of notation, when displaying single realization, we ignore the exponent ^{r}. The SNR is expressed in "decibels", denoted dB. This measure of performance requires the knowledge of the clean signal f_0, and should thus only be considered as an experimentation tool, that might not be available in a real life denoising scenario where clean data are not available. Furthermore, the use of an ℓ^2 measure of performance is questionable, and one should also observe the result to judge of the visual quality of the denoising.

6.2 Linear Denoising using Filtering

6.2.1 Translation Invariant Estimators

A linear estimator $\mathcal{E}(f) = \tilde{f}$ of f_0 depends linearly on f, so that $\mathcal{E}(f+g) = \mathcal{E}(f) + \mathcal{E}(g)$. A translation invariant estimator commutes with translation, so that $\mathcal{E}(f_\tau) = \mathcal{E}(f)_\tau$, where $f_\tau(t) = f(t-\tau)$. Such a denoiser can always be written as a filtering

$$\tilde{f} = f \star h$$

where $h \in \mathbb{R}^N$ is a (low pass) filter, that should satisfy at least

$$\sum_n h_n = \hat{h}_0 = 1$$

where $\hat{h}$ is the discrete Fourier transform.

Figure 6.6 shows an example of denoising using a low pass filter.

f_0 f $\tilde{f}$

Figure 6.5: Left: clean image, center: noisy image, right: denoised image.

The filtering strength is usually controlled the width s of h. A typical example is the Gaussian filter

$$\forall\ -N/2 < i \leqslant N/2, \quad h_{s,i} = \frac{1}{Z_s} \exp\left(-\frac{i^2}{2s^2}\right) \tag{6.1}$$

where Z_s ensures that $\sum_i h_{s,i} = 1$ (low pass). Figure 6.6 shows the effect of Gaussian filtering over the spacial and Fourier domains.

Figure 6.7 shows the effect of low pass filtering on a signal and an image with an increasing filter width s. Linear filtering introduces a blur and are thus only efficient to denoise smooth signals and image. For signals and images with discontinuities, this blur deteriorates the signal. Removing a large amount of noise necessitates to also smooth significantly edges and singularities.

6.2.2 Optimal Filter Selection

The selection of an optimal filter is a difficult task. Its choice depends both on the regularity of the (unknown) data f_0 and the noise level σ. A simpler option is to optimize the filter width s among a parametric family of filters, such as for instance the Gaussian filters defined in (6.1).

The denoising error can be decomposed as

$$\|\tilde{f} - f_0\| \leqslant \|h_s \star f_0 - f_0\| + \|h_s \star w\|$$

The filter width s should be optimized to perform a tradeoff between removing enough noise ($\|h_s \star w\|$ decreases with s) and not smoothing too much the singularities (($\|h_s \star f_0 - f_0\|$ increases with s).

Figure (6.8) shows the oracle SNR performance, defined in (**??**).

Figure 6.9 and 6.10 show the results of denoising using the optimal filter width $s^\star$ that minimizes the SNR for a given noisy observation.

These optimal filtering appear quite noisy, and the optimal SNR choice is usually quite conservative. Increasing the filter width introduces a strong blurring that deteriorates the SNR, although it might look visually more pleasant.

6.2.3 Wiener Filter

If one has a random model both for the noise $w \sim W$ and for for the signal $f_0 \sim F$, one can derives an optimal filters in average over both the noise and the signal realizations. One further assumes that w and

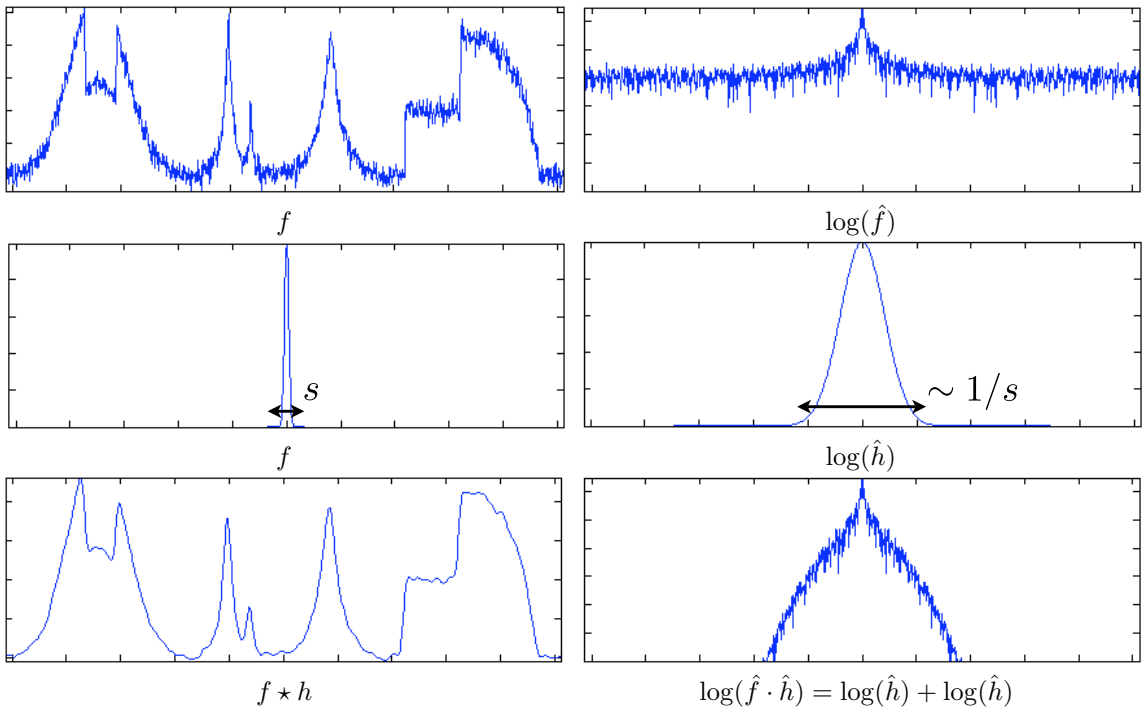


Figure 6.6: Denoising by filtering over the spacial (left) and Fourier (right) domains.

f_0 are independent realization. The optimal h thus minimizes

$$\mathbb{E}_{W,F}(\|h \star (F+W) - F\|^2)$$

If both F is wide-sense stationary, and W is a Gaussian white noise of variance σ^2, then the optimal filer is known as the Wiener filter

$$\hat{h}_\omega = \frac{|\hat{F}_\omega|^2}{|\hat{F}_\omega|^2 + \sigma^2}$$

where $|\hat{F}|^2$ is the power spectrum of F,

$$\hat{F}_\omega = \hat{C}_\omega \quad \text{where} \quad C_n = \mathbb{E}(\langle F, F[\cdot + n] \rangle),$$

the Fourier transform of an infinite vector is defined in Section 2.3.

In practice, one rarely has such a random model for the signal, and interesting signals are often not stationary. Most signals exhibit discontinuities, and are thus poorly restored with filtering.

6.2.4 Denoising and Linear Approximation

In order to study linear (and also non-linear, see the section below) denoising without assuming a random signal model, one should use approximation theory as studied in Chapter 4. We thus consider an ortho-basis $\mathcal{B} = (\psi_m)_m$ of $\mathbb{R}^N$, and consider a simple denoising obtained by keeping only the M first term elements of the approximation of the noisy observation in $\mathcal{B}$

$$\tilde{f} \stackrel{\text{def.}}{=} \sum_{m=1}^{M} \langle f, \psi_m \rangle \psi_m. \tag{6.2}$$

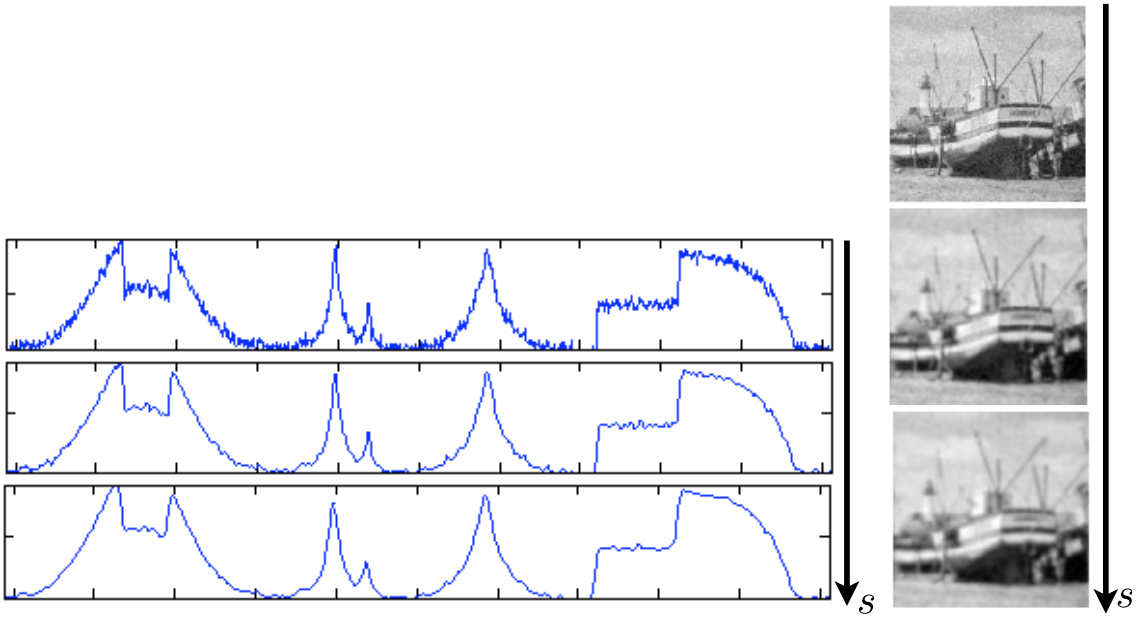


Figure 6.7: Denoising using a filter of increasing width s.

This is a linear projection on the space spanned by $(\psi_m)_{m=1}^M$. This denoising scheme is thus parameterized by some integer $M > 0$, increasing M increases the denoising strength. For instance, when $\mathcal{B}$ is the discrete Fourier basis, this corresponds to an ideal low-pass filter against a (discretized) Dirichlet kernel.

More advanced linear denoising operator can be designed by computing weighted average $\sum_m \lambda_m \langle f, \psi_m \rangle \psi_m$, and (6.2) is retrieve when using binary weights $\alpha_n = 1$ for $n \leqslant M$, and $\alpha_n = 0$ otherwise. The asymptotic theoretical performances described by the following theorem are however not improved by using non-binary weights.

Theorem 8. *We assume that $f_0 \in \mathbb{R}^N$ has a linear approximation error decay that satisfies*

$$\forall\, M, \quad \|f_0 - f_{0,M}^{lin}\|^2 \leqslant C M^{-2\beta} \quad \textit{where} \quad f_{0,M}^{lin} \overset{\text{def.}}{=} \sum_{m=1}^{M} \langle f_0, \psi_m \rangle \psi_m$$

for some constant C. Then the linear denoising error using (6.2) satisfies

$$\mathbb{E}(\|f_0 - \tilde{f}\|^2) \leqslant 2 C^{\frac{1}{2\beta+1}} \sigma^{2 - \frac{1}{\beta+1/2}},$$

when choosing

$$M = C^{\frac{1}{2\beta+1}} \sigma^{-\frac{2}{2\beta+1}}. \tag{6.3}$$

Proof. One has, thanks to the ortho-normality of $(\psi_m)_m$

$$\begin{aligned}
\mathbb{E}(\|f_0 - \tilde{f}\|^2) &= \mathbb{E}(\sum_m \langle f_0 - \tilde{f}, \psi_m \rangle^2) = \mathbb{E}(\sum_{m=1}^{M} \langle f_0 - f, \psi_m \rangle^2 + \sum_{m>M} \langle f_0, \psi_m \rangle^2) \\
&= \mathbb{E}\left(\sum_{m=1}^{M} \langle w, \psi_m \rangle^2 \right) + \sum_{m>M} \langle f_0, \psi_m \rangle^2 = M\sigma^2 + \|f_0 - f_{0,M}^{\text{lin}}\|^2 \\
&\leqslant M\sigma^2 + C M^{-2\beta}.
\end{aligned}$$

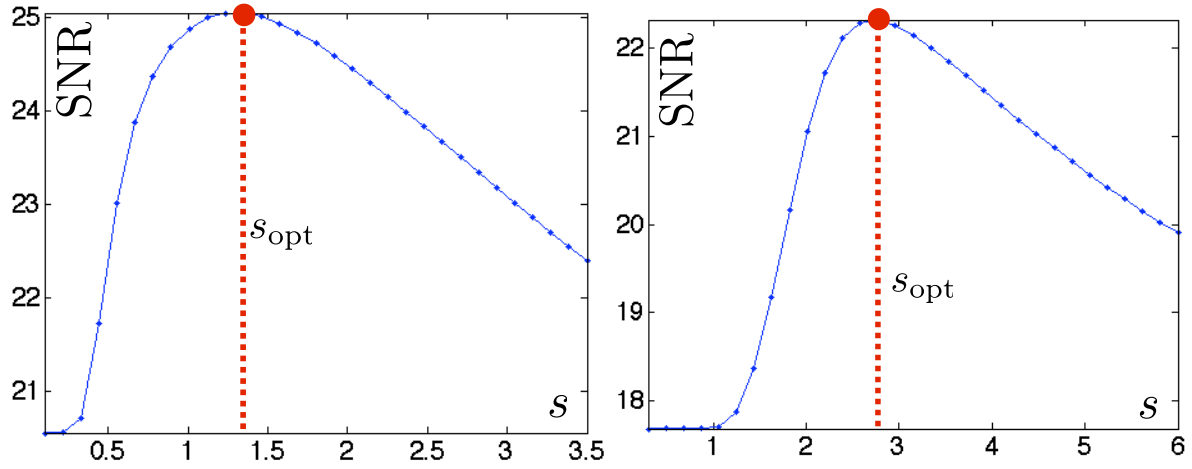


Figure 6.8: Curves of SNR as a function of the filtering width in 1-D (left) and 2-D (right).

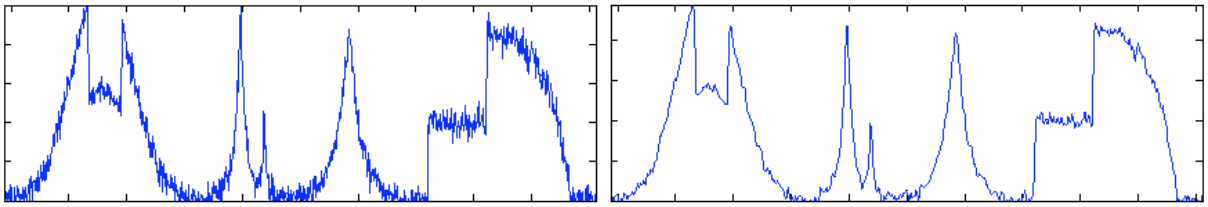


Figure 6.9: Noisy image (left) and denoising (right) using the optimal filter width.

Here we use the fundamental fact that $(\langle w, \psi_m \rangle)_m$ is also $\mathcal{N}(0, \sigma^2 \mathrm{Id}_N)$. Choosing M such that $M\sigma^2 = CM^{-2\beta}$, i.e. $M = C^{\frac{1}{2\beta+1}} \sigma^{-\frac{2}{2\beta+1}}$ leads to

$$\mathbb{E}(\|f_0 - \tilde{f}\|^2) = 2CM^{-2\beta} = 2CC^{-\frac{2\beta}{2\beta+1}} \sigma^{\frac{4\beta}{2\beta+1}} = 2C^{\frac{1}{2\beta+1}} \sigma^{2-\frac{1}{\beta+1/2}}.$$

□

There are several important remark regarding this simple but important result:

– Thanks to the decay of the linear approximation error, the denoising error $\mathbb{E}(\|f_0 - \tilde{f}\|^2)$ is bounded *independently* of the sampling size N, although the input noise level $\mathbb{E}(\|w\|^2) = N\sigma^2$ growth with N.

– If the signal is well approximated linearly, i.e. if β is large, then the denoising error decays fast when the noise level σ drops to zero. The upper bound approaches the optimal rate σ^2 by taking β large enough.

– This theory is finite dimensional, i.e. this computation makes only sense when introducing some discretization step N. This is natural because random noise vectors of finite energy are necessarily finite dimensional. For the choice (6.3) to be realizable, one should however have $M \leqslant N$, i.e. $N \geqslant C^{\frac{1}{2\beta+1}} \sigma^{-\frac{2}{2\beta+1}}$. Thus N should increase when the noise diminish for the denoising effect to kick-in.

– Section 4.3.1 bounds the linear approximation error for infinite dimensional signal and image model. This theory can be applied provided that the discretization error is smaller than the denoising error, i.e. once again, one should use N large enough.

A typical setup where this denoising theorem can be applied is for the Sobolev signal and image model detailed in Section 4.2.1. In the discrete setting, where the sampling size N is intended to grow (specially if σ diminishes), one can similarly consider a "Sobolev-like" model, and similarely as for Proposition 18, this model implies a decay of the linear approximation error.

Figure 6.10: Noisy image (left) and denoising (right) using the optimal filter width.

Proposition 21. *Assuming that*

$$\sum_{m=1}^{N} m^{2\alpha} |\langle f_0, \psi_m \rangle|^2 \leqslant C \tag{6.4}$$

then

$$\forall\, M, \quad \|f_0 - f_{0,M}^{lin}\|^2 \leqslant C M^{-2\alpha}$$

Proof.

$$C \geqslant \sum_{m=1}^{N} m^{2\alpha} |\langle f_0, \psi_m \rangle|^2 \geqslant \sum_{m>M} m^{2\alpha} |\langle f_0, \psi_m \rangle|^2 \geqslant M^{2\alpha} \sum_{m>M} |\langle f_0, \psi_m \rangle|^2 \geqslant M^{2\alpha} \|f_0 - f_{0,M}^{\text{lin}}\|^2.$$

□

If ψ_m is the discrete Fourier basis defined in (2.8), then this discrete Sobolev model (6.4) is equivalent to the continuous Sobolev model of Section 4.2.1, up to a discretization error which tends to 0 as N increase. Choosing N large enough shows that smooth signals and image are thus efficiently denoised by a simple linear projection on the first M element of the Fourier basis.

6.3 Non-linear Denoising using Thresholding

6.3.1 Hard Thresholding

We consider an orthogonal basis $\{\psi_m\}_m$ of $\mathbb{C}^N$, for instance a discrete wavelet basis. The noisy coefficients satisfy

$$\langle f, \psi_m \rangle = \langle f_0, \psi_m \rangle + \langle w, \psi_m \rangle. \tag{6.5}$$

Since a Gaussian white noise is invariant under an orthogonal transformation, $\langle w, \psi_m \rangle$ is also a Gaussian white noise of variance σ^2. If the basis $\{\psi_m\}_m$ is efficient to represent f_0, then most of the coefficients $\langle f_0, \psi_m \rangle$ are close to zero, and one observes a large set of small noisy coefficients, as shown on Figure 6.11. This idea of using thresholding estimator for denoising was first systematically explored by Donoho and Jonhstone [13].

A thresholding estimator removes these small amplitude coefficients using a non-linear hard thresholding

$$\tilde{f} = \sum_{|\langle f, \psi_m \rangle| > T} \langle f, \psi_m \rangle \psi_m = \sum_m S_T(\langle f, \psi_m \rangle) \psi_m.$$

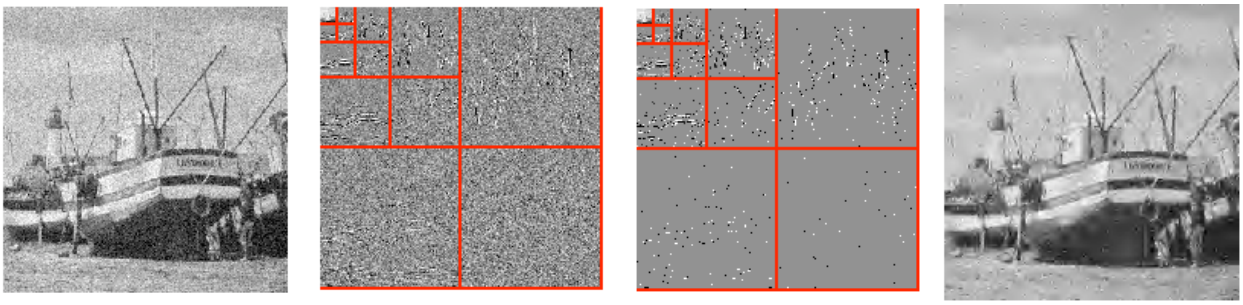


Figure 6.11: Denoising using thresholding of wavelet coefficients.

where S_T is defined in (9.4). This corresponds to the computation of the best M-term approximation $\tilde{f} = f_M$ of the noisy function f. Figure 6.11 shows that if T is well chose, this non-linear estimator is able to remove most of the noise while maintaining sharp features, which was not the case with linear filtering estimatiors.

6.3.2 Soft Thresholding

We recall that the hard thresholding operator is defined as

$$S_T(x) = S^0_T(x) = \begin{cases} x & \text{if} \quad |x| > T, \\ 0 & \text{if} \quad |x| \leqslant T. \end{cases} \tag{6.6}$$

This thresholding performs a binary decision that might introduces artifacts. A less aggressive nonlinearity is the soft thresholding

$$S^1_T(x) = \max(1 - T/|x|, 0)x. \tag{6.7}$$

Figure 6.12 shows the 1-D curves of these 1-D non-linear mapping.

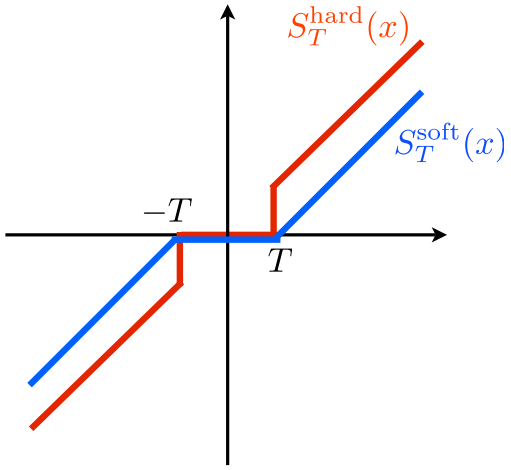


Figure 6.12: Hard and soft thresholding functions.

For $q = 0$ and $q = 1$, these thresholding defines two different estimators

$$\tilde{f}^q = \sum_m S^q_T(\langle f, \psi_m \rangle)\psi_m \tag{6.8}$$

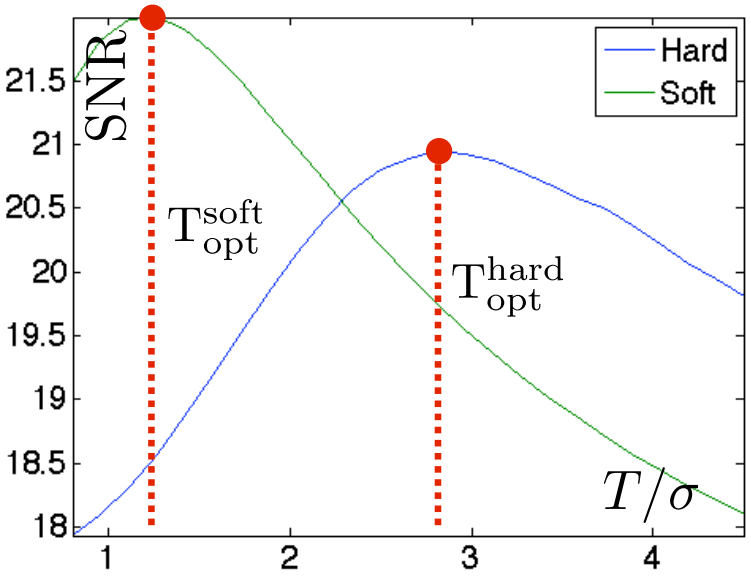


Figure 6.13: Curves of SNR with respect to T/σ for hard and soft thresholding.

Coarse scale management. The soft thresholded S_T^1 introduces a bias since it diminishes the value of large coefficients. For wavelet transforms, it tends to introduces unwanted low-frequencies artifacts by modifying coarse scale coefficients. If the coarse scale is 2^{j_0}, one thus prefers not to threshold the coarse approximation coefficients and use, for instance in 1-D,

$$\tilde{f}^1 = \sum_{0 \leqslant n < 2^{-j_0}} \langle f, \varphi_{j_0,n} \rangle \varphi_{j_0,n} + \sum_{j=j_0}^{0} \sum_{0 \leqslant n < 2^{-j}} S_T^1(\langle f, \psi_{j_0,n} \rangle) \psi_{j_0,n}.$$

Empirical choice of the threshold. Figure 6.13 shows the evolution of the SNR with respect to the threshold T for these two estimators, for a natural image f_0. For the hard thresholding, the best result is obtained around $T \approx 3\sigma$, while for the soft thresholding, the optimal choice is around $T \approx 3\sigma/2$. These results also shows that numerically, for thresholding in orthogonal bases, soft thresholding is slightly superior than hard thresholding on natural signals and images.

Although these are experimental conclusions, these results are robust across various natural signals and images, and should be considered as good default parameters.

6.3.3 Minimax Optimality of Thresholding

Sparse coefficients estimation. To analyze the performance of the estimator, and gives an estimate for the value of T, we first assumes that the coefficients

$$a_{0,m} = \langle f_0, \psi_m \rangle \in \mathbb{R}^N$$

are sparse, meaning that most of the $a_{0,m}$ are zero, so that its ℓ^0 norm

$$\|a_0\|_0 = \#\{m \,;\, a_{0,m} \neq 0\}$$

is small. As shown in (6.5), noisy coefficients

$$\langle f, \psi_m \rangle = a_m = a_{0,m} + z_m$$

20.9dB 21.8dB

Figure 6.14: Comparison of hard (left) and soft (right) thresholding.

are perturbed with an additive Gaussian white noise of variance σ^2. Figure 6.15 shows an example of such a noisy sparse signal.

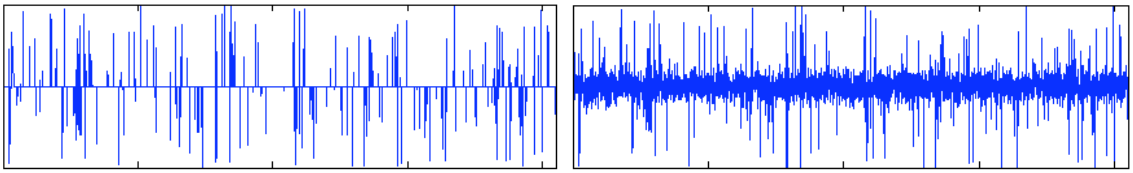


Figure 6.15: Left: sparse signal a, right: noisy signal.

Universal threshold value. If

$$\min_{m:a_{0,m}\neq 0} |a_{0,m}|$$

is large enough, then $\|f_0 - \tilde{f}\| = \|a_0 - S_T(a)\|$ is minimum for

$$T \approx \tau_N = \max_{0\leqslant m<N} |z_m|.$$

τ_N is a random variable that depends on N. One can show that its mean is $\sigma\sqrt{2\log(N)}$, and that as N increases, its variance tends to zero and τ_N is highly concentrated close to its mean. Figure 6.16 shows that this is indeed the case numerically.

Asymptotic optimality. Donoho and Jonhstone [13] have shown that the universal threshold $T = \sigma\sqrt{2\log(N)}$ is a good theoretical choice for the denoising of signals that are well approximated non-linearly in $\{\psi_m\}_m$. The obtain denoising error decay rate with σ can also be shown to be in some sense optimal.

Theorem 9. *We assume that $f_0 \in \mathbb{R}^N$ has a non-linear approximation error decay that satisfies*

$$\forall\, M, \quad \|f_0 - f_{0,M}^{nlin}\|^2 \leqslant CM^{-2\beta} \quad \textit{where} \quad f_{0,M}^{nlin} \stackrel{\text{def.}}{=} \sum_{r=1}^{M} \langle f_0,\, \psi_{m_r}\rangle \psi_{m_r}$$

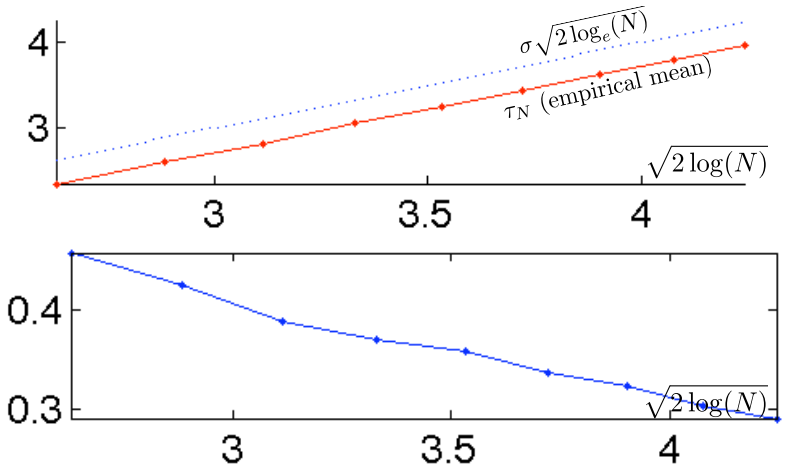


Figure 6.16: Empirical estimation of the mean of Z_n (top) and standard deviation of Z_n (bottom)

for some constant C, where here $(\langle f_0, \psi_{m_r}\rangle)_r$ are the coefficient sorted by decaying magnitude. Then the non-linear denoising error using (6.2) satisfies

$$\mathbb{E}(\|f_0 - \tilde{f}^q\|^2) \leqslant C' \ln(N) \sigma^{2-\frac{1}{\beta+1/2}},$$

for some constant C', when choosing $T = \sqrt{2\ln(N)}$, where $\tilde{f}^q$ is defined in (6.8) for $q \in \{0,1\}$.

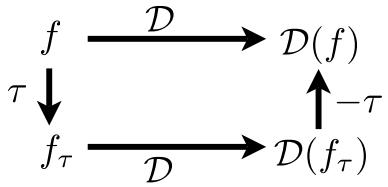
This universal threshold choice $T = \sqrt{2\ln(N)}$ is however very conservative since it is guaranteed to remove almost all the noise. In practice, as shown in Figure 6.14, better results are obtained on natural signals and images by using $T \approx 3\sigma$ and $T \approx 3\sigma/2$ for hard and soft thresholdings.

6.3.4 Translation Invariant Thresholding Estimators

Translation invariance. Let $f \mapsto \tilde{f} = \mathcal{D}(f)$ by a denoising method, and $f_\tau(x) = f(x-\tau)$ be a translated signal or image for $\tau \in \mathbb{R}^d$, ($d = 1$ or $d = 2$). The denoising is said to be translation invariant at precision Δ if

$$\forall\, \tau \in \Delta, \quad \mathcal{D}(f) = \mathcal{D}(f_\tau)_{-\tau}$$

where Δ is a lattice of $\mathbb{R}^d$. The denser Δ is, the more translation invariant the method is. This corresponds to the fact that $\mathcal{D}$ computes with the translation operator.



Imposing translation invariance for a fine enough set Δ is a natural constraint, since intuitively the denoising results should not depend on the location of features in the signal or image. Otherwise, some locations might be favored by the denoising process, which might result in visually unpleasant denoising artifacts.

For denoising by thresholding

$$\mathcal{D}(f) = \sum_m S_T(\langle f, \psi_m\rangle)\psi_m.$$

then translation invariance is equivalent to asking that the basis $\{\psi_m\}_m$ is translation invariant at precision Δ,

$$\forall\, m,\ \forall\, \tau \in \Delta,\ \exists m,\ \exists \lambda \in \mathbb{C}, \qquad (\psi_{m'})_\tau = \lambda \psi_m$$

where $|\lambda| = 1$.

The Fourier basis is fully translation invariant for $\Delta = \mathbb{R}^d$ over $[0,1]^d$ with periodic boundary conditions and the discrete Fourier basis is translation invariant for all interger translations $\Delta = \{0, \ldots, N_0 - 1\}^d$ where $N = N_0$ is the number of points in 1-D, and $N = N_0 \times N_0$ is the number of pixels in 2-D.

Unfortunately, an orthogonal wavelet basis

$$\{\psi_m = \psi_{j,n}\}_{j,n}$$

is not translation invariant both in the continuous setting or in the discrete setting. For instance, in 1-D,

$$(\psi_{j',n'})_\tau \notin \{\psi_{j,n}\} \quad \text{for} \quad \tau = 2^j/2.$$

Cycle spinning. A simple way to turn a denoiser Δ into a translation invariant denoiser is to average the result of translated images

$$\mathcal{D}_{\text{inv}}(f) = \frac{1}{|\Delta|} \sum_{\tau \in \Delta} \mathcal{D}(f_\tau)_{-\tau}. \tag{6.9}$$

One easily check that

$$\forall\, \tau \in \Delta, \quad \mathcal{D}_{\text{inv}}(f) = \mathcal{D}_{\text{inv}}(f_\tau)_{-\tau}$$

To obtain a translation invariance up to the pixel precision for a data of N samples, one should use a set of $|\Delta| = N$ translation vectors. To obtain a pixel precision invariance for wavelets, this will result in $O(N^2)$ operations.

Figure 6.17 shows the result of applying cycle spinning to an orthogonal hard thresholding denoising using wavelets, where we have used the following translation of the continuous wavelet basis $\Delta = \{0, 1/N, 2/N, 3/N\}^2$, which corresponds to discrete translation by $\{0,1,2,3\}^2$ on the discretized image. The complexity of the denoising scheme is thus 16 wavelet transforms. The translation invariance brings a very large SNR improvement, and significantly reduces the oscillating artifacts of orthogonal thresholding. This is because this artifacts pop-out at random locations when τ changes, so that the averaging process reduces significantly these artifacts.

Figure 6.18 shows that translation invariant hard thresholding does a slightly better job than translation invariant soft thresholding. The situation is thus reversed with respect to thresholding in an orthogonal wavelet basis.

Translation invariant wavelet frame. An equivalent way to define a translation invariant denoiser is to replace the orthogonal basis $\mathcal{B} = \{\psi_m\}$ by a redundant family of translated vectors

$$\mathcal{B}_{\text{inv}} = \{(\psi_m)_\tau\}_{m, \tau \in \Delta}. \tag{6.10}$$

One should be careful about the fact that $\mathcal{B}_{\text{inv}}$ is not any more an orthogonal basis, but it still enjoy a conservation of energy formula

$$\|f\|^2 = \frac{1}{|\Delta|} \sum_{m,\tau \in \Delta} |\langle f, (\psi_m)_\tau \rangle|^2 \quad \text{and} \quad f = \frac{1}{|\Delta|} \sum_{m,\tau \in \Delta} \langle f, (\psi_m)_\tau \rangle (\psi_m)_\tau.$$

This kind of redundant family are called tight frames.

One can then define a translation invariant thresholding denoising

$$\mathcal{D}_{\text{inv}}(f) = \frac{1}{|\Delta|} \sum_{m,\tau \in \Delta} S_T(\langle f, (\psi_m)_\tau \rangle)(\psi_m)_\tau. \tag{6.11}$$

21.8dB 23.4dB

Figure 6.17: Comparison of wavelet orthogonal soft thresholding (left) and translation invariant wavelet hard thresholding (right).

This denoising is the same as the cycle spinning denoising defined in (6.9).

The frame $\mathcal{B}_{\text{inv}}$ might contain up to $|\Delta||\mathcal{B}|$ basis element. For a discrete basis of signal with N samples, and a translation lattice of $|\Delta| = N$ vectors, it corresponds to up to N^2 elements in $\mathcal{B}_{\text{inv}}$. Hopefully, for a hierarchical basis such as a discrete orthogonal wavelet basis, one might have

$$(\psi_m)_\tau = (\psi_{m'})_{\tau'} \quad \text{for} \quad m \neq m' \quad \text{and} \quad \tau \neq \tau',$$

so that the number of elements in $\mathcal{B}_{\text{inv}}$ might be much smaller than N^2. For instance, for an orthogonal wavelet basis, one has

$$(\psi_{j,n})_{k2^j} = \psi_{j,n+k},$$

so that the number of basis elements is $|\mathcal{B}_{\text{inv}}| = N \log_2(N)$ for a 2-D basis, and $3N \log_2(N)$ for a 2-D basis. The fast translation invariant wavelet transform, also called "a trou" wavelet transform, computes all the inner products $\langle f, (\psi_m)_\tau \rangle$ in $O(N \log_2(N))$ operations. Implementing formula (6.11) is thus much faster than applying the cycle spinning (6.9) equivalent formulation.

Translation invariant wavelet coefficients are usually grouped by scales in $\log_2(N)$ (for $d = 1$) or by scales and orientations $3 \log_2(N)$ (for $d = 2$) sets of coefficients. For instance, for a 2-D translation invariant transform, one consider

$$\forall\, n \in \{0, \ldots, 2^j N_0 - 1\}^2,\ \forall\, k \in \{0, \ldots, 2^{-j}\}^2, \quad d_j^\omega[2^{-j}n + k] = \langle f, (\psi_{j,n})_{k2^j} \rangle$$

where $\omega \in \{V, H, D\}$ is the orientation. Each set d_j^ω has N coefficients and is a band-pass filtered version of the original image f, as shown on Figure 6.19.

Figure 6.20 shows how these set of coefficients are hard thresholded by the translation invariant estimator.

6.3.5 Exotic Thresholdings

It is possible to devise many thresholding nonlinearities that interpolate between the hard and soft thresholder. We present here two examples, but many more exist in the literature. Depending on the statistical distribution of the wavelet coefficients of the coefficients of f in the basis, these thresholders might produce slightly better results.

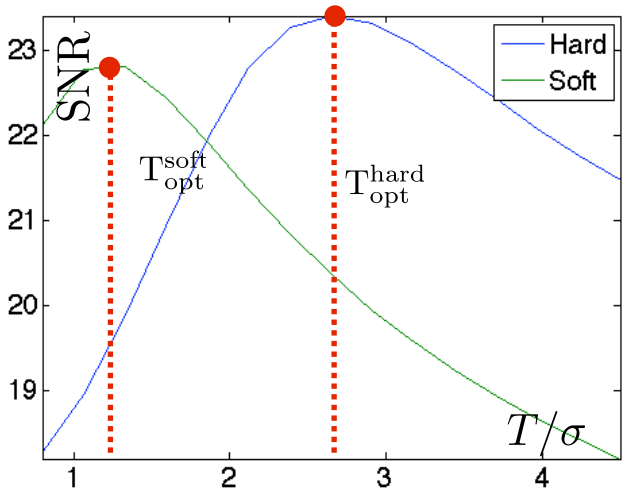


Figure 6.18: Curve of SNR with respect to T/σ for translation invariant thresholding.

Semi-soft thresholding. One can define a family of intermediate thresholder that depends on a parameter $\mu > 1$

$$S_T^\theta(x) = g_{\frac{1}{1-\theta}}(x) \quad \text{where} \quad g_\mu(x) = \begin{cases} 0 & \text{if} \quad |x| < T \\ x & \text{if} \quad |x| > \mu T \\ \text{sign}(x)\frac{|x|-T}{\mu-1} & \text{otherwise.} \end{cases}$$

One thus recovers the hard thresholding as S_T^0 and the soft thresholding as S_T^1. Figure 6.21 display an example of such a non-linearity.

Figure 6.22 shows that a well chosen value of μ might actually improves over both hard and soft thresholders. The improvement is however hardly noticeable visually.

Stein thresholding. The Stein thresholding is defined using a quadratic attenuation of large coefficients

$$S_T^{\text{Stein}}(x) = \max\left(1 - \frac{T^2}{|x|^2}, 0\right) x.$$

This should be compared with the linear attenuation of the soft thresholding

$$S_T^1(x) = \max\left(1 - \frac{T}{|x|}, 0\right) x.$$

The advantage of the Stein thresholder with respect to the soft thresholding is that

$$|S_T^{\text{Stein}}(x) - x| \to 0 \quad \text{whereas} \quad |S_T^1(x) - x| \to T,$$

where $x \to \pm\infty$. This means that Stein thresholding does not suffer from the bias of soft thresholding.

For translation invariant thresholding, Stein and hard thresholding perform similarly on natural images.

6.3.6 Block Thresholding

The non-linear thresholding method presented in the previous section are diagonal estimators, since they operate a coefficient-by-coefficient attenuation

$$\tilde{f} = \sum_m A_T^q(\langle f, \psi_m \rangle)\langle f, \psi_m \rangle \psi_m$$

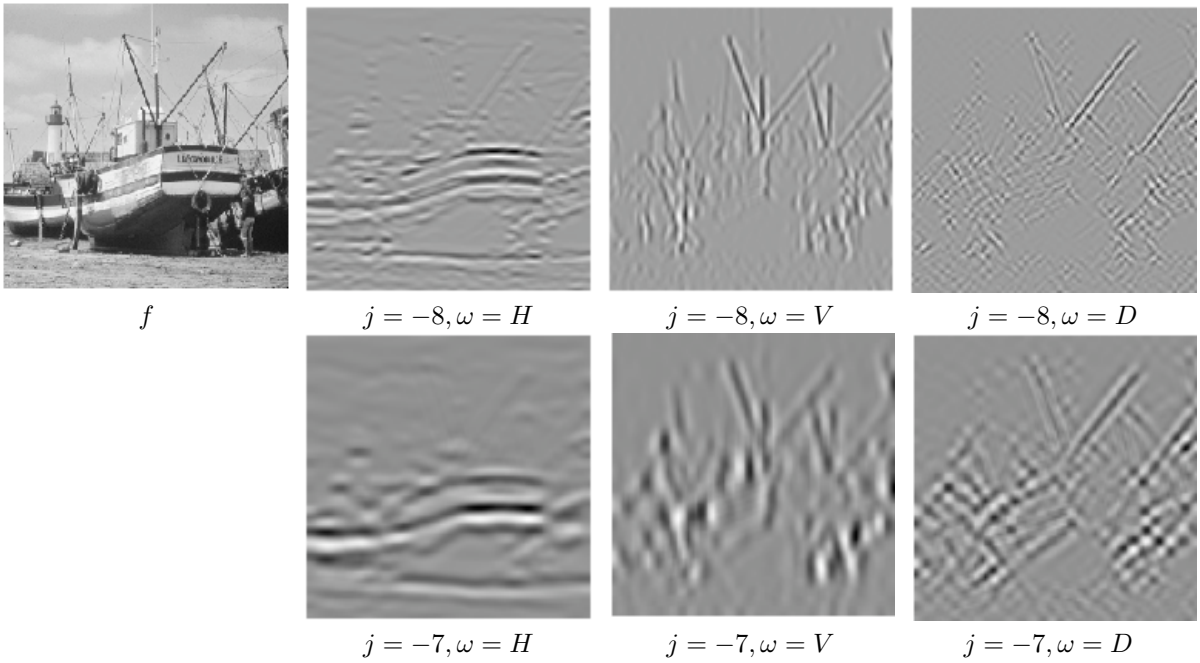


Figure 6.19: Translation invariant wavelet coefficients.

where

$$A_T^q(x) = \begin{cases} \max(1-x^2/T^2,0) & \text{for} \quad q=\text{Stein} \\ \max(1-|x|/T,0) & \text{for} \quad q=1 \text{ (soft)} \\ 1_{|x|>T} & \text{for} \quad q=0 \text{ (hard)} \end{cases}$$

Block thresholding takes advantage of the statistical dependancy of wavelet coefficients, by computing the attenuation factor on block of coefficients. This is especially efficient for natural images, where edges and geometric features create clusters of high magnitude coefficients. Block decisions also help to remove artifacts due to isolated noisy large coefficients in regular areas.

The set of coefficients is divided into disjoint blocks, and for instance for 2-D wavelet coefficients

$$\{(j,n,\omega)\}_{j,n,\omega} = \bigcup_k B_k,$$

where each B_k is a square of $s \times s$ coefficients, where the block size s is a parameter of the method. Figure 6.24 shows an example of such a block.

The block energy is defined as

$$B_k = \frac{1}{s^2} \sum_{m \in B_k} |\langle f,\, \psi_m \rangle|^2,$$

and the block thresholding

$$\tilde{f} = \sum_m S_T^{\text{block},q}(\langle f,\, \psi_m \rangle)\psi_m$$

makes use of the same attenuation for all coefficients within a block

$$\forall\, m \in B_k, \quad S_T^{\text{block},q}(\langle f,\, \psi_m \rangle) = A_T^q(E_k)\langle f,\, \psi_m \rangle.$$

for $q \in \{0, 1, \text{stein}\}$. Figure 6.24 shows the effect of this block attenuation, and the corresponding denoising result.

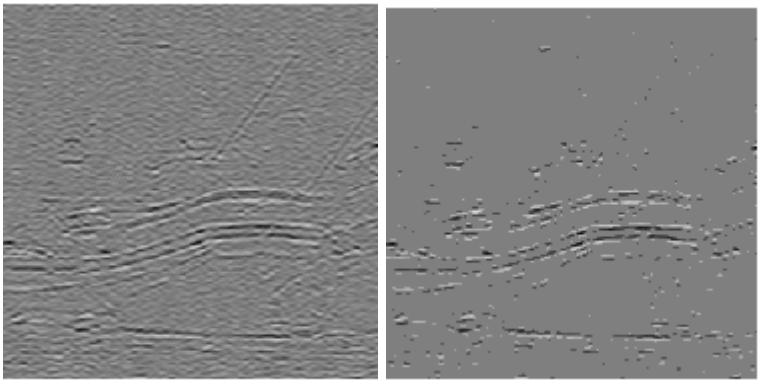


Figure 6.20: Left: translation invariant wavelet coefficients, for $j = -8, \omega = H$, right: tresholded coefficients.

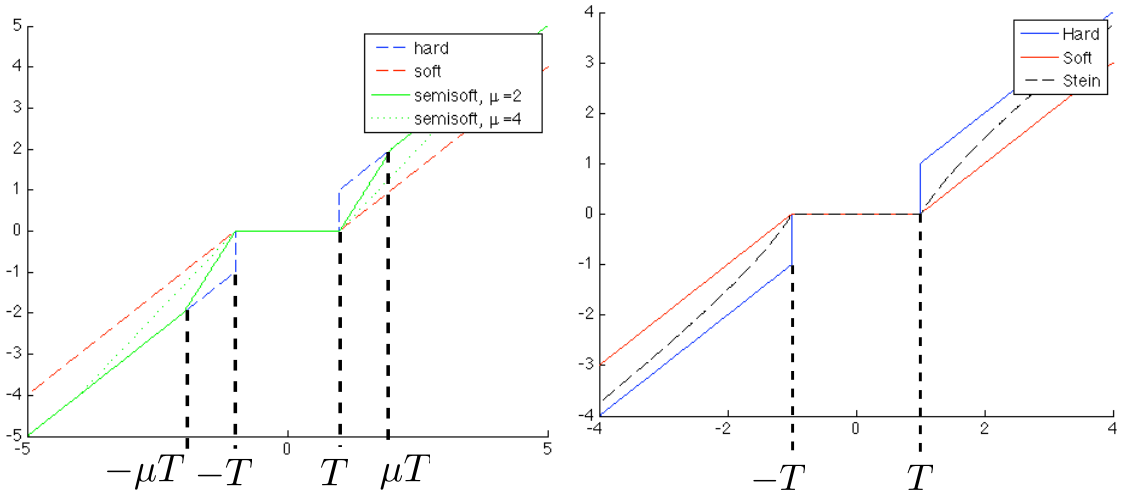


Figure 6.21: Left: semi-soft thresholder, right: Stein thresholder.

Figure 6.25, left, compares the three block thresholding obtained for $q \in \{0, 1, \text{stein}\}$. Numerically, on natural images, Stein block thresholding gives the best results. Figure 6.25, right, compares the block size for the Stein block thresholder. Numerically, for a broad range of images, a value of $s = 4$ works well.

Figure 6.26 shows a visual comparison of the denoising results. Block stein thresholding of orthogonal wavelet coefficients gives a result nearly as good as a translation invariant wavelet hard thresholding, with a faster algorithm. The block thresholding strategy can also be applied to wavelet coefficients in translation invariant tight frame, which produces the best results among all denoisers detailed in this book.

Code **??** implement this block thresholding.

One should be aware that more advanced denoisers use complicated statistical models that improves over the methods proposed in this book, see for instance [21].

6.4 Data-dependant Noises

For many imaging devices, the variance of the noise that perturbs $f_{0,n}$ depends on the value of $f_{0,n}$. This is a major departure from the additive noise formation model considered so far. We present here two popular examples of such non-additive models.

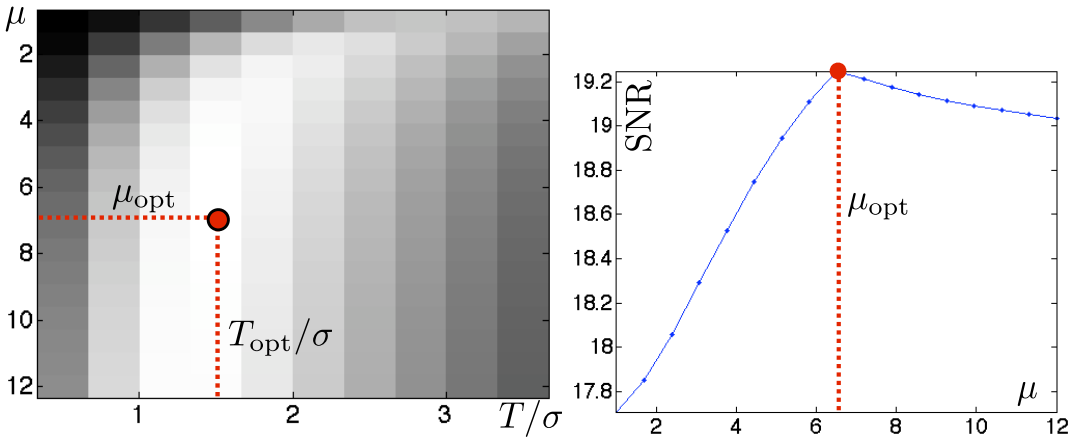


Figure 6.22: Left: image of SNR with respect to the parameters μ and T/σ, right: curve of SNR with respect to μ using the best T/σ for each μ.

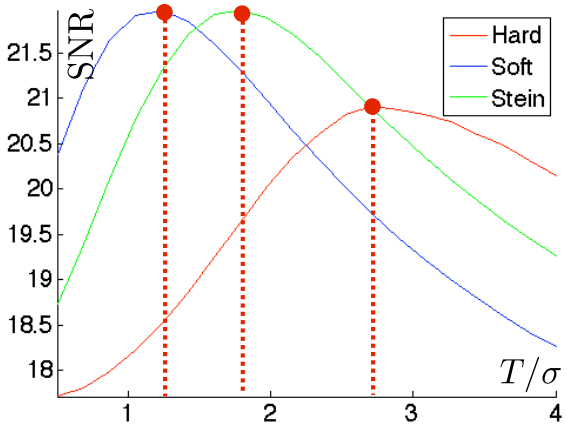


Figure 6.23: SNR curves with respect to T/σ for Stein threhsolding.

6.4.1 Poisson Noise

Many imaging devices sample an image through a photons counting operation. This is for instance the case in digital camera, confocal microscopy, TEP and SPECT tomography.

Poisson model. The uncertainty of the measurements for a quantized unknown image $f_{0,n} \in \mathbb{N}$ is then modeled using a Poisson noise distribution

$$f_n \sim \mathcal{P}(\lambda) \quad \text{where} \quad \lambda = f_{0,n} \in \mathbb{N},$$

and where the Poisson distribution $\mathcal{P}(\lambda)$ is defined as

$$\mathbb{P}(f_n = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

and thus varies from pixel to pixel. Figure 6.27 shows examples of Poisson distributions.

One has

$$\mathbb{E}(f_n) = \lambda = f_{0,n} \quad \text{and} \quad \text{Var}(f_n) = \lambda = f_{0,n}$$

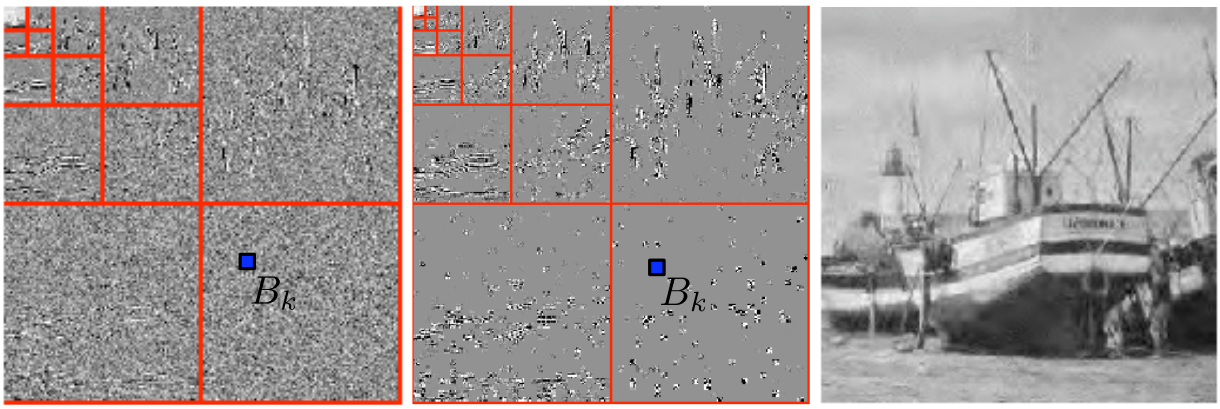


Figure 6.24: Left: wavelet coefficients, center: block thresholded coefficients, right: denoised image.

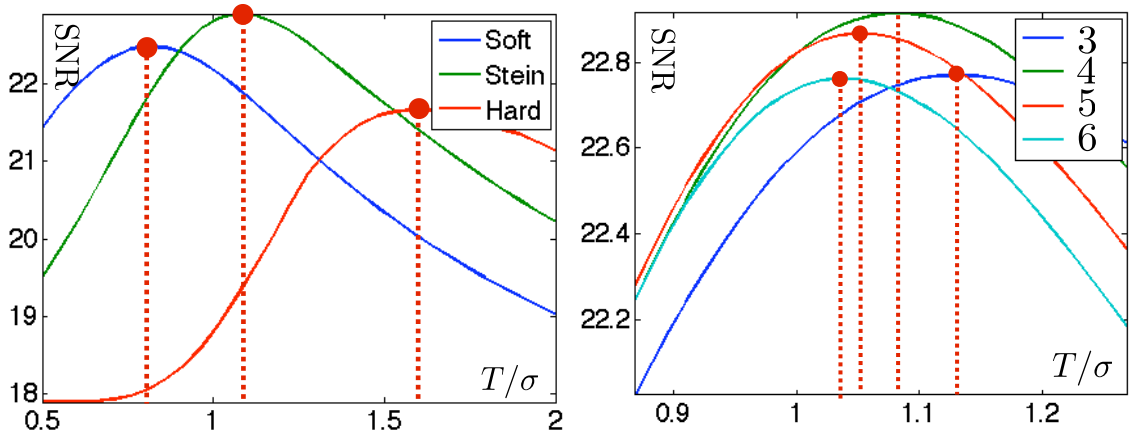


Figure 6.25: Curve of SNR with respect to T/σ (left) and comparison of SNR for different block size (right).

so that the denoising corresponds to estimating the mean of a random vector from a single observation, but the variance now depends on the pixel intensity. This shows that the noise level increase with the intensity of the pixel (more photons are coming to the sensor) but the relative variation $(f_n - f_{0,n})/f_{0,n}$ tends to zero in expectation when $f_{0,n}$ increases.

Figure 6.28 shows examples of a clean image f_0 quantized using different values of $\lambda_{\max}$ and perturbed with the Poisson noise model.

Variance stabilization. Applying thresholding estimator

$$\mathcal{D}(f) = \sum_m S_T^q(\langle f, \psi_m \rangle)\psi_m$$

to f might give poor results since the noise level fluctuates from point to point, and thus a single threshold T might not be able to capture these variations. A simple way to improve the thresholding results is to first apply a variance stabilization non-linearity $\varphi : \mathbb{R} \to \mathbb{R}$ to the image, so that $\varphi(f)$ is as close as possible to an additive Gaussian white noise model

$$\varphi(f) \approx \varphi(f_0) + w \tag{6.12}$$

SNR=23.4dB 22.8dB 23.8dB

Figure 6.26: Left: translation invariant wavelet hard thresholding, center: block orthogonal Stein thresholding, right: block translation invariant Stein thresholding.

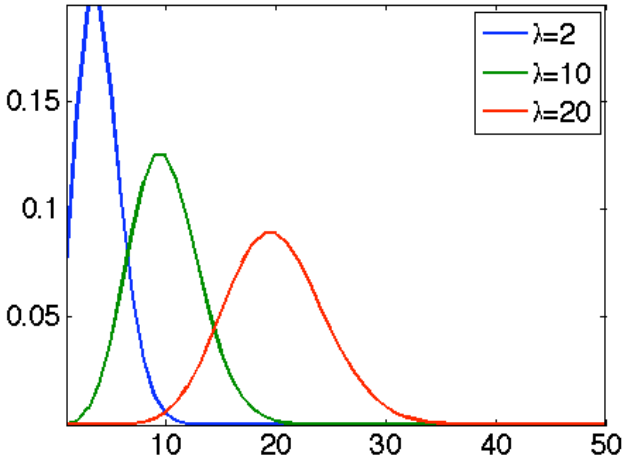


Figure 6.27: Poisson distributions for various λ.

where $w_n \sim \mathcal{N}(0, \sigma)$ is a Gaussian white noise of fixed variance σ^2.

Perfect stabilization is impossible, so that (6.12) only approximately holds for a limited intensity range of $f_{0,n}$. Two popular variation stabilization functions for Poisson noise are the Anscombe mapping

$$\varphi(x) = 2\sqrt{x + 3/8}$$

and the mapping of Freeman and Tukey

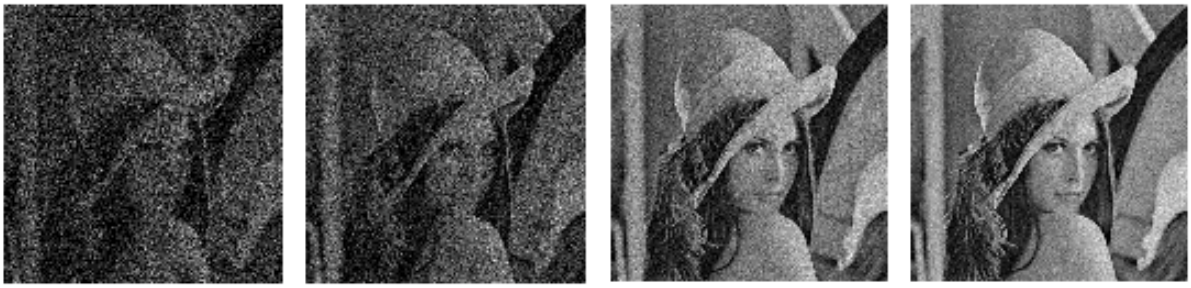
$$\varphi(x) = \sqrt{x+1} + \sqrt{x}.$$

Figure 6.29 shows the effect of these variance stabilizations on the variance of $\varphi(f)$.

A variance stabilized denoiser is defined as

$$\Delta^{\text{stab},q}(f) = \varphi^{-1}(\sum_m S_T^q(\langle \varphi(f),\, \psi_m \rangle)\psi_m)$$

where φ^{-1} is the inverse mapping of φ.



$\lambda_{\max} = 5$ $\lambda_{\max} = 50$ $\lambda_{\max} = 50$ $\lambda_{\max} = 100$

Figure 6.28: Noisy image with Poisson noise model, for various $\lambda_{\max} = \max_n f_{0,n}$.

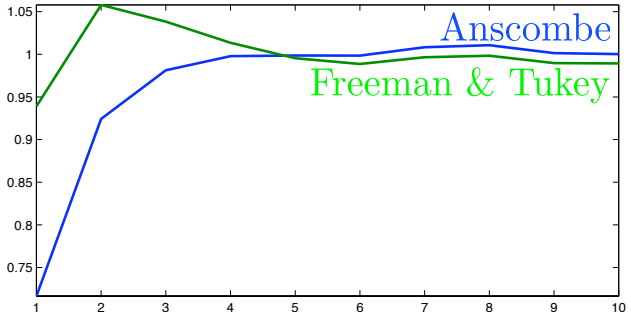


Figure 6.29: Comparison of variariance stabilization: display of $\mathrm{Var}(\varphi(f_n))$ as a function of $f_{0,n}$.

Figure 6.30 shows that for moderate intensity range, variance stabilization improves over non-stabilized denoising.

6.4.2 Multiplicative Noise

Multiplicative image formation. A multiplicative noise model assumes that

$$f_n = f_{0,n} w_n$$

where w is a realization of a random vector with $\mathbb{E}(w) = 1$. Once again, the noise level depends on the pixel value

$$\mathbb{E}(f_n) = f_{0,n} \quad \text{and} \quad \mathrm{Var}(f_n) = f_{0,n}^2 \sigma^2 \quad \text{where} \quad \sigma^2 = \mathrm{Var}(w).$$

Such a mutiplicative noise is a good model for SAR satellite imaging, where f is obtained by averaging S images

$$\forall\, 0 \leqslant s < K, \quad f_n^{(s)} = f_{0,n} w_n^{(s)} + r_n^{(s)}$$

where $r^{(s)}$ is a Gaussian white noise, and $w_n^{(s)}$ is distributed according to a one-sided exponential distribution

$$\mathcal{P}(w_n^{(s)} = x) \propto e^{-x}\, \mathbb{I}_{x>0}.$$

For K large enough, averaging the images cancels the additive noise and one obtains

$$f_n = \frac{1}{K} \sum_{s=1}^{K} f_n^{(s)} \approx f_{0,n} w_n$$

Figure 6.30: Left: noisy image, center: denoising without variance stabilization, right: denoising after variance stabilization.

where w is distributed according to a Gamma distribution

$$w \sim \Gamma(\sigma = K^{-\frac{1}{2}}, \mu = 1) \quad \text{where} \quad \mathbb{P}(w = x) \propto x^{K-1} e^{-Kx},$$

One should note that increasing the value of K reduces the overall noise level.

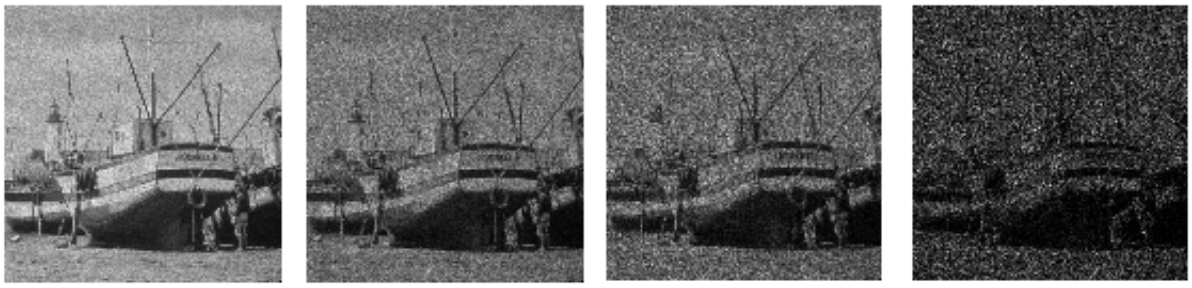


Figure 6.31: Noisy images with multiplicative noise, with varying σ.

Figure **??** shows an example of such image formation for a varying number $K = 1/\sigma^2$ of averaged images.

A simple variance stabilization transform is

$$\varphi(x) = \log(x) - c$$

where

$$c = \mathrm{E}(\log(w)) = \psi(K) - \log(K) \quad \text{where} \quad \psi(x) = \Gamma'(x)/\Gamma(x)$$

and where Γ is the Gamma function that generalizes the factorial function to non-integer. One thus has

$$\varphi(f)_n = \varphi(f_0)_n + z_n,$$

where $z_n = \log(w) - c$ is a zero-mean additive noise.

Figure 6.32 shows the effect of this variance stabilization on the repartition of w and z.

Figure 6.33 shows that for moderate noise level σ, variance stabilization improves over non-stabilized denoising.

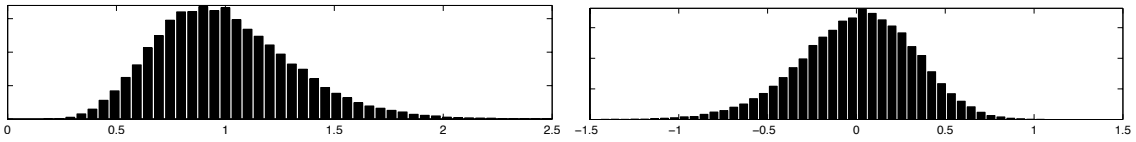


Figure 6.32: Histogram of multiplicative noise before (left) and after (right) stabilization.

Figure 6.33: Left: noisy image, center: denoising without variance stabilization, right: denoising after variance stabilization.

Chapter 7

Variational Priors and Regularization

7.1 Sobolev and Total Variation Priors

The simplest prior are obtained by integrating local differential quantity over the image. They corresponds to norms in functional spaces that imposes some smoothness on the signal of the image. We detail here the Sobolev and the total variation priors, that are the most popular in image processing.

7.1.1 Continuous Priors

In the following, we consider either continuous functions $f \in \mathrm{L}^2([0,1]^2)$ or discrete vectors $f \in \mathbb{R}^N$, and consider continuous priors and there discrete counterparts in Section 7.1.2.

Sobolev prior. The prior energy $J(f) \in \mathbb{R}$ is intended to be low for images in a class $f \in \Theta$. The class of uniformly smooth functions detailed in Section 4.2.1 corresponds to functions in Sobolev spaces. A simple prior derived from this Sobolev class is thus

$$J_{\mathrm{Sob}}(f) = \frac{1}{2}\|f\|^2_{\mathrm{Sob}} = \frac{1}{2}\int \|\nabla f(x)\|^2 \mathrm{d}x, \tag{7.1}$$

where ∇f is the gradient in the sense of distributions.

Total variation prior. To take into account discontinuities in images, one considers a total variation energy, introduced in Section 4.2.3. It was introduced for image denoising by Rudin, Osher and Fatemi [22]

The total variation of a smooth image f is defined as

$$J_{\mathrm{TV}}(f) = \|f\|_{\mathrm{TV}} = \int \|\nabla_x f\| \mathrm{d}x. \tag{7.2}$$

This energy extends to non-smooth functions of bounded variations $f \in \mathrm{BV}([0,1]^2)$. This class contains indicators functions $f = 1_\Omega$ of sets Ω with a bounded perimeter $|\partial\Omega|$.

The total variation norm can be computed alternatively using the co-area formula (4.12), which shows in particular that $\|1_\Omega\|_{\mathrm{TV}} = |\partial\Omega|$.

7.1.2 Discrete Priors

An analog image $f \in \mathrm{L}^2([0,1]^2)$ is discretized through an acquisition device to obtain a discrete image $f \in \mathbb{R}^N$. Image processing algorithms work on these discrete data, and we thus need to define discrete priors for finite dimensional images.

Discrete gradient. Discrete Sobolev and total variation priors are obtained by computing finite differences approximations of derivatives, using for instance forward differences

$$\delta_1 f_{n_1,n_2} = f_{n_1+1,n_2} - f_{n_1,n_2}$$
$$\delta_2 f_{n_1,n_2} = f_{n_1,n_2+1} - f_{n_1,n_2},$$

and one can use higher order schemes to process more precisely smooth functions. One should be careful with boundary conditions, and we consider here for simplicity periodic boundary conditions, which correspond to computing the indexes $n_i + 1$ modulo N. More advanced symmetric boundary conditions can be used as well to avoid boundary artifacts.

A discrete gradient is defined as

$$\nabla f_n = (\delta_1 f_n, \delta_2 f_n) \in \mathbb{R}^2$$

which corresponds to a mapping from images to vector fields

$$\nabla : \mathbb{R}^N \longrightarrow \mathbb{R}^{N\times 2}.$$

Figure 7.1 shows examples of gradient vectors. They point in the direction of the largest slope of the function discretized by f. Figure 7.2 shows gradients and their norms displayed as an image. Regions of high gradients correspond to large intensity variations, and thus typically to edges or textures.

Figure 7.1: Discrete gradient vectors.

Discrete divergence. One can also use backward differences,

$$\tilde{\delta}_1 f_{n_1,n_2} = f_{n_1,n_2} - f_{n_1-1,n_2}$$
$$\tilde{\delta}_2 f_{n_1,n_2} = f_{n_1,n_2} - f_{n_1,n_2-1}.$$

They are dual to the forward differences, so that

$$\delta_i^* = -\tilde{\delta}_i,$$

which means that

$$\forall\, f, g \in \mathbb{R}^N, \quad \langle \delta_i f,\, g\rangle = -\langle f,\, \tilde{\delta}_i g\rangle,$$

which is a discrete version of the integration by part formula

$$\int_0^1 f' g = -\int_0^1 f g'$$

for smooth periodic functions on $[0,1]$.

A divergence operator is defined using backward differences,

$$\mathrm{div}(v)_n = \tilde{\delta}_1 v_{1,n} + \tilde{\delta}_2 v_{2,n},$$

and corresponds to a mapping from vector fields to images

$$\mathrm{div} : \mathbb{R}^{N\times 2} \longrightarrow \mathbb{R}^N.$$

It is related to the dual of the gradient

$$\mathrm{div} = -\nabla^*$$

which means that

$$\forall\, f \in \mathbb{R}^N,\ \forall\, v \in \mathbb{R}^{N\ \times 2}, \quad \langle \nabla f,\, v\rangle_{\mathbb{R}^{N\times 2}} = -\langle f,\, \mathrm{div}(v)\rangle_{\mathbb{R}^N}$$

which corresponds to a discrete version of the divergence theorem.

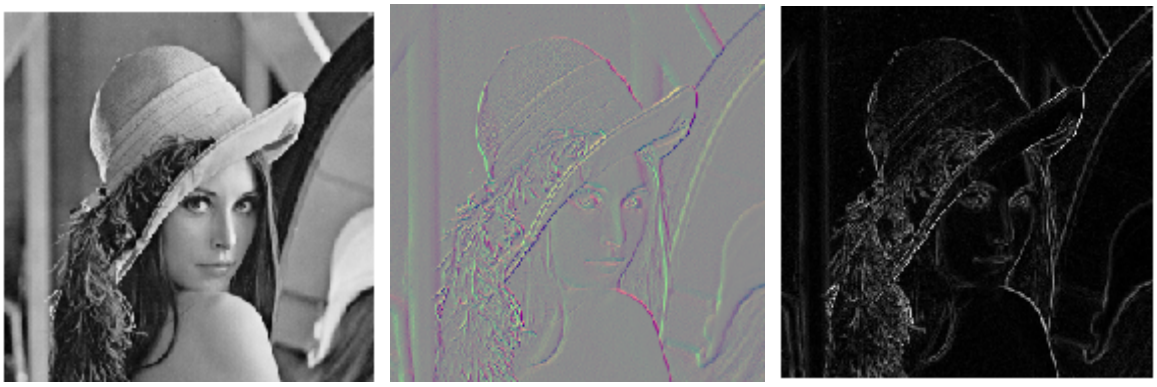


Image f $\quad$ ∇f $\quad$ $\|\nabla f\|$

Figure 7.2: Discrete operators.

Discrete laplacian. A general definition of a Laplacian is

$$\Delta f = \mathrm{div}(\nabla f),$$

which corresponds to a semi-definite negative operator.

For discrete images, and using the previously defined gradient and divergence, it is a local high pass filter

$$\Delta f_n = \sum_{p\in V_4(n)} f_p - 4 f_n, \tag{7.3}$$

that approximates the continuous second order derivative

$$\frac{\partial^2 f}{\partial x_1^2}(x) + \frac{\partial^2 f}{\partial x_2^2} \approx N^2 \Delta f_n \quad \text{for} \quad x = n/N.$$

Lapalacian operators thus correspond to filterings. A continuous Laplacian is equivalently defined over the Fourier domain in diagonal form as

$$g = \Delta f \quad \Longrightarrow \quad \hat{g}(\omega) = \|\omega\|^2 \hat{f}(\omega)$$

and the discrete Laplacian (7.3) as

$$g = \Delta f \quad \Longrightarrow \quad \hat{g}_\omega = \rho_\omega^2 \hat{f}(\omega) \quad \text{where} \quad \rho_\omega^2 = \sin\left(\frac{\pi}{N}\omega_1\right)^2 + \sin\left(\frac{\pi}{N}\omega_2\right)^2. \tag{7.4}$$

Discrete energies. A discrete Sobolev energy is obtained by using the ℓ^2 norm of the discrete gradient vector field

$$J_{\mathrm{Sob}}(f) = \frac{1}{2}\sum_n (\delta_1 f_n)^2 + (\delta_2 f_n)^2 = \frac{1}{2}\|\nabla f\|^2. \tag{7.5}$$

Similarly, a discrete TV energy is defined as the ℓ^1 norm of the gradient field

$$J_{\mathrm{TV}}(f) = \sum_n \sqrt{(\delta_1 f_n)^2 + (\delta_2 f_n)^2} = \|\nabla f\|_1 \tag{7.6}$$

where the ℓ^1 norm of a vector field $v \in \mathbb{R}^{N\times 2}$ is

$$\|v\|_1 = \sum_n \|v_n\| \tag{7.7}$$

where $v_n \in \mathbb{R}^2$.

7.2 PDE and Energy Minimization

Image smoothing is obtained by minimizing the prior using a gradient descent.

7.2.1 General Flows

The gradient of the prior $J : \mathbb{R}^N \to \mathbb{R}$ is a vector $\operatorname{grad} J(f)$. It describes locally up to the first order the variation of the prior

$$J(f+\varepsilon) = J(f) + \langle \varepsilon, \operatorname{grad} J(f)\rangle + o(\|\varepsilon\|).$$

If J is a smooth function of the image f, a discrete energy minimization is obtained through a gradient descent

$$f^{(k+1)} = f^{(k)} - \tau \operatorname{grad} J(f^{(k)}), \tag{7.8}$$

where the step size τ must be small enough to guarantee convergence.

For infinitesimal step size τ, one replaces the discrete parameter k by a continuous time, and the flow

$$t > 0 \longmapsto f_t \in \mathbb{R}^N$$

solves the following partial differential equation

$$\frac{\partial f_t}{\partial t} = -\operatorname{grad} J(f_t) \quad \text{and} \quad f_0 = f. \tag{7.9}$$

The gradient descent can be seen as an explicit discretization in time of this PDE at times $t_k = k\tau$.

7.2.2 Heat Flow

The heat flow corresponds to the instantiation of the generic PDE (7.9) to the case of the Sobolev energies $J_{\mathrm{Sob}}(f)$ defined for continuous function in (7.1) and for discrete images in (7.5).

Since it is a quadratic energy, its gradient is easily computed

$$J(f+\varepsilon) = \frac{1}{2}\|\nabla f + \nabla \varepsilon\|^2 = J(f) - \langle \Delta f, \varepsilon\rangle + o(\|\varepsilon\|^2),$$

so that

$$\operatorname{grad} J_{\mathrm{Sob}}(f) = -\Delta f.$$

Figure 7.4, left, shows an example of Laplacian. It is typically large (positive or negative) near edges.

The heat flow is thus

$$\frac{\partial f_t}{\partial t}(x) = -(\operatorname{grad} J(f_t))(x) = \Delta f_t(x) \quad \text{and} \quad f_0 = f. \tag{7.10}$$

Figure 7.3: Display of f_t for increasing time t for heat flow (top row) and TV flow (bottom row).

Continuous in space. For continuous images and an unbounded domain $\mathbb{R}^2$, the PDE (7.10) has an explicit solution as a convolution with a Gaussian kernel of increasing variance as time increases

$$f_t = f \star h_t \quad \text{where} \quad h_t(x) = \frac{1}{4\pi t} e^{-\frac{-\|x\|^2}{4t}}. \tag{7.11}$$

This shows the regularizing property of the heat flow, that operates a blurring to make the image more regular as time evolves.

Discrete in space. The discrete Sobolev energy (7.5) minimization defined a PDE flow that is discrete in space

$$\frac{\partial f_{n,t}}{\partial t} = -(\operatorname{grad} J(f_t))_n = (\Delta f_t)_n.$$

It can be further discretized in time as done in (7.8) and leads to a fully discrete flow

$$f_n^{(k+1)} = f_n^{(k)} + \tau\Big(\sum_{p \in V_4(n)} f_p - 4 f_n \Big) = (f \star h)_n$$

where $V_4(n)$ are the four neighbor to a pixel n. The flow thus corresponds to iterative convolutions

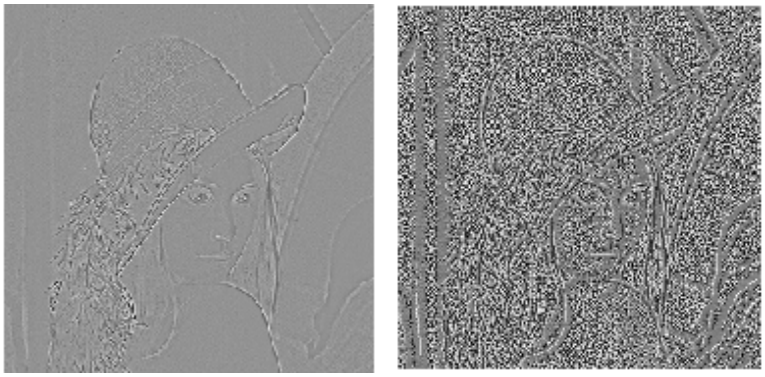
$$f^{(k)} = f \star h \star \ldots \star h = f \star^k h.$$

where h is a discrete filter.

It can be shown to be stable and convergent if $\tau < 1/4$.

7.2.3 Total Variation Flows

Total variation gradient. The total variation energy J_{TV}, both continuous (7.2) and discrete (7.6) is not a smooth function of the image. For instance, the discrete J_{TV} is non-differentiable at an image f such that there exists a pixel n where $\nabla f_n = 0$.



$\operatorname{div}(\nabla f)$ $\operatorname{div}(\nabla f/\|\nabla f\|)$

Figure 7.4: Discrete Laplacian and discrete TV gradient.

If $\nabla f_n \neq 0$, one can compute the gradient of the TV energy specialized at pixel n as

$$(\operatorname{grad} J(f))_n = -\operatorname{div}\left(\frac{\nabla f}{\|\nabla f\|}\right)_n$$

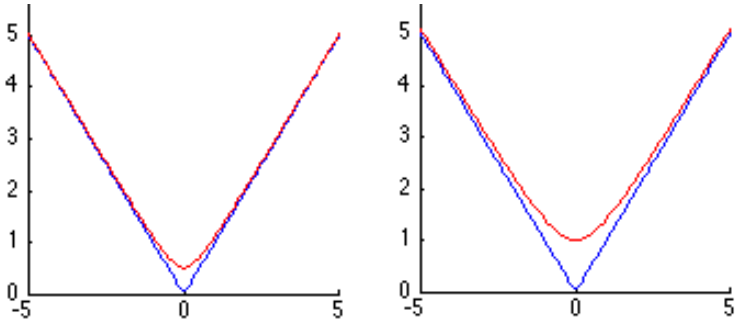
which exhibits a division by zero singularity for a point with vanishing gradient. Figure 7.4 shows an example of TV gradient, which appears noisy in smooth areas, because $\|\nabla f_n\|$ is small in such regions.

This non-differentiabilty makes impossible the definition of a gradient descent and a TV flow.

Regularized total variation. To avoid this issue, one can modify the TV energy, and define a smoothed TV prior

$$J^{\varepsilon}_{\mathrm{TV}}(f) = \sum_n \sqrt{\varepsilon^2 + \|\nabla f_n\|^2} \tag{7.12}$$

where $\varepsilon > 0$ is a small regularization parameter. Figure 7.5 shows this effect of this regularization on the absolute value function.



$\varepsilon = 0.1$ $\varepsilon = 0.01$

Figure 7.5: Regularized absolute value $x \mapsto \sqrt{x^2 + \varepsilon^2}$.

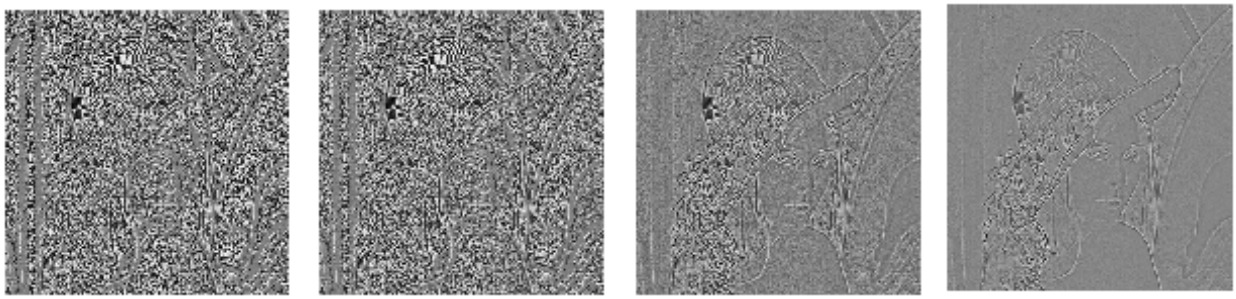
This smoothed TV energy is a differentiable function of the image, and its gradient is

$$\operatorname{grad} J^{\varepsilon}_{\mathrm{TV}}(f) = -\operatorname{div}\left(\frac{\nabla f}{\sqrt{\varepsilon^2 + \|\nabla f\|^2}}\right). \tag{7.13}$$

One can see that this smoothing interpolate between TV and Sobolev, as

$$\operatorname{grad}_f^\varepsilon \sim -\Delta/\varepsilon \quad \text{when} \quad \varepsilon \to +\infty.$$

Figure 7.6 shows the evolution of this gradient for several value of the smoothing parameter.



$\varepsilon = 0.01$ $\varepsilon = 0.05$ $\varepsilon = 0.1$ $\varepsilon = 0.5$

Figure 7.6: Regularized gradient norm $\sqrt{\|\nabla f(x)\|^2 + \varepsilon^2}$.

Regularized total variation flow. The smoothed total variation flow is then defined as

$$\frac{\partial f_t}{\partial t} = \operatorname{div}\left(\frac{\nabla f_t}{\sqrt{\varepsilon^2 + \|\nabla f_t\|^2}}\right). \tag{7.14}$$

Choosing a small ε makes the flow closer to a minimization of the total variation, but makes the computation unstable.

In practice, the flow is computed with a discrete gradient descent (7.8). For the smoothed total variation flow to converge, one needs to impose that $\tau < \varepsilon/4$, which shows that being more faithful to the TV energy requires smaller time steps and thus slower algorithms.

Figure 7.3 shows a comparison between the heat flow and the total variation flow for a small value of ε. This shows that the TV flow smooth less the edges than heat diffusion, which is consistent with the ability of the TV energy to better characterize sharp edges.

7.2.4 PDE Flows for Denoising

PDE flows can be used to remove noise from an observation $f = f_0 + w$. As detailed in Section 6.1.2 a simple noise model assumes that each pixel is corrupted with a Gaussian noise $w_n \sim \mathcal{N}(0, \sigma)$, and that these perturbations are independent (white noise).

The denoising is obtained using the PDE flow within initial image f at time $t = 0$

$$\frac{\partial f_t}{\partial t} = -\operatorname{grad}_{f_t} J \quad \text{and} \quad f_{t=0} = f.$$

An estimator $\tilde{f} = f_{t_0}$ is obtained for a well chose $t = t_0$. Figure 7.7 shows examples of Sobolev and TV flows for denoising.

Since f_t converges to a constant image when $t \to +\infty$, the choice of t_0 corresponds to a tradeoff between removing enough noise and not smoothing too much the edges in the image. This is usually a difficult task. During simulation, if one has access to the clean image f_0, one can monitor the denoising error $\|f_0 - f_t\|$ and choose the $t = t_0$ that minimizes this error. Figure 7.8, top row, shows an example of this oracle estimation of the best stopping time.

Figure 7.7: Denoising using f_t displayed for various time t for Sobolev (top) and TV (bottom) flows.

7.3 Regularization for Denoising

Instead of performing a gradient descent flow for denoising as detailed in Section 7.2.4 and select a stopping time, one can formulate an optimization problem. The estimator is then defined as a solution of this optimization. This setup has the advantage as being well defined mathematically even for non-smooth priors such as the TV prior J_{TV} or the sparsity prior J_1. Furthermore, this regularization framework is also useful to solve general inverse problems as detailed in Chapter **??**.

7.3.1 Regularization

Given some noisy image $f = f_0 + w$ of N pixels and a prior J, we consider the convex optimization problem

$$f_\lambda^\star \in \operatorname*{argmin}_{g \in \mathbb{R}^N} \frac{1}{2}\|f - g\|^2 + \lambda J(g), \tag{7.15}$$

where $\lambda > 0$ is a Lagrange multiplier parameter that weights the influence of the data fitting term $\|f - g\|^2$ and the regularization term $J(g)$.

If one has at his disposal a clean original image f_0, one can minimize the denoising error $\|f_\lambda^\star - f_0\|$, but it is rarely the case. In practice, this parameter should be adapted to the noise level and to the regularity of the unknown image f_0, which might be a non trivial task.

We note that since we did not impose J to be convex, the problem (7.15) might have several optimal solutions.

An estimator is thus defined as

$$\tilde{f} = f_\lambda^\star$$

for a well chosen λ.

If J is differentiable and convex, one can compute the solution of (7.15) through a gradient descent

$$f^{(k+1)} = f^{(k)} - \tau \left(f^{(k)} - \lambda \operatorname{grad} J(f^{(k)}) \right) \tag{7.16}$$

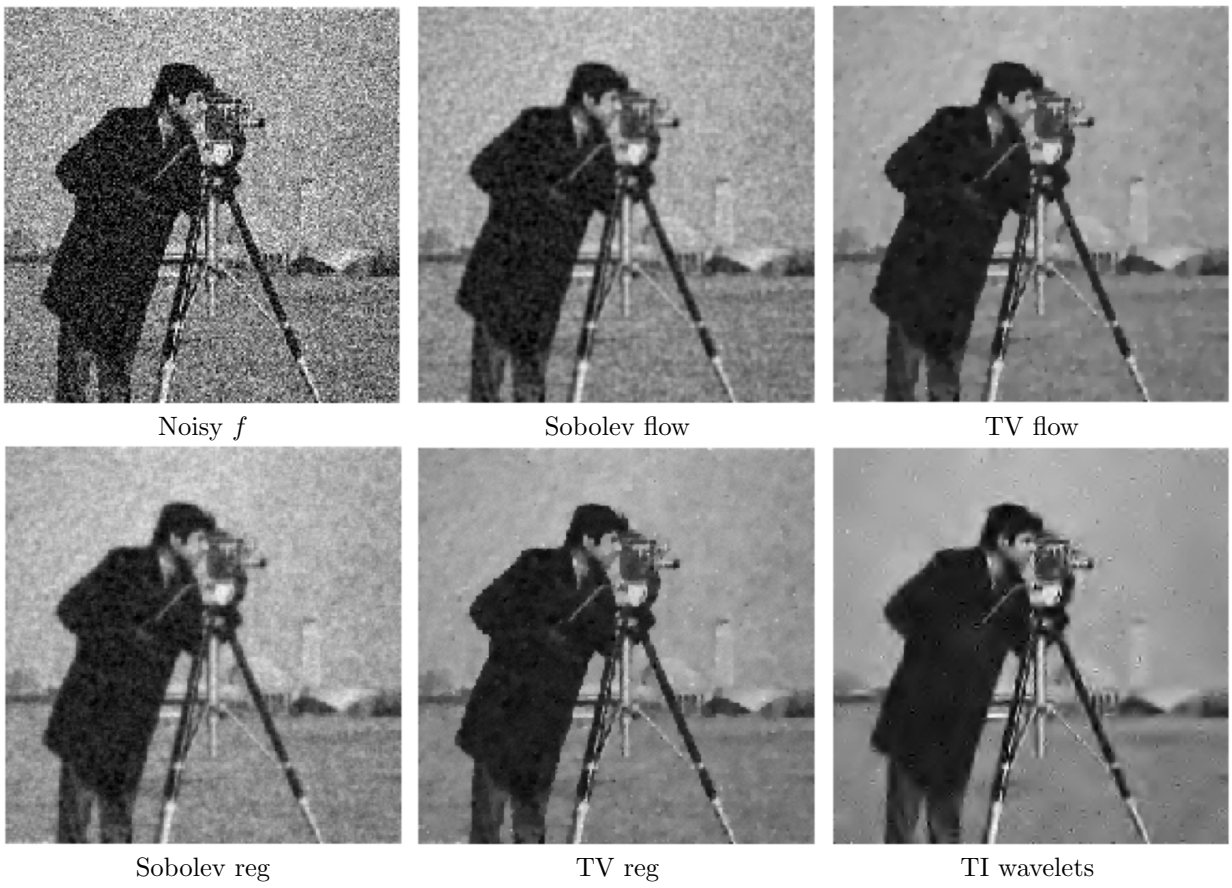


Figure 7.8: Denoising using PDE flows and regularization.

where the descent step size $\tau > 0$ should be small enough. This gradient descent is similar to the time-discretized minimization flow (7.8), excepted that the data fitting term prevent the flow to converge to a constant image.

Unfortunately, priors such as the total variation J_{TV} or the sparsity J_1 are non-differentiable. Some priors such as the ideal sparsity J_0 might even be non-convex. This makes the simple gradient descent not usable to solve for (7.15). In the following Section we show how to compute $f_\lambda^\star$ for several priors.

7.3.2 Sobolev Regularization

The discrete Sobolev prior defined in (7.5) is differentiable, and the gradient descent (7.16) reads

$$f^{(k+1)} = (1-\tau) f^{(k)} + \tau f - \tau\lambda\Delta J(f^{(k)}.$$

Since $J(f) = \|\nabla f\|^2$ is quadratic, one can use a conjugate gradient descent, which converges faster.

The solution $f_\lambda^\star$ can be computed in closed form as the solution of a linear system

$$f_\lambda^\star = (\mathrm{Id}_N - \lambda\Delta)^{-1} f,$$

which shows that the regularization (7.15) is computing an estimator that depends linearly on the observations f.

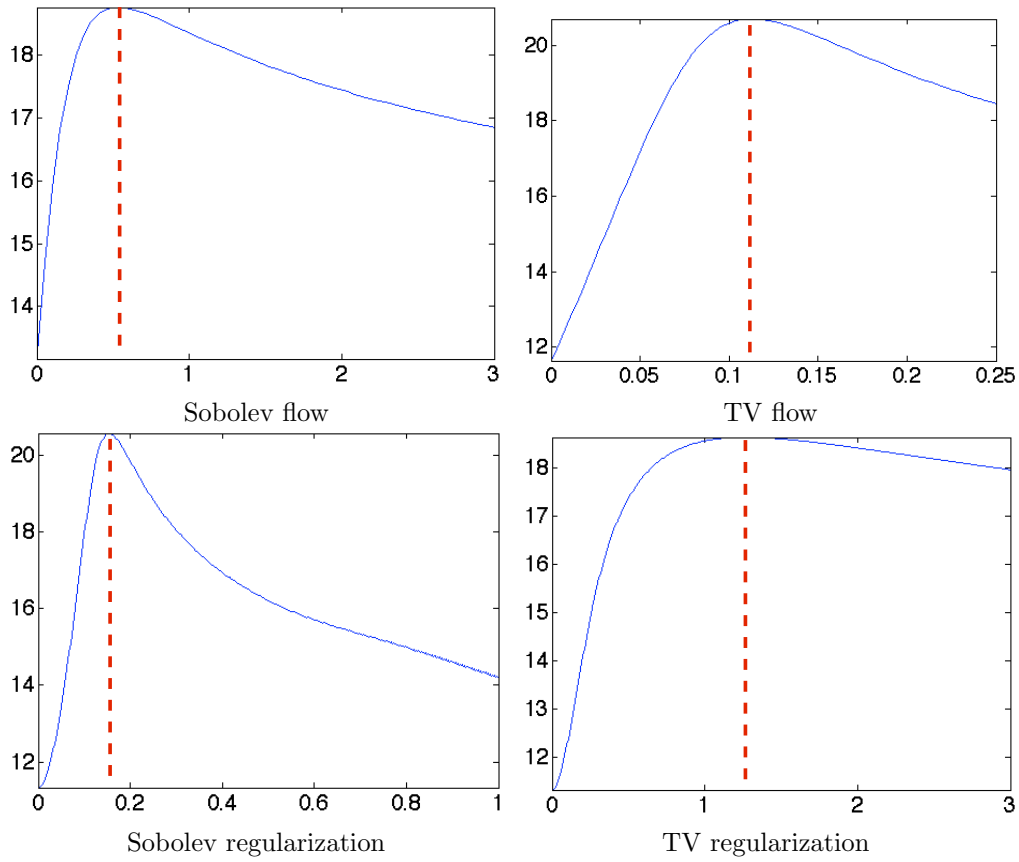


Figure 7.9: SNR as a function of time t for flows (top) and λ for regularization (bottom).

If the differential operators are computed with periodic boundary conditions, this linear system can be solved exactly over the Fourier domain

$$(\hat{f}^\star_\lambda)_\omega = \frac{1}{1+\lambda\rho_\omega^2}\hat{f}_\omega \tag{7.17}$$

where ρ_ω depends on the discretization of the Laplacian, see for instance (7.4).

Equation (7.17) shows that denoising using Sobolev regularization corresponds to a low pass filtering, whose strength is controlled by λ. This should be related to the solution (7.11) of the heat equation, which also corresponds to a filtering, but using a Gaussian low pass kernel parameterized by its variance t^2.

This Sobolev regularization denoising is a particular case of the linear estimator considered in Section 6.2. The selection of the parameter λ is related to the selection of an optimal filter as considered in Section 6.2.2, but with the restriction that the filter is computed in a parametric family.

7.3.3 TV Regularization

The total variation prior J_{TV} defined in (7.6) is non-differentiable. One can either use a smoothed approximation of the prior, or use an optimization algorithm not based on a gradient descent.

The TV prior can be approximated to obtain the prior $J^\varepsilon_{\mathrm{TV}}(g)$ defined in (7.12), which is differentiable with respect to g. Using the gradient of this prior defined in (7.13), one obtains the following instantiation

of the gradient descent (7.16)

$$f^{(k+1)} = (1-\tau) f^{(k)} + \tau f + \lambda \tau \operatorname{div}\left(\frac{\nabla f_t}{\sqrt{\varepsilon^2 + \|\nabla f_t\|^2}} \right). \tag{7.18}$$

which converge to the unique minimizer $f_\lambda^\star$ of (7.15).

Section 17.4.1 details a better alternative which does not require introducing this ε smoothing.

Chapter 8

Inverse Problems

The main references for this chapter are [17, 23, 14].

8.1 Inverse Problems Regularization

Increasing the resolution of signals and images requires to solve an ill posed inverse problem. This corresponds to inverting a linear measurement operator that reduces the resolution of the image. This chapter makes use of convex regularization introduced in Chapter **??** to stabilize this inverse problem.

We consider a (usually) continuous linear map $\Phi : \mathcal{S} \to \mathcal{H}$ where $\mathcal{S}$ can be an Hilbert or a more general Banach space. This operator is intended to capture the hardware acquisition process, which maps a high resolution unknown signal $f_0 \in \mathcal{S}$ to a noisy low-resolution obervation

$$y = \Phi f_0 + w \in \mathcal{H}$$

where $w \in \mathcal{H}$ models the acquisition noise. In this section, we do not use a random noise model, and simply assume that $\|w\|_{\mathcal{H}}$ is bounded.

In most applications, $\mathcal{H} = \mathbb{R}^P$ is finite dimensional, because the hardware involved in the acquisition can only record a finite (and often small) number P of observations. Furthermore, in order to implement numerically a recovery process on a computer, it also makes sense to restrict the attention to $\mathcal{S} = \mathbb{R}^N$, where N is number of point on the discretization grid, and is usually very large, $N \gg P$. However, in order to perform a mathematical analysis of the recovery process, and to be able to introduce meaningful models on the unknown f_0, it still makes sense to consider infinite dimensional functional space (especially for the data space $\mathcal{S}$).

The difficulty of this problem is that the direct inversion of Φ is in general impossible or not advisable because Φ^{-1} have a large norm or is even discontinuous. This is further increased by the addition of some measurement noise w, so that the relation $\Phi^{-1}y = f_0 + \Phi^{-1}w$ would leads to an explosion of the noise $\Phi^{-1}w$.

We now gives a few representative examples of forward operators Φ.

Denoising. The case of the identity operator $\Phi = \mathrm{Id}_{\mathcal{S}}$, $\mathcal{S} = \mathcal{H}$ corresponds to the classical denoising problem, already treated in Chapters **??** and **??**.

De-blurring and super-resolution. For a general operator Φ, the recovery of f_0 is more challenging, and this requires to perform both an inversion and a denoising. For many problem, this two goals are in contradiction, since usually inverting the operator increases the noise level. This is for instance the case for the deblurring problem, where Φ is a translation invariant operator, that corresponds to a low pass filtering with some kernel h

$$\Phi f = f \star h. \tag{8.1}$$

One can for instance consider this convolution over $\mathcal{S} = \mathcal{H} = L^2(\mathbb{T}^d)$, see Proposition 3. In practice, this convolution is followed by a sampling on a grid $\Phi f = \{(f \star h)(x_k)\ ;\ 0 \leqslant k < P\}$, see Figure 8.1, middle, for an example of a low resolution image Φf_0. Inverting such operator has important industrial application to upsample the content of digital photos and to compute high definition videos from low definition videos.

Interpolation and inpainting. Inpainting corresponds to interpolating missing pixels in an image. This is modelled by a diagonal operator over the spacial domain

$$(\Phi f)(x) = \begin{cases} 0 & \text{if} \quad x \in \Omega, \\ f(x) & \text{if} \quad x \notin \Omega. \end{cases} \tag{8.2}$$

where $\Omega \subset [0,1]^d$ (continuous model) or $\{0, \ldots, N-1\}$ which is then a set of missing pixels. Figure 8.1, right, shows an example of damaged image Φf_0.

Original f_0 Low resolution Φf_0 Masked Φf_0

Figure 8.1: Example of inverse problem operators.

Medical imaging. Most medical imaging acquisition device only gives indirect access to the signal of interest, and is usually well approximated by such a linear operator Φ. In scanners, the acquisition operator is the Radon transform, which, thanks to the Fourier slice theorem, is equivalent to partial Fourier mesurments along radial lines. Medical resonance imaging (MRI) is also equivalent to partial Fourier measures

$$\Phi f = \left\{ \hat{f}(x)\ ;\ x \in \Omega \right\}. \tag{8.3}$$

Here, Ω is a set of radial line for a scanner, and smooth curves (e.g. spirals) for MRI.

Other indirect application are obtained by electric or magnetic fields measurements of the brain activity (corresponding to MEG/EEG). Denoting $\Omega \subset \mathbb{R}^3$ the region around which measurements are performed (e.g. the head), in a crude approximation of these measurements, one can assume $\Phi f = \{(\psi \star f)(x)\ ;\ x \in \partial\Omega\}$ where $\psi(x)$ is a kernel accounting for the decay of the electric or magnetic field, e.g. $\psi(x) = 1/\|x\|^2$.

Regression for supervised learning. While the focus of this chapter is on imaging science, a closely related problem is supervised learning using linear model. The typical notations associated to this problem are usually different, which causes confusion. This problem is detailed in Chapter 12.3, which draws connection between regression and inverse problems. In statistical learning, one observes pairs $(x_i, y_i)_{i=1}^n$ of n observation, where the features are $x_i \in \mathbb{R}^p$. One seeks for a linear prediction model of the form $y_i = \langle \beta, x_i \rangle$ where the unknown parameter is $\beta \in \mathbb{R}^p$. Storing all the x_i as rows of a matrix $X \in \mathbb{R}^{n \times p}$, supervised learning aims at approximately solving $X\beta \approx y$. The problem is similar to the inverse problem $\Phi f = y$ where one

performs the change of variable $\Phi \mapsto X$ and $f \mapsto \beta$, with dimensions $(P, N) \to (n, p)$. In statistical learning, one does not assume some well specified model $y = \Phi f_0 + w$, and the major difference is that the matrix X is random, which add extra "noise" which needs to be controlled as $n \to +\infty$. The recovery is performed by the normalized ridge regression problem

$$\min_{\beta} \frac{1}{2n}\|X\beta - y\|^2 + \lambda\|\beta\|^2$$

so that the natural change of variable should be $\frac{1}{n}X^*X \sim \Phi^*\Phi$ (empirical covariance) and $\frac{1}{n}X^*y \sim \Phi^*y$. The law of large number shows that $\frac{1}{n}X^*X$ and $\frac{1}{n}X^*y$ are contaminated by a noise of amplitude $1/\sqrt{n}$, which plays the role of $\|w\|$.

8.2 Theoretical Study of Quadratic Regularization

We now give a glimpse on the typical approach to obtain theoretical guarantee on recovery quality in the case of Hilbert space. The goal is not to be exhaustive, but rather to insist on the modelling hypotethese, namely smoothness implies a so called "source condition", and the inherent limitations of quadratic methods (namely slow rates and the impossibility to recover information in $\ker(\Phi)$, i.e. to achieve super-resolution).

8.2.1 Singular Value Decomposition

Finite dimension. Let us start by the simple finite dimensional case $\Phi \in \mathbb{R}^{P\times N}$ so that $\mathcal{S} = \mathbb{R}^N$ and $\mathcal{H} = \mathbb{R}^P$ are Hilbert spaces. In this case, the Singular Value Decomposition (SVD) is the key to analyze the operator very precisely, and to describe linear inversion process.

Proposition 22 (SVD). *There exists $(U, V) \in \mathbb{R}^{P\times R} \times \mathbb{R}^{N\times R}$, where $R = \mathrm{rank}(\Phi) = \dim(\mathrm{Im}(\Phi))$, with $U^\top U = V^\top V = \mathrm{Id}_R$, i.e. having orthogonal columns $(u_m)_{m=1}^R \subset \mathbb{R}^N, (v_m)_{m=1}^R \subset \mathbb{R}^P$, and $(\sigma_m)_{m=1}^R$ with $\sigma_m > 0$, such that*

$$\Phi = U\,\mathrm{diag}_m(\sigma_m)V^\top = \sum_{m=1}^{R} \sigma_m u_m v_m^\top. \tag{8.4}$$

Proof. We first analyze the problem, and notice that if $\Phi = U\Sigma V^\top$ with $\Sigma = \mathrm{diag}_m(\sigma_m)$, then $\Phi\Phi^\top = U\Sigma^2U^\top$ and then $V^\top = \Sigma^{-1}U^\top\Phi$. We can use this insight. Since $\Phi\Phi^\top$ is a positive symmetric matrix, we write its eigendecomposition as $\Phi\Phi^\top = U\Sigma^2U^\top$ where $\Sigma = \mathrm{diag}_{m=1}^R(\sigma_m)$ with $\sigma_m > 0$. We then define $V \overset{\text{def.}}{=} \Phi^\top U\Sigma^{-1}$. One then verifies that

$$V^\top V = (\Sigma^{-1}U^\top\Phi)(\Phi^\top U\Sigma^{-1}) = \Sigma^{-1}U^\top(U\Sigma^2U^\top)U\Sigma^{-1} = \mathrm{Id}_R \quad \text{and} \quad U\Sigma V^\top = U\Sigma\Sigma^{-1}U^\top\Phi = \Phi.$$

□

This theorem is still valid with complex matrice, replacing $^\top$ by *. Expression (8.4) describes Φ as a sum of rank-1 matrices $u_m v_m^\top$. One usually order the singular values $(\sigma_m)_m$ in decaying order $\sigma_1 \geqslant \ldots \geqslant \sigma_R$. If these values are different, then the SVD is unique up to ± 1 sign change on the singular vectors.

The left singular vectors U is an orthonormal basis of $\mathrm{Im}(\Phi)$, while the right singular values is an orthonormal basis of $\mathrm{Im}(\Phi^\top) = \ker(\Phi)^\perp$. The decomposition (8.4) is often call the "reduced" SVD because one has only kept the R non-zero singular values. The "full" SVD is obtained by completing U and V to define orthonormal bases of the full spaces $\mathbb{R}^P$ and $\mathbb{R}^N$. Then Σ becomes a rectangular matrix of size $P \times N$.

A typical example is for $\Phi f = f \star h$ over $\mathbb{R}^P = \mathbb{R}^N$, in which case the Fourier transform diagonalizes the convolution, i.e.

$$\Phi = (u_m)_m^* \,\mathrm{diag}(\hat{h}_m)(u_m)_m \tag{8.5}$$

where $(u_m)_n \overset{\text{def.}}{=} \frac{1}{\sqrt{N}} e^{\frac{2\mathrm{i}\pi}{N}nm}$ so that the singular values are $\sigma_m = |\hat{h}_m|$ (removing the zero values) and the singular vectors are $(u_m)_n$ and $(v_m\theta_m)_n$ where $\theta_m \overset{\text{def.}}{=} |\hat{h}_m|/\hat{h}_m$ is a unit complex number.

Computing the SVD of a full matrix $\Phi \in \mathbb{R}^{N\times N}$ has complexity N^3.

Compact operators. One can extend the decomposition to compact operators $\Phi : \mathcal{S} \to \mathcal{H}$ between separable Hilbert space. A compact operator is such that ΦB_1 is pre-compact where $B_1 = \{s \in \mathcal{S} \; ; \; \|s\| \leqslant 1\}$ is the unit-ball. This means that for any sequence $(\Phi s_k)_k$ where $s_k \in B_1$ one can extract a converging sub-sequence. Note that in infinite dimension, the identity operator $\Phi : \mathcal{S} \to \mathcal{S}$ is never compact.

Compact operators Φ can be shown to be equivalently defined as those for which an expansion of the form (8.4) holds

$$\Phi = \sum_{m=1}^{+\infty} \sigma_m u_m v_m^\top \tag{8.6}$$

where $(\sigma_m)_m$ is a decaying sequence converging to 0, $\sigma_m \to 0$. Here in (8.6) convergence holds in the operator norm, which is the algebra norm on linear operator inherited from those of $\mathcal{S}$ and $\mathcal{H}$

$$\|\Phi\|_{\mathcal{L}(\mathcal{S},\mathcal{H})} \overset{\text{def.}}{=} \min_{\|\Phi u\|_{\mathcal{H}}} \|u\|_{\mathcal{S}} \leqslant 1.$$

For Φ having an SVD decomposition (8.6), $\|\Phi\|_{\mathcal{L}(\mathcal{S},\mathcal{H})} = \sigma_1$.

When $\sigma_m = 0$ for $m > R$, Φ has a finite rank $R = \dim(\mathrm{Im}(\Phi))$. As we explain in the sections below, when using *linear* recovery methods (such as quadratic regularization), the inverse problem is equivalent to a finite dimensional problem, since one can restrict its attention to functions in $\ker(\Phi)^\perp$ which as dimension R. Of course, this is not true anymore when one can retrieve function inside $\ker(\Phi)$, which is often referred to as a "super-resolution" effect of non-linear methods. Another definition of compact operator is that they are the limit of finite rank operator. They are thus in some sense the extension of finite dimensional matrices, and are the correct setting to model ill-posed inverse problems. This definition can be extended to linear operator between Banach spaces, but this conclusion does not holds.

Typical example of compact operator are matrix-like operator with a continuous kernel $k(x, y)$ for $(x, y) \in \Omega$ where Ω is a compact sub-set of $\mathbb{R}^d$ (or the torus $\mathbb{T}^d$), i.e.

$$(\Phi f)(x) = \int_\Omega k(x, y) f(y) \mathrm{d}y$$

where $\mathrm{d}y$ is the Lebesgue measure. An example of such a setting which generalizes (8.5) is when $\Phi f = f \star h$ on $\mathbb{T}^d = (\mathbb{R}/2\pi\mathbb{Z})^d$, which is corresponds to a translation invariant kernel $k(x, y) = h(x - y)$, in which case $u_m(x) = (2\pi)^{-d/2} e^{\mathrm{i}\omega x}$, $\sigma_m = |\hat{f}_m|$. Another example on $\Omega = [0, 1]$ is the integration, $(\Phi f)(x) = \int_0^x f(y)\mathrm{d}y$, which corresponds to k being the indicator of the "triangle", $k(x, y) = 1_{x \leqslant y}$.

Pseudo inverse. In the case where $w = 0$, it makes to try to directly solve $\Phi f = y$. The two obstruction for this is that one not necessarily has $y \in \mathrm{Im}(\Phi)$ and even so, there are an infinite number of solutions if $\ker(\Phi) \neq \{0\}$. The usual workaround is to solve this equation in the least square sense

$$f^+ \overset{\text{def.}}{=} \operatorname*{argmin}_{\Phi f = y^+} \|f\|_{\mathcal{S}} \quad \text{where} \quad y^+ = \mathrm{Proj}_{\mathrm{Im}(\Phi)}(y) = \operatorname*{argmin}_{z \in \mathrm{Im}(\Phi)} \|y - z\|_{\mathcal{H}}.$$

The following proposition shows how to compute this least square solution using the SVD and by solving linear systems involving either $\Phi\Phi^*$ or $\Phi^*\Phi$.

Proposition 23. *One has*

$$f^+ = \Phi^+ y \quad \textit{where} \quad \Phi^+ = V \,\mathrm{diag}_m(1/\sigma_m) U^*. \tag{8.7}$$

In case that $\mathrm{Im}(\Phi) = \mathcal{H}$, *one has* $\Phi^+ = \Phi^*(\Phi\Phi^*)^{-1}$. *In case that* $\ker(\Phi) = \{0\}$, *one has* $\Phi^+ = (\Phi^*\Phi)^{-1}\Phi^*$.

Proof. Since U is an ortho-basis of $\mathrm{Im}(\Phi)$, $y^+ = UU^*y$, and thus $\Phi f = y^+$ reads $U\Sigma V^* f = UU^* y$ and hence $V^* f = \Sigma^{-1} U^* y$. Decomposition orthogonally $f = f_0 + r$ where $f_0 \in \ker(\Phi)^\perp$ and $r \in \ker(\Phi)$, one has $f_0 = VV^* f = V\Sigma^{-1}U^*y = \Phi^+ y$ is a constant. Minimizing $\|f\|^2 = \|f_0\|^2 + \|r\|^2$ is thus equivalent to minimizing $\|r\|$ and hence $r = 0$ which is the desired result. If $\mathrm{Im}(\Phi) = \mathcal{H}$, then $R = N$ so that $\Phi\Phi^* = U\Sigma^2 U^*$ is the eigen-decomposition of an invertible and $(\Phi\Phi^*)^{-1} = U\Sigma^{-2}U^*$. One then verifies $\Phi^*(\Phi\Phi^*)^{-1} = V\Sigma U^* U \Sigma^{-2} U^*$ which is the desired result. One deals similarly with the second case. □

For convolution operators $\Phi f = f \star h$, then

$$\Phi^+ y = y \star h^+ \quad \text{where} \quad \hat{h}^+_m = \begin{cases} \hat{h}_m^{-1} & \text{if} \quad \hat{h}_m \neq 0 \\ 0 & \text{if} \quad \hat{h}_m = 0. \end{cases} .$$

8.2.2 Tikonov Regularization

Regularized inverse. When there is noise, using formula (8.7) is not acceptable, because then

$$\Phi^+ y = \Phi^+ \Phi f_0 + \Phi^+ w = f_0^+ + \Phi^+ w \quad \text{where} \quad f_0^+ \overset{\text{def.}}{=} \text{Proj}_{\ker(\Phi)^\perp},$$

so that the recovery error is $\|\Phi^+ y - f_0^+\| = \|\Phi^+ w\|$. This quantity can be as larges as $\|w\|/\sigma_R$ if $w \propto u_R$. The noise is thus amplified by the inverse $1/\sigma_R$ of the smallest amplitude non-zero singular values, which can be very large. In infinite dimension, one typically has $R = +\infty$, so that the inverse is actually not bounded (discontinuous). It is thus mendatory to replace Φ^+ by a regularized approximate inverse, which should have the form

$$\Phi^+_\lambda = V \, \text{diag}_m(\mu_\lambda(\sigma_m)) U^* \tag{8.8}$$

where μ_λ, indexed by some parameter $\lambda > 0$, is a regularization of the inverse, that should typically satisfies

$$\mu_\lambda(\sigma) \leqslant C_\lambda < +\infty \quad \text{and} \quad \lim_{\lambda \to 0} \mu_\lambda(\sigma) = \frac{1}{\sigma}.$$

Figure 8.2, left, shows a typical example of such a regularized inverse curve, obtained by thresholding.

Variational regularization. A typical example of such regularized inverse is obtained by considering a penalized least square involving a regularization functional

$$f_\lambda \overset{\text{def.}}{=} \operatorname*{argmin}_{f \in \mathcal{S}} \|y - \Phi f\|^2_{\mathcal{H}} + \lambda J(f) \tag{8.9}$$

where J is some regularization functional which should at least be continuous on $\mathcal{S}$. The simplest example is the quadratic norm $J = \|\cdot\|^2_{\mathcal{S}}$,

$$f_\lambda \overset{\text{def.}}{=} \operatorname*{argmin}_{f \in \mathcal{S}} \|y - \Phi f\|^2_{\mathcal{H}} + \lambda \|f\|^2 \tag{8.10}$$

which is indeed a special case of (8.8) as proved in Proposition 24 below. In this case, the regularized solution is obtained by solving a linear system

$$f_\lambda = (\Phi^* \Phi + \lambda \text{Id}_P)^{-1} \Phi^* y. \tag{8.11}$$

This shows that $f_\lambda \in \text{Im}(\Phi^*) = \ker(\Phi)^\perp$, and that it depends linearly on y.

Proposition 24. *The solution of (8.10) has the form $f_\lambda = \Phi^+_\lambda y$ as defined in (8.8) for the specific choice of function*

$$\forall \, \sigma \in \mathbb{R}, \quad \mu_\lambda(\sigma) = \frac{\sigma}{\sigma^2 + \lambda}.$$

Proof. Using expression (8.11) and plugging the SVD $\Phi = U \Sigma V^*$ leads to

$$\Phi^+_\lambda = (V \Sigma^2 V^* + \lambda V V^*)^{-1} V \Sigma U^* = V^* (\Sigma^2 + \lambda)^{-1} \Sigma U^*$$

which is the desired expression since $(\Sigma^2 + \lambda)^{-1} \Sigma = \text{diag}(\mu_\lambda(\sigma_m))_m$. □

A special case is when $\Phi f = f \star h$ is a convolution operator. In this case, the regularized inverse is computed in $O(N \log(N))$ operation using the FFT as follow

$$\hat{f}_{\lambda,m} = \frac{\hat{h}_m^*}{|\hat{h}_m|^2 + \sigma^2} \hat{y}_m.$$

Figure 8.2 contrast the regularization curve associated to quadratic regularization (8.11) (right) to the simpler thresholding curve (left).

The question is to understand how to choose λ as a function of the noise level $\|w\|_{\mathcal{H}}$ in order to guarantees that $f_\lambda \to f_0$ and furthermore establish convergence speed. One first needs to ensure at least $f_0 = f_0^+$, which in turns requires that $f_0 \in \mathrm{Im}(\Phi^*) = \ker(\Phi)^\perp$. Indeed, an important drawback of linear recovery methods (such as quadratic regularization) is that necessarily $f_\lambda \in \mathrm{Im}(\Phi^*) = \ker(\Phi^\perp)$ so that no information can be recovered inside $\ker(\Phi)$. Non-linear methods must be used to achieve a "super-resoltution" effect and recover this missing information.

Source condition. In order to ensure convergence speed, one quantify this condition and impose a so-called source condition of order β, which reads

$$f_0 \in \mathrm{Im}((\Phi^*\Phi)^\beta) = \mathrm{Im}(V \,\mathrm{diag}(\sigma_m^{2\beta}) V^*). \tag{8.12}$$

In some sense, the larger β, the farther f_0 is away from $\ker(\Phi)$, and thus the inversion problem is "easy". This condition means that there should exists $z \in \mathbb{R}^P$ such that $f_0 = V \,\mathrm{diag}(\sigma_m^{2\beta}) V^* z$, i.e. $z = V \,\mathrm{diag}(\sigma_m^{-2\beta}) V^* f_0$. In order to control the strength of this source condition, we assume $\|z\| \leqslant \rho$ where $\rho > 0$. The source condition thus corresponds to the following constraint

$$\sum_m \sigma_m^{-2\beta} \langle f_0,\, v_m \rangle^2 \leqslant \rho^2 < +\infty. \tag{$S_{\beta,\rho}$}$$

This is a Sobolev-type constraint, similar to those imposed in 6.4. A prototypical example is for a low-pass filter $\Phi f = f \star h$ where h as a slow polynomial-like decay of frequency, i.e. $|\hat{h}_m| \sim 1/m^\alpha$ for large m. In this case, since v_m is the Fourier basis, the source condition ($S_{\beta,\rho}$) reads

$$\sum_m \|m\|^{2\alpha\beta} |\hat{f}_m|^2 \leqslant \rho^2 < +\infty,$$

which is a Sobolev ball of radius ρ and differential order $\alpha\beta$.

Sublinear convergence speed. The following theorem shows that this source condition leads to a convergence speed of the regularization. Imposing a bound $\|w\| \leqslant \delta$ on the noise, the theoretical analysis of the inverse problem thus depends on the parameters (δ, ρ, β). Assuming $f_0 \in \ker(\Phi)^\perp$, the goal of the theoretical analysis corresponds to studying the speed of convergence of f_λ toward f_0, when using $y = \Phi f_0 + w$ as $\delta \to 0$. This requires to decide how λ should depends on δ.

Theorem 10. *Assuming the source condition ($S_{\beta,\rho}$) with $0 < \beta \leqslant 2$, then the solution of (8.10) for $\|w\| \leqslant \delta$ satisfies*

$$\|f_\lambda - f_0\| \leqslant C \rho^{\frac{1}{\beta+1}} \delta^{\frac{\beta}{\beta+1}}$$

for a constant C which depends only on β, and for a choice

$$\lambda \sim \delta^{\frac{2}{\beta+1}} \rho^{-\frac{2}{\beta+1}}.$$

Proof. Because of the source condition, $f_0 \in \mathrm{Im}(\Phi^*)$. We decompose

$$f_\lambda = \Phi_\lambda^+(\Phi f_0 + w) = f_\lambda^0 + \Phi_\lambda^+ w \quad \text{where} \quad f_\lambda^0 \overset{\text{def.}}{=} \Phi_\lambda^+(\Phi f_0),$$

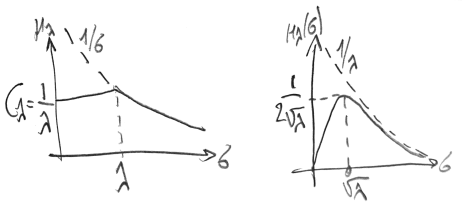


Figure 8.2: Bounding $\mu_\lambda(\sigma) \leqslant C_\lambda = \frac{1}{2\sqrt{\lambda}}$.

so that $f_\lambda = f_\lambda^0 + \Phi_\lambda^+ w$, one has for any regularized inverse of the form (8.8)

$$\|f_\lambda - f_0\| \leqslant \|f_\lambda - f_\lambda^0\| + \|f_\lambda^0 - f_0\|. \tag{8.13}$$

The term $\|f_\lambda - f_\lambda^0\|$ is a variance term which account for residual noise, and thus decays when λ increases (more regularization). The term $\|f_\lambda^0 - f_0\|$ is independent of the noise, it is a bias term coming from the approximartion (smoothing) of f_0, and thus increases when λ increases. The choice of an optimal λ thus results in a bias-variance tradeoff between these two terms. Assuming

$$\forall\, \sigma \geqslant 0, \quad \mu_\lambda(\sigma) \leqslant C_\lambda$$

the variance term is bounded as

$$\|f_\lambda - f_\lambda^0\|^2 = \|\Phi_\lambda^+ w\|^2 = \sum_m \mu_\lambda(\sigma_m)^2 w_m^2 \leqslant C_\lambda^2 \|w\|_{\mathcal{H}}^2.$$

The bias term is bounded as, since $\frac{f_{0,m}^2}{\sigma_m^{2\beta}} = z_m^2$,

$$\|f_\lambda^0 - f_0\|^2 = \sum_m (1 - \mu_\lambda(\sigma_m)\sigma_m)^2 f_{0,m}^2 = \sum_m \left((1 - \mu_\lambda(\sigma_m)\sigma_m)\sigma_m^\beta\right)^2 \frac{f_{0,m}^2}{\sigma_m^{2\beta}} \leqslant D_{\lambda,\beta}^2 \rho^2 \tag{8.14}$$

where we assumed

$$\forall\, \sigma \geqslant 0, \quad \left|(1 - \mu_\lambda(\sigma)\sigma)\sigma^\beta\right| \leqslant D_{\lambda,\beta}. \tag{8.15}$$

Note that for $\beta > 2$, one has $D_{\lambda,\beta} = +\infty$ Putting (8.14) and (8.15) together, one obtains

$$\|f_\lambda - f_0\| \leqslant C_\lambda \delta + D_{\lambda,\beta}\rho. \tag{8.16}$$

In the case of the regularization (8.10), one has $\mu_\lambda(\sigma) = \frac{\sigma}{\sigma^2+\lambda}$, and thus $(1 - \mu_\lambda(\sigma)\sigma)\sigma^\beta = \frac{\lambda\sigma^\beta}{\sigma^2+\lambda}$. For $\beta \leqslant 2$, one verifies (see Figure 8.2 and 12.9) that

$$C_\lambda = \frac{1}{2\sqrt{\lambda}} \quad \text{and} \quad D_{\lambda,\beta} = c_\beta \lambda^{\frac{\beta}{2}},$$

for some constant c_β. Equalizing the contributions of the two terms in (8.16) (a better constant would be reached by finding the best λ) leads to selecting $\frac{\delta}{\sqrt{\lambda}} = \lambda^{\frac{\beta}{2}}\rho$ i.e. $\lambda = (\delta/\rho)^{\frac{2}{\beta+1}}$. With this choice,

$$\|f_\lambda - f_0\| = O(\delta/\sqrt{\lambda}) = O(\delta(\delta/\rho)^{-\frac{1}{\beta+1}}) = O(\delta^{\frac{\beta}{\beta+1}}\rho^{\frac{1}{\beta+1}}).$$

□

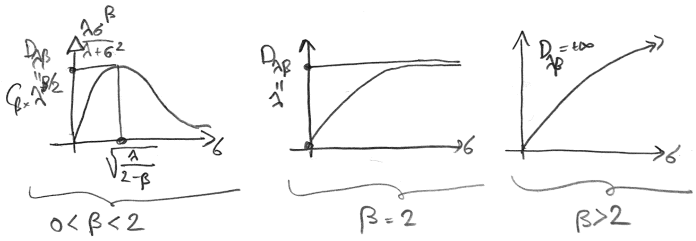


Figure 8.3: Bounding $\lambda \frac{\sigma^\beta}{\lambda+\sigma^2} \leqslant D_{\lambda,\beta}$.

This theorem shows that using larger $\beta \leqslant 2$ leads to faster convergence rates as $\|w\|$ drops to zero. The rate (8.13) however suffers from a "saturation" effect, indeed, choosing $\beta > 2$ does not helps (it gives the same rate as $\beta = 2$), and the best possible rate is thus

$$\|f_\lambda - f_0\| = O(\rho^{\frac{1}{3}} \delta^{\frac{2}{3}}).$$

By choosing more alternative regularization functional μ_λ and choosing β large enough, one can show that it is possible to reach rate $\|f_\lambda - f_0\| = O(\delta^{1-\kappa})$ for an arbitrary small $\kappa > 0$. Figure 8.2 contrast the regularization curve associated to quadratic regularization (8.11) (right) to the simpler thresholding curve (left) which does not suffers from saturation. Quadratic regularization however is much simpler to implement because it does not need to compute an SVD, is defined using a variational optimization problem and is computable as the solution of a linear system. One cannot however reach a linear rate $\|f_\lambda - f_0\| = O(\|w\|)$. Such rates are achievable using non-linear sparse ℓ^1 regularizations as detailed in Chapter 9.

8.3 Quadratic Regularization

After this theoretical study in infinite dimension, we now turn our attention to more practical matters, and focus only on the finite dimensional setting.

Convex regularization. Following (8.9), the ill-posed problem of recovering an approximation of the high resolution image $f_0 \in \mathbb{R}^N$ from noisy measures $y = \Phi f_0 + w \in \mathbb{R}^P$ is regularized by solving a convex optimization problem

$$f_\lambda \in \operatorname*{argmin}_{f \in \mathbb{R}^N} \mathcal{E}(f) \stackrel{\text{def.}}{=} \frac{1}{2}\|y - \Phi f\|^2 + \lambda J(f) \tag{8.17}$$

where $\|y - \Phi f\|^2$ is the data fitting term (here $\|\cdot\|$ is the ℓ^2 norm on $\mathbb{R}^P$) and $J(f)$ is a convex functional on $\mathbb{R}^N$.

The Lagrange multiplier λ weights the importance of these two terms, and is in practice difficult to set. Simulation can be performed on high resolution signal f_0 to calibrate the multiplier by minimizing the super-resolution error $\|f_0 - \tilde{f}\|$, but this is usually difficult to do on real life problems.

In the case where there is no noise, $w = 0$, the Lagrange multiplier λ should be set as small as possible. In the limit where $\lambda \to 0$, the unconstrained optimization problem (8.17) becomes a constrained optimization, as the following proposition explains. Let us stress that, without loss of generality, we can assume that $y \in \mathrm{Im}(\Phi)$, because one has the orthogonal decomposition

$$\|y - \Phi f\|^2 = \|y - \mathrm{Proj}_{\mathrm{Im}(\Phi)}(y)\|^2 + \|\mathrm{Proj}_{\mathrm{Im}(\Phi)}(y) - \Phi f\|^2$$

so that one can replace y by $\mathrm{Proj}_{\mathrm{Im}(\Phi)}(y)$ in (8.17).

Let us recall that a function J is coercive if

$$\lim_{\|f\| \to +\infty} J(f) = +\infty$$

i.e.

$$\forall\, K,\ \exists R, \|x\| \geqslant R \quad \Longrightarrow \quad |J(f)| \geqslant K.$$

This means that its non-empty levelsets $\{f\ ;\ J(f) \leqslant c\}$ are bounded (and hence compact) for all c.

Proposition 25. *We assume that J is coercive and that $y \in \mathrm{Im}(\Phi)$. Then, if for each λ, f_λ is a solution of (8.17), then $(f_\lambda)_\lambda$ is a bounded set and any accumulation point $f^\star$ is a solution of*

$$f^\star = \operatorname*{argmin}_{f \in \mathbb{R}^N} \{J(f)\ ;\ \Phi f = y\}. \tag{8.18}$$

Proof. Denoting h, any solution to (8.18), which in particular satisfies $\Phi h = y$, because of the optimality condition of f_λ for (8.17), one has

$$\frac{1}{2\lambda}\|\Phi f_\lambda - y\|^2 + J(f_\lambda) \leqslant \frac{1}{2\lambda}\|\Phi h - y\|^2 + J(h) = J(h).$$

This shows that $J(f_\lambda) \leqslant J(h)$ so that since J is coercive the set $(f_\lambda)_\lambda$ is bounded and thus one can consider an accumulation point $f_{\lambda_k} \to f^\star$ for $k \to +\infty$. Since $\|\Phi f_{\lambda_k} - y\|^2 \leqslant \lambda_k J(h)$, one has in the limit $\Phi f^\star = y$, so that $f^\star$ satisfies the constraints in (8.18). Furthermore, by continuity of J, passing to the limit in $J(f_{\lambda_k}) \leqslant J(h)$, one obtains $J(f^\star) \leqslant J(h)$ so that $f^\star$ is a solution of (8.18). □

Note that it is possible to extend this proposition in the case where J is not necessarily coercive on the full space (for instance the TV functionals in Section 8.4.1 below) but on the orthogonal to $\ker(\Phi)$. The proof is more difficult.

Quadratic Regularization. The simplest class of prior functional are quadratic, and can be written as

$$J(f) = \frac{1}{2}\|Gf\|_{\mathbb{R}^K}^2 = \frac{1}{2}\langle Lf,\, f\rangle_{\mathbb{R}^N} \tag{8.19}$$

where $G \in \mathbb{R}^{K\times N}$ and where $L = G^*G \in \mathbb{R}^{N\times N}$ is a positive semi-definite matrix. The special case (8.10) is recovered when setting $G = L = \mathrm{Id}_N$.

Writing down the first order optimality conditions for (8.17) leads to

$$\nabla \mathcal{E}(f) = \Phi^*(\Phi f - y) + \lambda L f = 0,$$

hence, if

$$\ker(\Phi) \cap \ker(G) = \{0\},$$

then (8.19) has a unique minimizer f_λ, which is obtained by solving a linear system

$$f_\lambda = (\Phi^*\Phi + \lambda L)^{-1}\Phi^* y. \tag{8.20}$$

In the special case where L is diagonalized by the singular basis $(v_m)_m$ of Φ, i.e. $L = V\,\mathrm{diag}(\alpha_m^2)V^*$, then f_λ reads in this basis

$$\langle f_\lambda,\, v_m\rangle = \frac{\sigma_m}{\sigma_m^2 + \lambda\alpha_m^2}\langle y,\, v_m\rangle. \tag{8.21}$$

Example of convolution. A specific example is for convolution operator

$$\Phi f = h \star f, \tag{8.22}$$

and using $G = \nabla$ be a discretization of the gradient operator, such as for instance using first order finite differences (2.16). This corresponds to the discrete Sobolev prior introduced in Section 7.1.2. Such an operator compute, for a d-dimensional signal $f \in \mathbb{R}^N$ (for instance a 1-D signal for $d = 1$ or an image when $d = 2$), an approximation $\nabla f_n \in \mathbb{R}^d$ of the gradient vector at each sample location n. Thus typically,

$\nabla : f \mapsto (\nabla f_n)_n \in \mathbb{R}^{N\times d}$ maps to d-dimensional vector fields. Then $-\nabla^* : \mathbb{R}^{N\times d} \to \mathbb{R}^N$ is a discretized divergence operator. In this case, $\Delta = -GG^*$ is a discretization of the Laplacian, which is itself a convolution operator. One then has

$$\hat{f}_{\lambda,m} = \frac{\hat{h}_m^* \hat{y}_m}{|\hat{h}_m|^2 - \lambda \hat{d}_{2,m}}, \tag{8.23}$$

where $\hat{d}_2$ is the Fourier transform of the filter d_2 corresponding to the Laplacian. For instance, in dimension 1, using first order finite differences, the expression for $\hat{d}_{2,m}$ is given in (2.18).

8.3.1 Solving Linear System

When Φ and L do not share the same singular spaces, using (8.21) is not possible, so that one needs to solve the linear system (8.20), which can be rewritten as

$$Af = b \quad \text{where} \quad A \overset{\text{def.}}{=} \Phi^*\Phi + \lambda L \quad \text{and} \quad b = \Phi^* y.$$

It is possible to solve exactly this linear system with direct methods for moderate N (up to a few thousands), and the numerical complexity for a generic A is $O(N^3)$. Since the involved matrix A is symmetric, the best option is to use Choleski factorization $A = BB^*$ where B is lower-triangular. In favorable cases, this factorization (possibly with some re-re-ordering of the row and columns) can take advantage of some sparsity in A.

For large N, such exact resolution is not an option, and should use approximate iterative solvers, which only access A through matrix-vector multiplication. This is especially advantageous for imaging applications, where such multiplications are in general much faster than a naive $O(N^2)$ explicit computation. If the matrix A is highly sparse, this typically necessitates $O(N)$ operations. In the case where A is symmetric and positive definite (which is the case here), the most well known method is the conjugate gradient methods, which is actually an optimization method solving

$$\min_{f\in\mathbb{R}^N} \mathcal{E}(f) \overset{\text{def.}}{=} \mathcal{Q}(f) \overset{\text{def.}}{=} \langle Af,\, f\rangle - \langle f,\, b\rangle \tag{8.24}$$

which is equivalent to the initial minimization (8.17). Instead of doing a naive gradient descent (as studied in Section 17.1 below), stating from an arbitrary $f^{(0)}$, it compute a new iterate $f^{(\ell+1)}$ from the previous iterates as

$$f^{(\ell+1)} \overset{\text{def.}}{=} \operatorname*{argmin}_f \left\{ \mathcal{E}(f) \;;\; f \in f^{(\ell)} + \operatorname{Span}(\nabla\mathcal{E}(f^{(0)}), \ldots, \nabla\mathcal{E}(f^{\ell})) \right\}.$$

The crucial and remarkable fact is that this minimization can be computed in closed form at the cost of two matrix-vector product per iteration, for $k \geqslant 1$ (posing initially $d^{(0)} = \nabla\mathcal{E}(f^{(0)}) = Af^{(0)} - b$)

$$f^{(\ell+1)} = f^{(\ell)} - \tau_\ell d^{(\ell)} \quad \text{where} \quad d^{(\ell)} = g_\ell + \frac{\|g^{(\ell)}\|^2}{\|g^{(\ell-1)}\|^2} d^{(\ell-1)} \quad \text{and} \quad \tau_\ell = \frac{\langle g_\ell,\, d^{(\ell)}\rangle}{\langle Ad^{(\ell)},\, d^{(\ell)}\rangle} \tag{8.25}$$

$g^{(\ell)} \overset{\text{def.}}{=} \nabla\mathcal{E}(f^{(\ell)}) = Af^{(\ell)} - b$. It can also be shown that the direction $d^{(\ell)}$ are orthogonal, so that after $\ell = N$ iterations, the conjugate gradient computes the unique solution $f^{(\ell)}$ of the linear system $Af = b$. It is however rarely used this way (as an exact solver), and in practice much less than N iterates are computed. It should also be noted that iterations (8.25) can be carried over for an arbitrary smooth convex function $\mathcal{E}$, and it typically improves over the gradient descent (although in practice quasi-Newton method are often preferred).

8.4 Non-Quadratic Regularization

8.4.1 Total Variation Regularization

A major issue with quadratic regularization such as (8.19) is that they typically leads to blurry recovered data f_λ, which is thus not a good approximation of f_0 when it contains sharp transition such as edges in

images. This is can clearly be seen in the convolutive case (8.23), this the restoration operator $\Phi_\lambda^+\Phi$ is a filtering, which tends to smooth sharp part of the data.

This phenomena can also be understood because the restored data f_λ always belongs to $\mathrm{Im}(\Phi^*) = \ker(\Phi)^\perp$, and thus cannot contains "high frequency" details that are lost in the kernel of Φ. To alleviate this shortcoming, and recover missing information in the kernel, it is thus necessarily to consider non-quadratic and in fact non-smooth regularization.

Total variation. The most well know instance of such a non-quadratic and non-smooth regularization is the total variation prior. For smooth function $f : \mathbb{R}^d \mapsto \mathbb{R}$, this amounts to replacing the quadratic Sobolev energy (often called Dirichlet energy)

$$J_{\mathrm{Sob}}(f) \overset{\text{def.}}{=} \frac{1}{2}\int_{\mathbb{R}^d} \|\nabla f\|_{\mathbb{R}^d}^2 \mathrm{d}x,$$

where $\nabla f(x) = (\partial_{x_1} f(x), \ldots, \partial_{x_d} f(x))^\top$ is the gradient, by the (vectorial) L^1 norm of the gradient

$$J_{\mathrm{TV}}(f) \overset{\text{def.}}{=} \int_{\mathbb{R}^d} \|\nabla f\|_{\mathbb{R}^d} \mathrm{d}x.$$

We refer also to Section 7.1.1 about these priors. Simply "removing" the square ^{2} inside the integral might seems light a small change, but in fact it is a game changer.

Indeed, while $J_{\mathrm{Sob}}(1_\Omega) = +\infty$ where 1_Ω is the indicator a set Ω with finite perimeter $|\Omega| < +\infty$, one can show that $J_{\mathrm{TV}}(1_\Omega) = |\Omega|$, if one interpret ∇f as a distribution Df (actually a vectorial Radon measure) and $\int_{\mathbb{R}^d} \|\nabla f\|_{\mathbb{R}^d} \mathrm{d}x$ is replaced by the total mass $|Df|(\Omega)$ of this distribution $m = Df$

$$|m|(\Omega) = \sup\left\{ \int_{\mathbb{R}^d} \langle h(x), \mathrm{d}m(x)\rangle \;;\; h \in \mathcal{C}(\mathbb{R}^d \mapsto \mathbb{R}^d), \forall\, x, \|h(x)\| \leqslant 1 \right\}.$$

The total variation of a function such that Df has a bounded total mass (a so-called bounded variation function) is hence defined as

$$J_{\mathrm{TV}}(f) \overset{\text{def.}}{=} \sup\left\{ \int_{\mathbb{R}^d} f(x)\, \mathrm{div}(h)(x) \mathrm{d}x \;;\; h \in \mathcal{C}_c^1(\mathbb{R}^d;\mathbb{R}^d), \|h\|_\infty \leqslant 1 \right\}.$$

Generalizing the fact that $J_{\mathrm{TV}}(1_\Omega) = |\Omega|$, the functional co-area formula reads

$$J_{\mathrm{TV}}(f) = \int_{\mathbb{R}} \mathcal{H}_{d-1}(L_t(f))\mathrm{d}t \quad \text{where} \quad L_t(f) = \{x \;;\; f(x) = t\}$$

and where $\mathcal{H}_{d-1}$ is the Hausforf measures of dimension $d-1$, for instance, for $d = 2$ if L has finite perimeter $|L|$, then $\mathcal{H}_{d-1}(L) = |L|$ is the perimeter of L.

Discretized Total variation. For discretized data $f \in \mathbb{R}^N$, one can define a discretized TV semi-norm as detailed in Section 7.1.2, and it reads, generalizing (7.6) to any dimension

$$J_{\mathrm{TV}}(f) = \sum_n \|\nabla f_n\|_{\mathbb{R}^d}$$

where $\nabla f_n \in \mathbb{R}^d$ is a finite difference gradient at location indexed by n.

The discrete total variation prior $J_{\mathrm{TV}}(f)$ defined in (7.6) is a convex but non differentiable function of f, since a term of the form $\|\nabla f_n\|$ is non differentiable if $\nabla f_n = 0$. We defer to chapters 16 and 17 the study of advanced non-smooth convex optimization technics that allows to handle this kind of functionals.

In order to be able to use simple gradient descent methods, one needs to smooth the TV functional. The general machinery proceeds by replacing the non-smooth ℓ^2 Euclidean norm $\|\cdot\|$ by a smoothed version, for instance

$$\forall\, u \in \mathbb{R}^d, \quad \|u\|_\varepsilon \overset{\text{def.}}{=} \sqrt{\varepsilon^2 + \|u\|}.$$

This leads to the definition of a smoothed approximate TV functional, already introduced in (7.12),

$$J_{\mathrm{TV}}^{\varepsilon}(f) \stackrel{\text{def.}}{=} \sum_n \|\nabla f_n\|_{\varepsilon}$$

One has the following asymptotics for $\varepsilon \to \{0, +\infty\}$

$$\|u\|_{\varepsilon} \stackrel{\varepsilon \to 0}{\longrightarrow} \|u\| \quad \text{and} \quad \|u\|_{\varepsilon} = \varepsilon + \frac{1}{2\varepsilon}\|u\|^2 + O(1/\varepsilon^2)$$

which suggest that $J_{\mathrm{TV}}^{\varepsilon}$ interpolates between J_{TV} and J_{Sob}.

The resulting inverse regularization problem (8.17) thus reads

$$f_{\lambda} \stackrel{\text{def.}}{=} \operatorname*{argmin}_{f \in \mathbb{R}^N} \mathcal{E}(f) = \frac{1}{2}\|y - \Phi f\|^2 + \lambda J_{\mathrm{TV}}^{\varepsilon}(f) \tag{8.26}$$

It is a strictly convex problem (because $\|\cdot\|_{\varepsilon}$ is strictly convex for $\varepsilon > 0$) so that its solution f_{λ} is unique.

8.4.2 Gradient Descent Method

The optimization program (8.26) is a example of smooth unconstrained convex optimization of the form

$$\min_{f \in \mathbb{R}^N} \mathcal{E}(f) \tag{8.27}$$

where $\mathcal{E} : \mathbb{R}^N \to \mathbb{R}$ is a $\mathcal{C}^1$ function. Recall that the gradient $\nabla \mathcal{E} : \mathbb{R}^N \mapsto \mathbb{R}^N$ of this functional (not to be confound with the discretized gradient $\nabla f \in \mathbb{R}^N$ of f) is defined by the following first order relation

$$\mathcal{E}(f + r) = \mathcal{E}(f) + \langle f, r \rangle_{\mathbb{R}^N} + O(\|r\|_{\mathbb{R}^N}^2)$$

where we used $O(\|r\|_{\mathbb{R}^N}^2)$ in place of $o(\|r\|_{\mathbb{R}^N})$ (for differentiable function) because we assume here $\mathcal{E}$ is of class $\mathcal{C}^1$ (i.e. the gradient is continuous).

For such a function, the gradient descent algorithm is defined as

$$f^{(\ell+1)} \stackrel{\text{def.}}{=} f^{(\ell)} - \tau_{\ell} \nabla \mathcal{E}(f^{(\ell)}), \tag{8.28}$$

where the step size $\tau_{\ell} > 0$ should be small enough to guarantee convergence, but large enough for this algorithm to be fast.

We refer to Section 17.1 for a detailed analysis of the convergence of the gradient descent, and a study of the influence of the step size τ_{ℓ}.

8.4.3 Examples of Gradient Computation

Note that the gradient of a quadratic function $\mathcal{Q}(f)$ of the form (8.24) reads

$$\nabla \mathcal{G}(f) = Af - b.$$

In particular, one retrieves that the first order optimality condition $\nabla \mathcal{G}(f) = 0$ is equivalent to the linear system $Af = b$.

For the quadratic fidelity term $\mathcal{G}(f) = \frac{1}{2}\|\Phi f - y\|^2$, one thus obtains

$$\nabla \mathcal{G}(f) = \Phi^*(\Phi y - y).$$

In the special case of the regularized TV problem (8.26), the gradient of $\mathcal{E}$ reads

$$\nabla \mathcal{E}(f) = \Phi^*(\Phi y - y) + \lambda \nabla J_{\mathrm{TV}}^{\varepsilon}(f).$$

Recall the chain rule for differential reads $\partial(\mathcal{G}_1 \circ \mathcal{G}_2) = \partial\mathcal{G}_1 \circ \partial\mathcal{G}_2$, but that gradient vectors are actually transposed of differentials, so that for $\mathcal{E} = \mathcal{F} \circ \mathcal{H}$ where $\mathcal{F} : \mathbb{R}^P \to \mathbb{R}$ and $\mathcal{H} : \mathbb{R}^N \to \mathbb{R}^P$, one has

$$\nabla\mathcal{E}(f) = [\partial\mathcal{H}(f)]^* (\nabla\mathcal{F}(\mathcal{H}f)) .$$

Since $J^\varepsilon_{\mathrm{TV}} = \|\cdot\|_{1,\varepsilon} \circ \nabla$, one thus has

$$\nabla J^\varepsilon_{\mathrm{TV}} = \nabla^\star \circ (\partial\|\cdot\|_{1,\varepsilon}) \quad \text{where} \quad \|u\|_{1,\varepsilon} = \sum_n \|u_n\|_\varepsilon$$

so that

$$J^\varepsilon_{\mathrm{TV}}(f) = -\operatorname{div}(\mathcal{N}^\varepsilon(\nabla f)),$$

where $\mathcal{N}^\varepsilon(u) = (u_n/\|u_n\|_\varepsilon)_n$ is the smoothed-normalization operator of vector fields (the differential of $\|\cdot\|_{1,\varepsilon}$), and where $\operatorname{div} = -\nabla^*$ is minus the adjoint of the gradient.

Since $\operatorname{div} = -\nabla^*$, their Lipschitz constants are equal $\|\operatorname{div}\|_{\mathrm{op}} = \|\nabla\|_{\mathrm{op}}$, and is for instance equal to $\sqrt{2d}$ for the discretized gradient operator. Computation shows that the Hessian of $\|\cdot\|_\varepsilon$ is bounded by $1/\varepsilon$, so that for the smoothed-TV functional, the Lipschitz constant of the gradient is upper-bounded by

$$L = \frac{\|\nabla\|^2}{\varepsilon} + \|\Phi\|^2_{\mathrm{op}}.$$

Furthermore, this functional is strongly convex because $\|\cdot\|_\varepsilon$ is ε-strongly convex, and the Hessian is lower bounded by

$$\mu = \varepsilon + \sigma_{\min}(\Phi)^2$$

where $\sigma_{\min}(\Phi)$ is the smallest singular value of Φ. For ill-posed problems, typically $\sigma_{\min}(\Phi) = 0$ or is very small, so that both L and μ degrades (tends respectively to 0 and $+\infty$) as $\varepsilon \to 0$, so that gradient descent becomes prohibitive for small ε, and it is thus required to use dedicated non-smooth optimization methods detailed in the following chapters. On the good news side, note however that in many case, using a small but non-zero value for ε often leads to better a visually more pleasing results, since it introduce a small blurring which diminishes the artifacts (and in particular the so-called "stair-casing" effect) of TV regularization.

8.5 Examples of Inverse Problems

We detail here some inverse problem in imaging that can be solved using quadratic regularization or non-linear TV.

8.5.1 Deconvolution

The blurring operator (8.1) is diagonal over Fourier, so that quadratic regularization are easily solved using Fast Fourier Transforms when considering periodic boundary conditions. We refer to (8.22) and the correspond explanations. TV regularization in contrast cannot be solved with fast Fourier technics, and is thus much slower.

8.5.2 Inpainting

For the inpainting problem, the operator defined in (8.3) is diagonal in space

$$\Phi = \operatorname{diag}_m(\delta_{\Omega^c}[m]),$$

and is an orthogonal projector $\Phi^* = \Phi$.

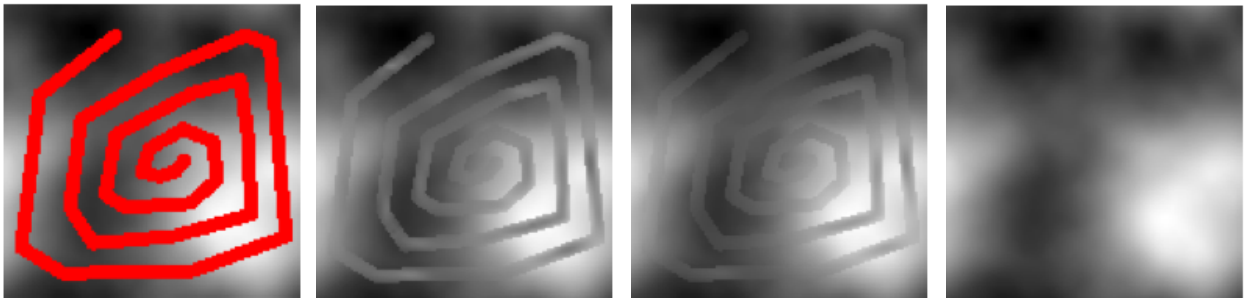
In the noiseless case, to constrain the solution to lie in the affine space $\{f \in \mathbb{R}^N \,;\, y = \Phi f\}$, we use the orthogonal projector

$$\forall\, x, \quad P_y(f)(x) = \begin{cases} f(x) & \text{if} \quad x \in \Omega, \\ y(x) & \text{if} \quad x \notin \Omega. \end{cases}$$

In the noiseless case, the recovery (8.18) is solved using a projected gradient descent. For the Sobolev energy, the algorithm iterates

$$f^{(\ell+1)} = P_y(f^{(\ell)} + \tau \Delta f^{(\ell)}).$$

which converges if $\tau < 2/\|\Delta\| = 1/4$. Figure 8.4 shows some iteration of this algorithm, which progressively interpolate within the missing area.



$k = 1$ $k = 10$ $k = 20$ $k = 100$

Figure 8.4: Sobolev projected gradient descent algorithm.

Figure 8.5 shows an example of Sobolev inpainting to achieve a special effect.

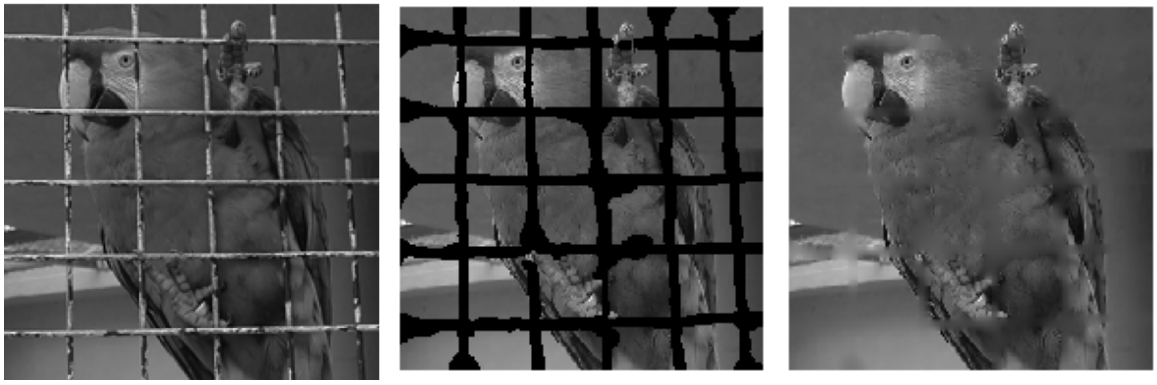


Image f_0 Observation $y = \Phi f_0$ Sobolev $f^\star$

Figure 8.5: Inpainting the parrot cage.

For the smoothed TV prior, the gradient descent reads

$$f^{(\ell+1)} = P_y\left(f^{(\ell)} + \tau \operatorname{div}\left(\frac{\nabla f^{(\ell)}}{\sqrt{\varepsilon^2 + \|\nabla f^{(\ell)}\|^2}}\right)\right)$$

which converges if $\tau < \varepsilon/4$.

Figure 8.6 compare the Sobolev inpainting and the TV inpainting for a small value of ε. The SNR is not improved by the total variation, but the result looks visually slightly better.

8.5.3 Tomography Inversion

In medical imaging, a scanner device compute projection of the human body along rays $\Delta_{t,\theta}$ defined

$$x \cdot \tau_\theta = x_1 \cos\theta + x_2 \sin\theta = t$$

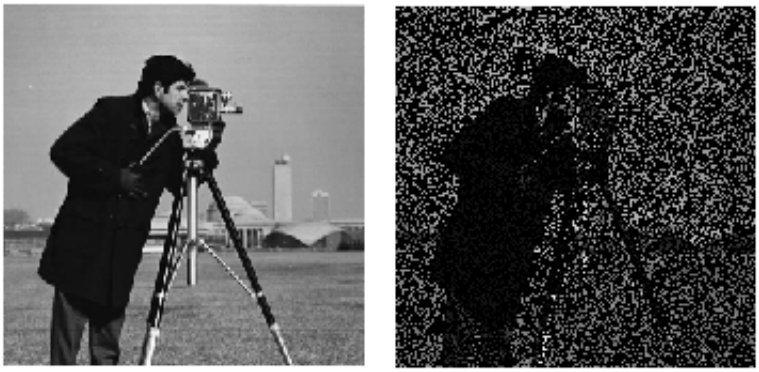


Image f_0 | Observation $y = \Phi f_0$

Sobolev $f^\star$
SNR=?dB | TV $f^\star$
SNR=?dB

Figure 8.6: Inpainting with Sobolev and TV regularization.

where we restrict ourself to 2D projection to simplify the exposition.

The scanning process computes a Radon transform, which compute the integral of the function to acquires along rays

$$\forall\, \theta \in [0,\pi), \forall\, t \in \mathbb{R}, \quad p_\theta(t) = \int_{\Delta_{t,\theta}} f(x)\, ds = \iint f(x)\, \delta(x \cdot \tau_\theta - t)\, dx$$

see Figure (8.7)

The Fourier slice theorem relates the Fourier transform of the scanned data to the 1D Fourier transform of the data along rays

$$\forall\, \theta \in [0,\pi)\ ,\ \forall\, \xi \in \mathbb{R} \quad \hat{p}_\theta(\xi) = \hat{f}(\xi\cos\theta, \xi\sin\theta). \tag{8.29}$$

This shows that the pseudo inverse of the Radon transform is computed easily over the Fourier domain using inverse 2D Fourier transform

$$f(x) = \frac{1}{2\pi} \int_0^\pi p_\theta \star h(x \cdot \tau_\theta)\, d\theta$$

with $\hat{h}(\xi) = |\xi|$.

Imaging devices only capture a limited number of equispaced rays at orientations $\{\theta_k = \pi/k\}_{0 \leqslant k < K}$. This defines a tomography operator which corresponds to a partial Radon transform

$$Rf = (p_{\theta_k})_{0 \leqslant k < K}.$$

Relation (8.29) shows that knowing Rf is equivalent to knowing the Fourier transform of f along rays,

$$\{\hat{f}(\xi\cos(\theta_k), \xi\sin(\theta_k))\}_k.$$

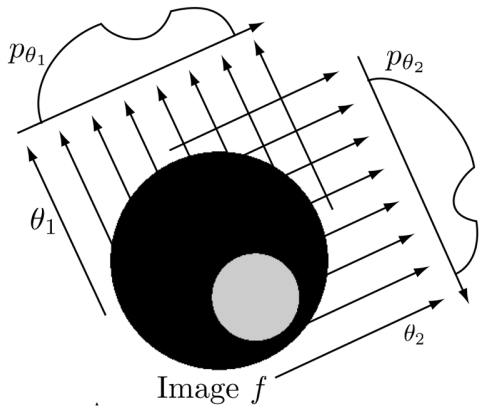


Figure 8.7: Principle of tomography acquisition.

We thus simply the acquisition process over the discrete domain and model it as computing directly samples of the Fourier transform

$$\Phi f = (\hat{f}[\omega])_{\omega \in \Omega} \in \mathbb{R}^P$$

where Ω is a discrete set of radial lines in the Fourier plane, see Figure 8.8, right.

In this discrete setting, recovering from Tomography measures $y = Rf_0$ is equivalent in this setup to inpaint missing Fourier frequencies, and we consider partial noisy Fourier measures

$$\forall\, \omega \in \Omega, \quad y[\omega] = \hat{f}[\omega] + w[\omega]$$

where $w[\omega]$ is some measurement noise, assumed here to be Gaussian white noise for simplicity.

The peuso-inverse $f^+ = R^+ y$ defined in (8.7) of this partial Fourier measurements reads

$$\hat{f}^+[\omega] = \begin{cases} y[\omega] & \text{if} \quad \omega \in \Omega, \\ 0 & \text{if} \quad \omega \notin \Omega. \end{cases}$$

Figure 8.9 shows examples of pseudo inverse reconstruction for increasing size of Ω. This reconstruction exhibit serious artifact because of bad handling of Fourier frequencies (zero padding of missing frequencies).

The total variation regularization (**??**) reads

$$f^\star \in \operatorname*{argmin}_f \frac{1}{2} \sum_{\omega \in \Omega} |y[\omega] - \hat{f}[\omega]|^2 + \lambda \|f\|_{\mathrm{TV}}.$$

It is especially suitable for medical imaging where organ of the body are of relatively constant gray value, thus resembling to the cartoon image model introduced in Section 4.2.4. Figure 8.10 compares this total variation recovery to the pseudo-inverse for a synthetic cartoon image. This shows the hability of the total variation to recover sharp features when inpainting Fourier measures. This should be contrasted with the difficulties that faces TV regularization to inpaint over the spacial domain, as shown in Figure 9.10.

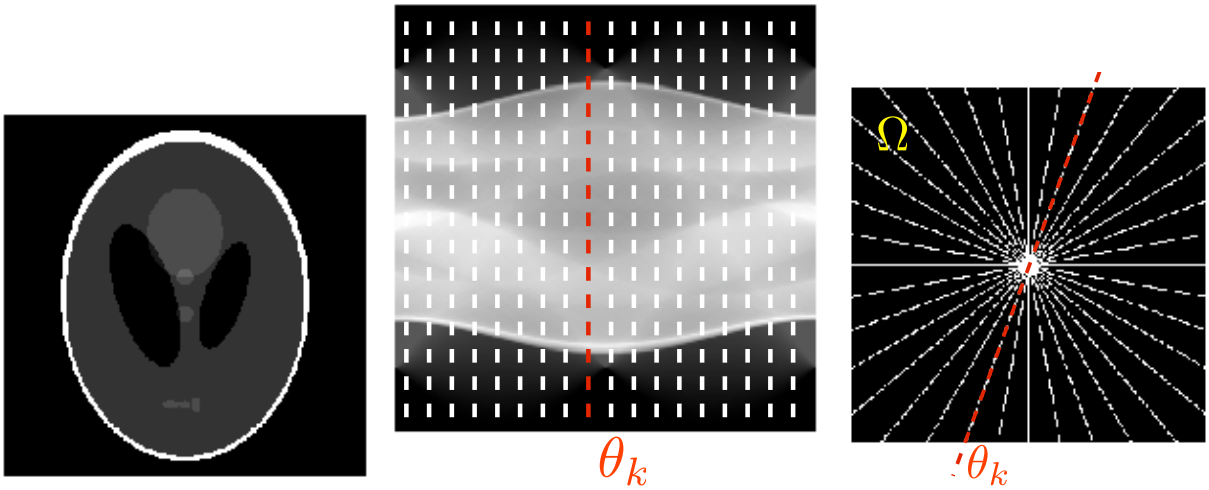


Image f | Radon sub-sampling | Fourier domain

Figure 8.8: Partial Fourier measures.

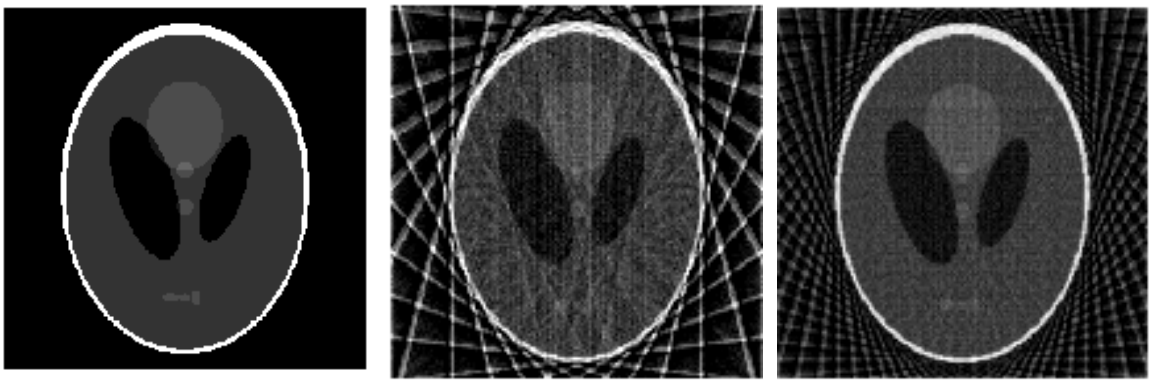


Image f_0 | 13 projections | 32 projections.

Figure 8.9: Pseudo inverse reconstruction from partial Radon projections.

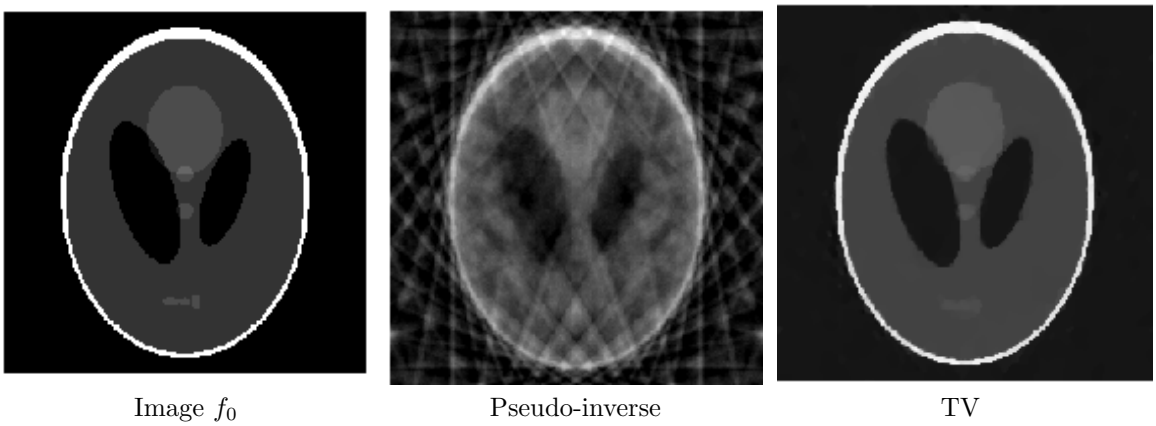


Image f_0 | Pseudo-inverse | TV

Figure 8.10: Total variation tomography inversion.

Chapter 9

Sparse Regularization

Ref [17, 25, 23]

9.1 Sparsity Priors

9.1.1 Ideal sparsity prior.

As detailed in Chapter **??**, it is possible to use an orthogonal basis $\mathcal{B} = \{\psi_m\}_m$ to efficiently approximate an image f in a given class $f \in \Theta$ with a few atoms from $\mathcal{B}$.

To measure the complexity of an approximation with $\mathcal{B}$, we consider the ℓ^0 prior, which counts the number of non-zero coefficients in $\mathcal{B}$

$$J_0(f) \overset{\text{def.}}{=} \#\{m \,;\, \langle f, \psi_m\rangle \neq 0\} \quad \text{where} \quad x_m = \langle f, \psi_m\rangle.$$

One often also denote it as the ℓ^0 "pseudo-norm"

$$\|x\|_0 \overset{\text{def.}}{=} J_0(f).$$

which we treat here as an ideal sparsity measure for the coefficients x of f in $\mathcal{B}$.

Natural images are not exactly composed of a few atoms, but they can be well approximated by a function f_M with a small ideal sparsity $M = J_0(f)$. In particular, the best M-term approximation defined in (4.3) is defined by

$$f_M = \sum_{|\langle f, \psi_m\rangle| > T} \langle f, \psi_m\rangle \psi_m \quad \text{where} \quad M = \#\{m \,;\, |\langle f, \psi_m\rangle| > T\}\,.$$

As detailed in Section 4.2, discontinuous images with bounded variation have a fast decay of the approximation error $\|f - f_M\|$. Natural images f are well approximated by images with a small value of the ideal sparsity prior J_0.

Figure 9.1 shows an examples of decomposition of a natural image in a wavelet basis, $\psi_m = \psi^\omega_{j,n}$ $m = (j, n, \omega)$. This shows that most $\langle f, \psi_m\rangle$ are small, and hence the decomposition is quite sparse.

9.1.2 Convex relaxation

Unfortunately, the ideal sparsity prior J_0 is difficult to handle numerically because $J_0(f)$ is not a convex function of f. For instance, if f and g have non-intersecting supports of there coefficients in $\mathcal{B}$, then $J_0((f+g)/2) = J_0(f) + J_0(g)$, which shows the highly non-convex behavior of J_0.

This ideal sparsity J_0 is thus not amenable to minimization, which is an issue to solve general inverse problems considered in Section **??**.

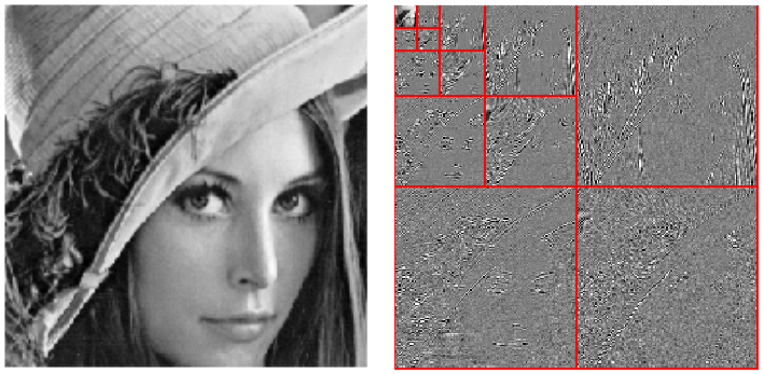


Image f Coefficients $\langle f, \psi_m \rangle$

Figure 9.1: Wavelet coefficients of natural images are relatively sparse.

We consider a family of ℓ^q priors for $q > 0$, intended to approximate the ideal prior J_0

$$J_q(f) = \sum_m |\langle f, \psi_m \rangle|^q.$$

As shown in Figure 14.1, the unit balls in $\mathbb{R}^2$ associated to these priors are shrinking toward the axes, which corresponds to the unit ball for the ℓ^0 pseudo norm. In some sense, the J_q priors are becoming closer to J_0 as q tends to zero, and thus J_q favors sparsity for small q.

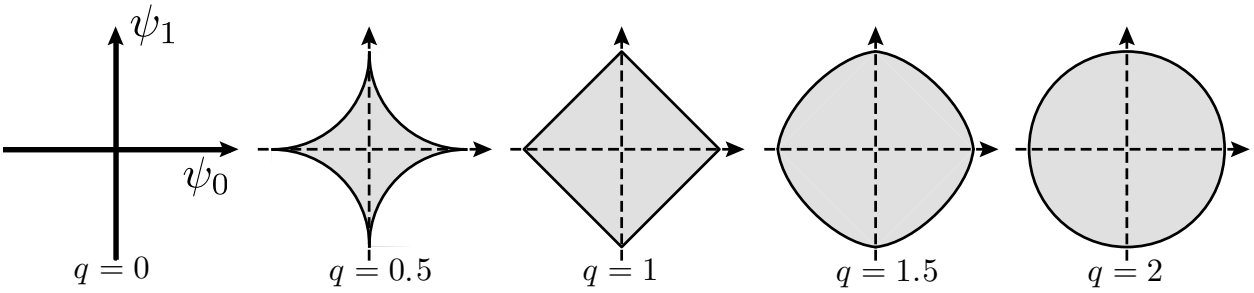


Figure 9.2: ℓ^q balls $\{x\ ;\ J_q(x) \leqslant 1\}$ for varying q.

The prior J_q is convex if and only if $q \geqslant 1$. To reach the highest degree of sparsity while using a convex prior, we consider the ℓ^1 sparsity prior J_1, which is thus defined as

$$J_1(f) = \|(\langle f, \psi_m \rangle)\|_1 = \sum_m |\langle f, \psi_m \rangle|. \tag{9.1}$$

In the following, we consider discrete orthogonal bases $\mathcal{B} = \{\psi_m\}_{m=0}^{N-1}$ of $\mathbb{R}^N$.

9.1.3 Sparse Regularization and Thresholding

Given some orthogonal basis $\{\psi_m\}_m$ of $\mathbb{R}^N$, the denoising by regularization (7.15) is written using the sparsity J_0 and J_1 as

$$f^\star = \operatorname*{argmin}_{g \in \mathbb{R}^N} \frac{1}{2}\|f - g\|^2 + \lambda J_q(f)$$

for $q = 0$ or $q = 1$. It can be re-written in the orthogonal basis as

$$f^\star = \sum_m x_m^\star \psi_m$$

$$\text{where} \quad x^\star = \operatorname*{argmin}_{y \in \mathbb{R}^N} \sum_m \frac{1}{2}|x_m - y_m|^2 + \lambda |y_m|^q$$

where $x_m \overset{\text{def.}}{=} \langle f, \psi_m \rangle$, $y_m \overset{\text{def.}}{=} \langle g, \psi_m \rangle$, and where we use the following slight abuse of notation for $q = 0$

$$\forall\, u \in \mathbb{R}, \quad |u|^0 = \begin{cases} 0 & \text{if} \quad u = 0, \\ 1 & \text{otherwise.} \end{cases}$$

Each coefficients of the denoised image is the solution of a 1-D optimization problem

$$x_m^\star = \operatorname*{argmin}_{u \in \mathbb{R}} \frac{1}{2}|x_m - u|^2 + \lambda |u|^q \tag{9.2}$$

and the following proposition this optimization is solved exactly in closed form using thresholding.

Proposition 26. *One has*

$$x_m^\star = S_T^q(x_m) \quad \textit{where} \quad \begin{cases} T = \sqrt{2\lambda} & \textit{for} \quad q = 0, \\ T = \lambda & \textit{for} \quad q = 1, \end{cases} \tag{9.3}$$

where

$$\forall\, u \in \mathbb{R}, \quad S_T^0(u) \overset{\text{def.}}{=} \begin{cases} 0 & \textit{if} \quad |u| < T, \\ u & \textit{otherwise} \end{cases} \tag{9.4}$$

is the hard thresholding introduced in (6.6), and

$$\forall\, u \in \mathbb{R}, \quad S_T^1(u) \overset{\text{def.}}{=} \operatorname{sign}(u)(|u| - T)_+ \tag{9.5}$$

is the soft thresholding introduced in (6.7).

Proof. One needs to solve (9.2). Figure 14.2, left shows the function $\|x - y\|^2 + T^2\|x\|_0$, and the minimum is clearly at $x = 0$ when $T \geqslant y$, and at $x = y$ otherwise. This is thus a hard thresholding with threshold $T^2 = 2\lambda$. Figure (14.2), right, shows the evolution with λ of the function $\frac{1}{2}\|x - y\|^2 + \lambda|x|$. For $x > 0$, one has $F'(x) = x - y + \lambda$ wich is 0 at $x = y - \lambda$. The minimum is at $x = y - \lambda$ for $\lambda \leqslant y$, and stays at 0 for all $\lambda > y$. □

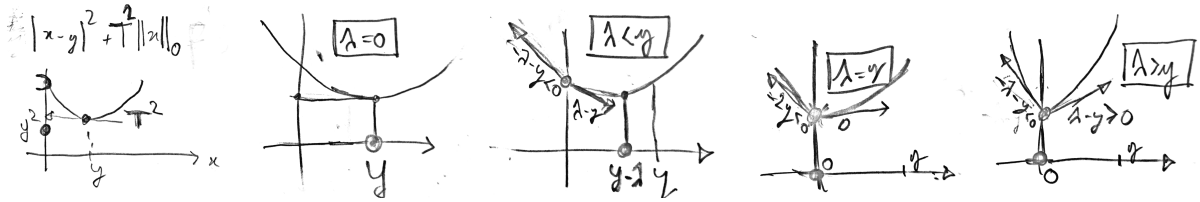


Figure 9.3: Leftmost: function $\| \cdot - y\|^2 + T^2\| \cdot \|_0$. Others: evolution with λ of the function $F(x) \overset{\text{def.}}{=} \frac{1}{2}\| \cdot - y\|^2 + \lambda| \cdot |$.

One thus has

$$f_{\lambda,q} = \sum_m S_T^q(\langle f, \psi_m \rangle)\psi_m.$$

As detailed in Section 6.3, these denoising methods has the advantage that the threshold is simple to set for Gaussian white noise w of variance σ^2. Theoretical values indicated that $T = \sqrt{2\log(N)}\sigma$ is asymptotically optimal, see Section 6.3.3. In practice, one should choose $T \approx 3\sigma$ for hard thresholding (ℓ^0 regularization), and $T \approx 3\sigma/2$ for soft thresholding (ℓ^1 regularization), see Figure 6.14.

9.2 Sparse Regularization of Inverse Problems

Sparse ℓ^1 regularization in an orthogonal basis $\{\psi_m\}_m$ of $\mathbb{R}^N$ makes use of the J_1 prior defined in (9.1), so that the inversion is obtained by solving the following convex program

$$f_\lambda \in \operatorname*{argmin}_{f\in\mathbb{R}^N} \frac{1}{2}\|y-\Phi f\|^2 + \lambda \sum_m |\langle f, \psi_m\rangle|. \tag{9.6}$$

This corresponds to the basis pursuit denoising for sparse approximation introduced by Chen, Donoho and Saunders in [9]. The resolution of (9.6) can be perform using an iterative thresholding algorithm as detailed in Section 9.3.

Analysis vs. synthesis priors. When the set of atoms $\Psi = \{\psi_m\}_{m=1}^Q$ is non-orthognal (and might even be redundant in the case $Q > N$), there is two distincts way to generalizes problem (9.6), which we formulate as in (**??**), by introducing a generic convex prior J

$$f_\lambda \in \operatorname*{argmin}_{f\in\mathbb{R}^N} \frac{1}{2}\|y-\Phi f\|^2 + \lambda J(f). \tag{9.7}$$

In the following, with a slight abuse of notation, we denote the "analysis" and "synthesis" operator as

$$\Psi : x \in \mathbb{R}^Q \mapsto \Psi x = \sum_m x_m \psi_m \quad \text{and} \quad \Psi^* : f \in \mathbb{R}^N \mapsto (\langle f, \psi_m\rangle)_{m=1}^Q \in \mathbb{R}^Q.$$

The so-called analysis-type prior is simply measuring the sparsity of the correlations with the atoms in the dictionary

$$J_1^{\mathrm{A}}(f) \overset{\text{def.}}{=} \sum_m |\langle f, \psi_m\rangle| = \|\Psi^* f\|_1. \tag{9.8}$$

The so-called synthesis-type priori in contrast measure the sparsity of the sparsest expansion of f in Ψ, i.e.

$$J_1^{\mathrm{S}}(f) \overset{\text{def.}}{=} \min_{x\in\mathbb{R}^q, \Psi x = f} \|x\|_1. \tag{9.9}$$

While the analysis regularization (9.8) seems simpler to handle, it is actually the contrary. Solving (9.7) with $J = J_1^{\mathrm{A}}$ is in fact quite involved, and necessitate typically primal-dual algorithm as detailed in Chapter 17. Furthermore, the theoretical study of the performance of the resulting regularization method is mostly an open problem.

We thus now focus on the synthesis regularization problem $J = J_1^{\mathrm{S}}$, and we re-write (9.7) conveniently as $f_\lambda = \Psi x_\lambda$ where x_λ is any solution of the following Basis Pursuit Denoising problem

$$x_\lambda \in \operatorname*{argmin}_{x\in\mathbb{R}^Q} \frac{1}{2\lambda}\|y - Ax\|^2 + \|x\|_1 \tag{9.10}$$

where we introduced the following matrix

$$A \overset{\text{def.}}{=} \Phi\Psi \in \mathbb{R}^{P\times Q}.$$

As $\lambda \to 0$, we consider the following limit constrained problem

$$x^\star = \operatorname*{argmin}_{Ax=y} \|x\|_1 \tag{9.11}$$

and the signal is recovered as $f^\star = \Psi x^\star \in \mathbb{R}^N$.

9.3 Iterative Soft Thresholding Algorithm

This section details an iterative algorithm that compute a solution of (9.10).

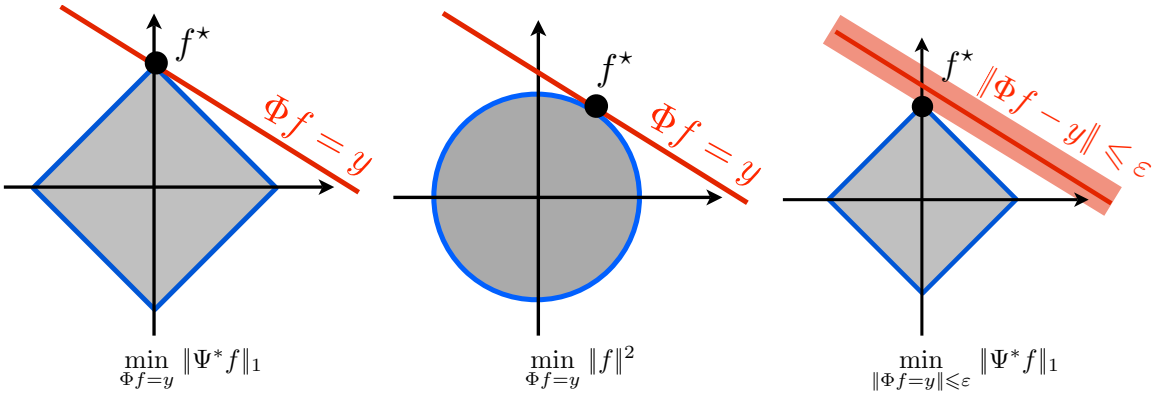


Figure 9.4: Geometry of convex optimizations.

9.3.1 Noiseless Recovery as a Linear Program

Before detailing this methods, which only deal with the case $\lambda > 0$, let us note that in the noiseless settig, $\lambda = 0$ and (9.11) is actually equivalent to a linear program. Indeed, decomposing $a = x_+ - x_-$ with $(x_+, x_-) \in (\mathbb{R}_+^Q)^2$, one has

$$x^\star = \operatorname*{argmin}_{(x_+,x_-)\in(\mathbb{R}_+^Q)^2} \{\langle x_+, \mathbb{1}_Q\rangle + \langle x_-, \mathbb{1}_Q\rangle \;;\; y = A(x_+ - x_-)\}. \tag{9.12}$$

which is a linear program. For small to medium scale problem (Q of the order of a few thousands) it can be solved using the simplex algorithm or interior point methods. For large scale problems such as those encountered in imaging or machine learning, this is not possible, and one has to ressort to simpler first order schemes. A possible option is the Douglas-Rachford splitting scheme, which is detailed in Section **??**. Let us however stress that the constrained problem (9.11), because of its polyhedral (linear) nature, is in fact harder to solve than the penalized problem (9.10) that we now target.

9.3.2 Projected Gradient Descent for ℓ^1.

As a first practical example to solve (9.10), we will show how to use the projected gradient descent method, which is analyzed in detailed in Section 17.1.3. Similarly to (9.12), we remap (9.10) as the resolution of a constraint minimization problem of the form (17.4) where here $\mathcal{C}$ is a positivity constraint and

$$u = (u_+, u_-) \in (\mathbb{R}^Q)^2, \quad \mathcal{C} = (\mathbb{R}_+^Q)^2, \quad \text{and} \quad \mathcal{E}(u) = \frac{1}{2}\|\Phi(u_+ - u_-) - y\|^2 + \lambda\langle u_+, \mathbb{1}_Q\rangle + \lambda\langle u_-, \mathbb{1}_Q\rangle.$$

The projection on $\mathcal{C}$ is here simple to compute

$$\mathrm{Proj}_{(\mathbb{R}_+^Q)^2}(u_+, u_-) = ((u_+)_\oplus, (u_-)_\oplus) \quad \text{where} \quad (r)_\oplus \overset{\text{def.}}{=} \max(r, 0),$$

and the gradient reads

$$\nabla\mathcal{E}(u_+, u_-) = (\eta + \lambda\mathbb{1}_Q, -\eta + \lambda\mathbb{1}_Q) \quad \text{where} \quad \eta = \Phi^*(\Phi(u_+ - u_-) - y)$$

Denoting $u^{(\ell)} = (u_+^{(\ell)}, u_-^{(\ell)})$ and $x^{(\ell)} \overset{\text{def.}}{=} u_+^{(\ell)} - u_-^{(\ell)}$, the iterate of the projected gradient descent algorithm (17.5) read

$$u_+^{(\ell+1)} \overset{\text{def.}}{=} \left(u_+^{(\ell)} - \tau_\ell(\eta^{(\ell)} + \lambda)\right)_\oplus \quad \text{and} \quad u_-^{(\ell+1)} \overset{\text{def.}}{=} \left(u_-^{(\ell)} - \tau_\ell(-\eta^{(\ell)} + \lambda)\right)_\oplus$$

where $\eta^{(\ell)} \overset{\text{def.}}{=} \Phi^*(\Phi x^{(\ell)} - y)$.

Theorem 30 ensures that $u^{(\ell)} \to u^\star$ a solution to (17.4) if

$$\forall\, \ell, \quad 0 < \tau_{\min} < \tau_\ell < \tau_{\max} < \frac{2}{\|\Phi\|^2},$$

and thus $x^{(\ell)} \to x^\star = u_+^\star - u_-^\star$ which is thus a solution to (9.10).

9.3.3 Iterative Soft Thresholding and Forward Backward

A drawback of this projected gradient descent scheme is that it necessitate to store $2Q$ coefficients. A closely related method, which comes with exactly the same convergence guarantees and rate, is the so called "iterative soft thresholding algorithm" (ISTA). This algorithm was derived by several authors, among which [15, 12], and belongs to the general family of forward-backward splitting in proximal iterations [11], which we detail in Section 17.3.2.

For the sake of simplicity, let us derive this algorithm in the special case of ℓ^1 by surrogate function minimization. We aim at minimizing (9.6)

$$\mathcal{E}(x) \overset{\text{def.}}{=} \frac{1}{2}\|y - Ax\|^2 + \lambda\|x\|_1$$

and we introduce for any fixed x' the function

$$\mathcal{E}_\tau(x, x') \overset{\text{def.}}{=} \mathcal{E}(x) - \frac{1}{2}\|Ax - Ax'\|^2 + \frac{1}{2\tau}\|x - x'\|^2.$$

We notice that $\mathcal{E}(x, x) = 0$ and one has

$$K(x, x') \overset{\text{def.}}{=} -\frac{1}{2}\|Ax - Ax'\|^2 + \frac{1}{2\tau}\|x - x'\| = \frac{1}{2}\langle\left(\frac{1}{\tau}\mathrm{Id}_N - A^*A\right)(x - x'),\, x - x'\rangle.$$

This quantity $K(x, x')$ is positive if $\lambda_{\max}(A^*A) \leqslant 1/\tau$ (maximum eigenvalue), i.e. $\tau \leqslant 1/\|A\|_{\text{op}}^2$, where we recall that $\|A\|_{\text{op}} = \sigma_{\max}(A)$ is the operator (algebra) norm. This shows that $\mathcal{E}_\tau(x, x')$ is a valid surrogate functional, in the sense that

$$\mathcal{E}(x) \leqslant \mathcal{E}_\tau(x, x'), \quad \mathcal{E}_\tau(x, x') = 0, \quad \text{and} \quad \mathcal{E}(\cdot) - \mathcal{E}_\tau(\cdot, x') \text{ is smooth.}$$

This leads to define

$$x^{(\ell+1)} \overset{\text{def.}}{=} \operatorname*{argmin}_x \mathcal{E}_{\tau_\ell}(x, x^{(\ell)}) \tag{9.13}$$

which by construction satisfies

$$\mathcal{E}(x^{(\ell+1)}) \leqslant \mathcal{E}(x^{(\ell)}).$$

Proposition 27. *The iterates $x^{(\ell)}$ defined by (14.6) satisfy*

$$x^{(\ell+1)} = \mathcal{S}^1_{\lambda\tau_\ell}\left(x^{(\ell)} - \tau_\ell A^*(Ax^{(\ell)} - y)\right) \tag{9.14}$$

where $\mathcal{S}^1_\lambda(x) = (S^1_\lambda(x_m))_m$ where $S^1_\lambda(r) = \mathrm{sign}(r)(|r| - \lambda)_\oplus$ is the soft thresholding operator defined in (9.5).

Proof. One has

$$\begin{aligned}
\mathcal{E}_\tau(x, x') &= \frac{1}{2}\|Ax - y\|^2 - \frac{1}{2}\|Ax - Ax'\|^2 + \frac{1}{2\tau}\|x - x'\|^2 + \lambda\|x\|_1 \\
&= C + \frac{1}{2}\|Ax\|^2 - \frac{1}{2}\|Ax\|^2 + \frac{1}{2\tau}\|x\|^2 - \langle Ax,\, y\rangle + \langle Ax,\, Ax'\rangle - \frac{1}{\tau}\langle x,\, x'\rangle + \lambda\|x\|_1 \\
&= C + \frac{1}{2\tau}\|x\|^2 + \langle x,\, -A^*y + AA^*x' - \frac{1}{\tau}x'\rangle + \lambda\|x\|_1 \\
&= C' + \frac{1}{\tau}\left(\|x - (x' - \tau A^*(Ax' - y))\|^2 + \tau\lambda\|x\|_1.\right)
\end{aligned}$$

Proposition (26) shows that the minimizer of $\mathcal{E}_\tau(x, x')$ is thus indeed $\mathcal{S}^1_{\lambda\tau}(x' - \tau_\ell A^*(Ax' - y))$. □

Of course, these iterations (14.7) are the same as the FB iterates (17.15), when, for the special case (9.6), one can consider a splitting of the form (17.15) defining

$$\mathcal{F} = \frac{1}{2}\|A \cdot -y\|^2 \quad \text{and} \quad \mathcal{G} = \lambda\|\cdot\|_1. \tag{9.15}$$

In the case (9.15), Proposition (26) shows that $\text{Prox}_{\rho J}$ is the soft thresholding.

9.4 Example: Sparse Deconvolution

9.4.1 Sparse Spikes Deconvolution

Sparse spikes deconvolution makes use of sparsity in the spacial domain, which corresponds to the orthogonal basis of Diracs $\psi_m[n] = \delta[n-m]$. This sparsity was first introduced in the seismic imaging community [], where the signal f_0 represent the change of density in the underground and is assumed to be composed of a few Diracs impulse.

In a simplified linearized 1D set-up, ignoring multiple reflexions, the acquisition of underground data f_0 is modeled as a convolution $y = h \star f_0 + w$, where h is a so-called "wavelet" signal sent in the ground. This should not be confounded with the construction of orthogonal wavelet bases detailed in Chapter **??**, although the term "wavelet" originally comes from seismic imaging.

The wavelet filter h is typically a band pass signal that perform a tradeoff between space and frequency concentration especially tailored for seismic exploration. Figure (9.5) shows a typical wavelet that is a second derivative of a Gaussian, together with its Fourier transform. This shows the large amount of information removed from f during the imaging process.

The sparse ℓ^1 regularization in the Dirac basis reads

$$f^\star = \operatorname*{argmin}_{f \in \mathbb{R}^N} \frac{1}{2}\|f \star h - y\|^2 + \lambda \sum_m |f_m|.$$

Figure 9.5 shows the result of ℓ^1 minimization for a well chosen λ parameter, that was optimized in an oracle manner to minimize the error $\|f^\star - f_0\|$.

The iterative soft thresholding for sparse spikes inversion iterates

$$\tilde{f}^{(k)} = f^{(k)} - \tau h \star (h \star f^{(k)} - y)$$

and

$$f_m^{(k+1)} = S^1_{\lambda\tau}(\tilde{f}_m^{(k)})$$

where the step size should obeys

$$\tau < 2/\|\Phi^*\Phi\| = 2/\max_\omega |\hat{h}(\omega)|^2$$

to guarantee convergence. Figure 9.6 shows the progressive convergence of the algorithm, both in term of energy minimization and iterates. Since the energy is not strictly convex, we note that convergence in energy is not enough to guarantee convergence of the algorithm.

9.4.2 Sparse Wavelets Deconvolution

Signal and image acquired by camera always contain some amount of blur because of objects being out of focus, movements in the scene during exposure, and diffraction. A simplifying assumption assumes a spatially invariant blur, so that Φ is a convolution

$$y = f_0 \star h + w.$$

In the following, we consider h to be a Gaussian filter of width $\mu > 0$. The number of effective measurements can thus be considered to be $P \sim 1/\mu$, since Φ nearly set to 0 large enough Fourier frequencies. Table **??** details the implementation of the sparse deconvolution algorithm.

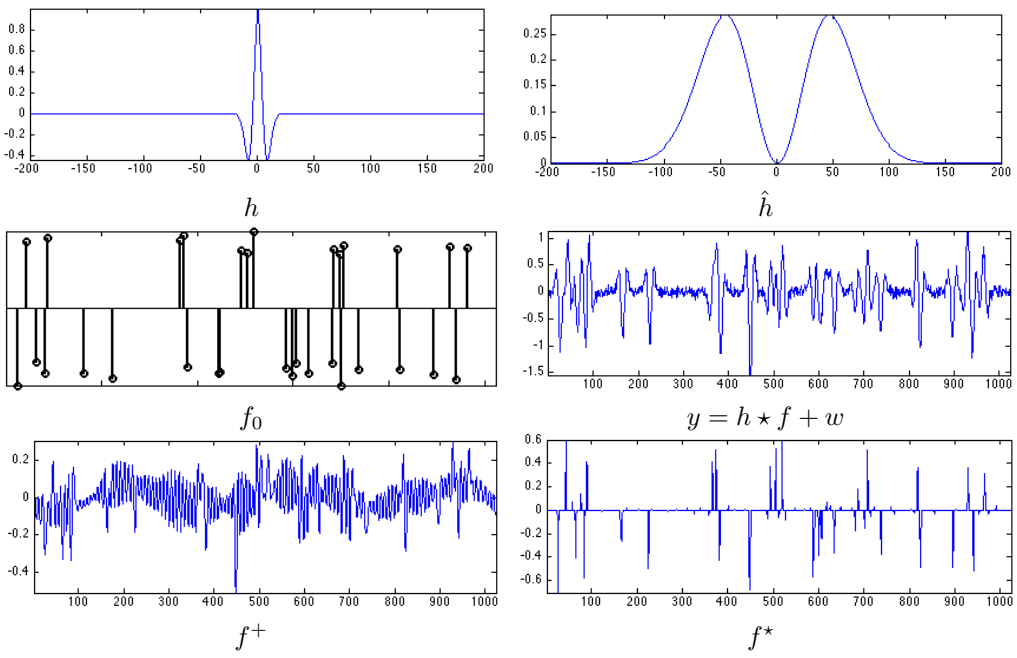


Figure 9.5: Pseudo-inverse and ℓ^1 sparse spikes deconvolution.

Figures 9.7 and 9.8 shows examples of signal and image acquisition with Gaussian blur.

Sobolev regularization (7.17) improves over ℓ^2 regularization (**??**) because it introduces an uniform smoothing that reduces noise artifact. It however fail to recover sharp edge and thus does a poor job in inverting the operator. To recover sharper transition and edges, one can use either a TV regularization or a sparsity in an orthogonal wavelet basis.

Figure 9.7 shows the improvement obtained in 1D with wavelets with respect to Sobolev. Figure 9.8 shows that this improvement is also visible for image deblurring. To obtain a better result with fewer artifact, one can replace the soft thresholding in orthogonal wavelets in during the iteration (**??**) by a thresholding in a translation invariant tight frame as defined in (6.10).

Figure 9.9 shows the decay of the SNR as a function of the regularization parameter λ. This SNR is computed in an oracle manner since it requires the knowledge of f_0. The optimal value of λ was used in the reported experiments.

9.4.3 Sparse Inpainting

This section is a follow-up of Section 8.5.2.

To inpaint using a sparsity prior without noise, we use a small value for λ. The iterative thresholding algorithm (**??**) is written as follow for $\tau = 1$,

$$f^{(k+1)} = \sum_m S^1_\lambda(\langle P_y(f^{(k)}),\, \psi_m \rangle)\psi_m$$

Figure 9.10 shows the improvevement obtained by the sparse prior over the Sobolev prior if one uses soft thresholding in a translation invariant wavelet frame.

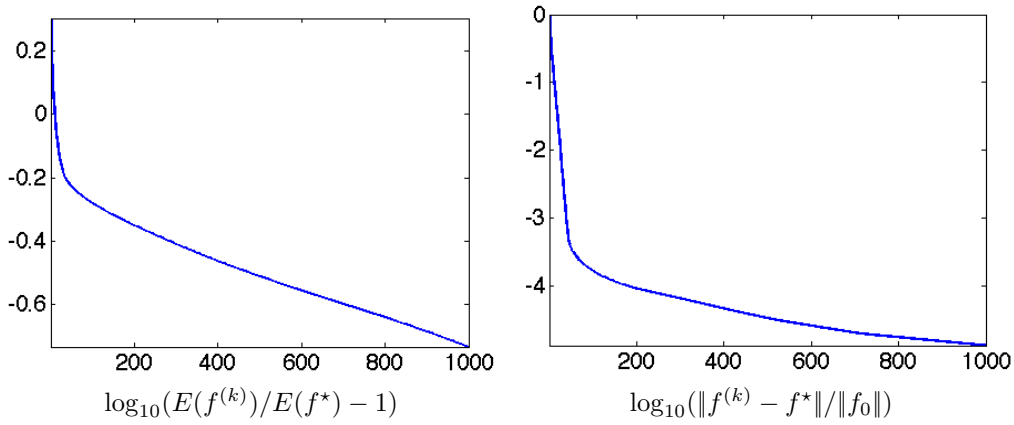


Figure 9.6: Decay of the energy and convergence through the iterative thresholding iterations.

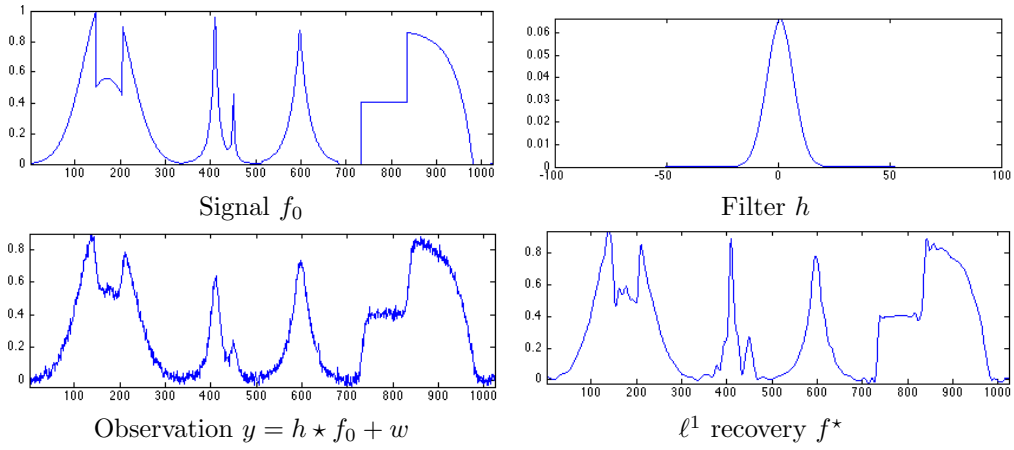


Figure 9.7: Sparse 1D deconvolution using orthogonal wavelets.

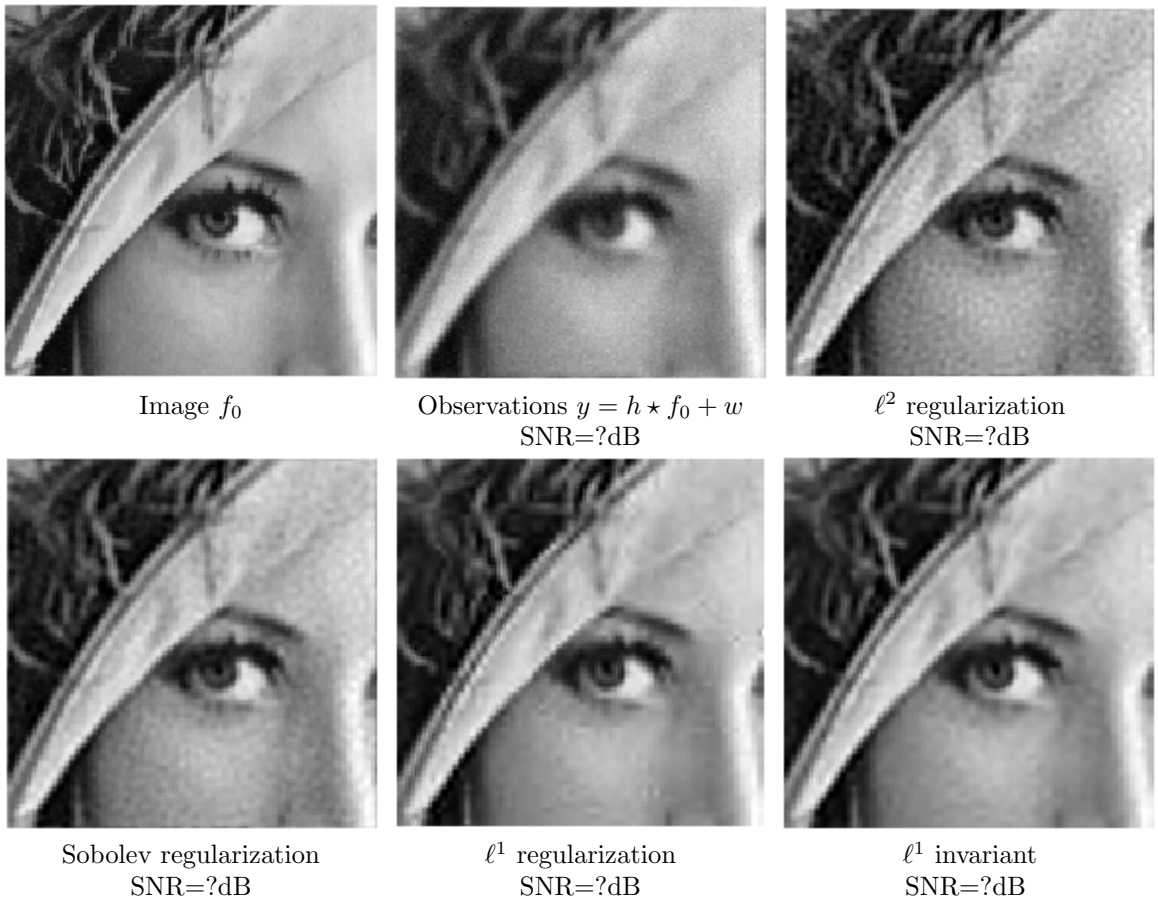


Figure 9.8: Image deconvolution.

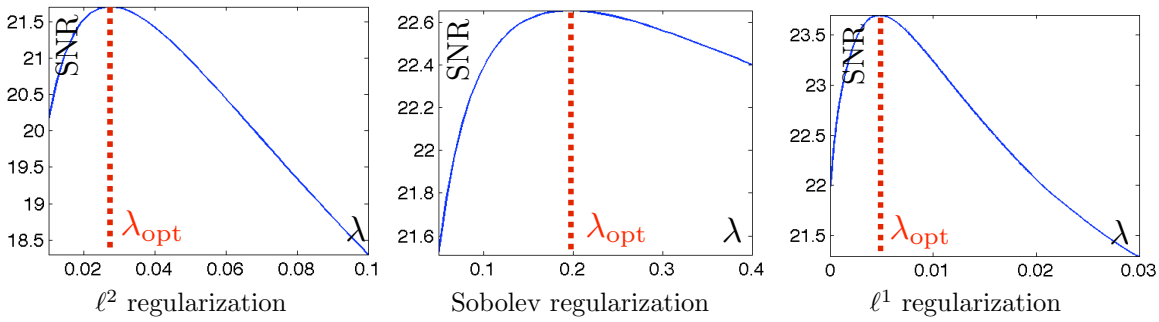


Figure 9.9: SNR as a function of λ.

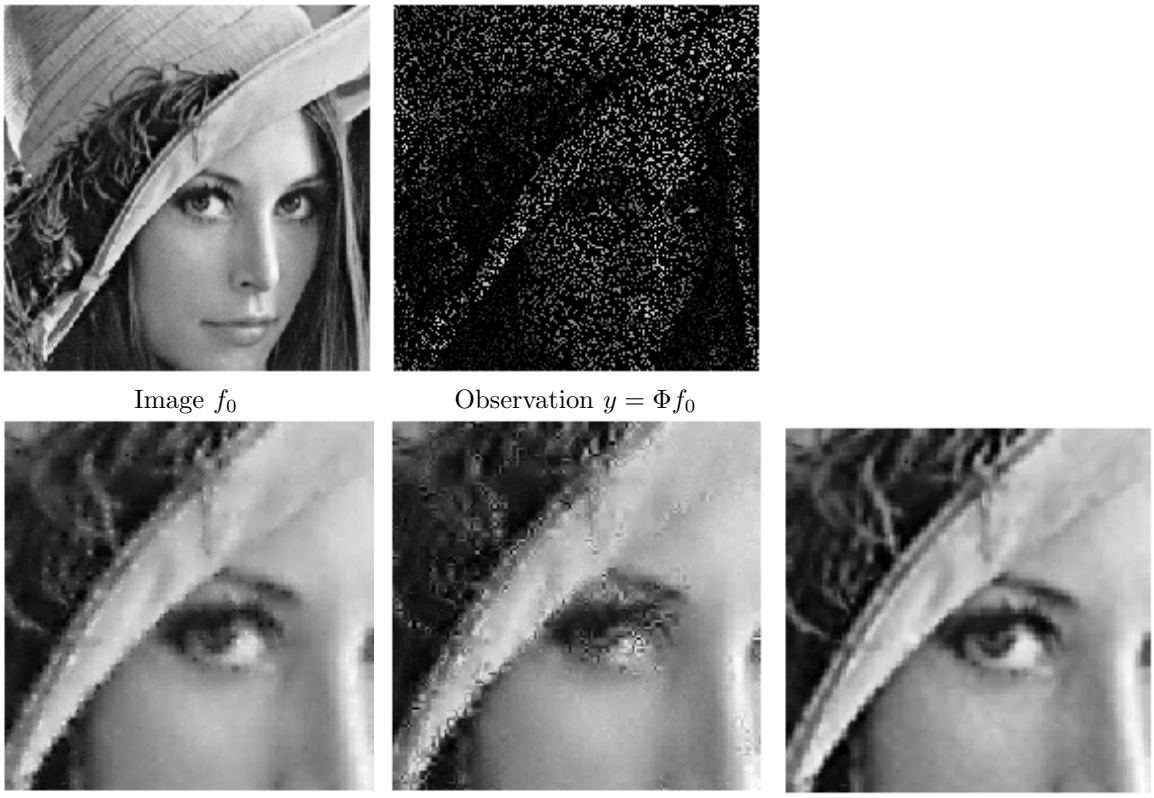


Image f_0 | Observation $y = \Phi f_0$

Sobolev $f^\star$ SNR=?dB | Ortho. wav $f^\star$ SNR=?dB | TI. wav $f^\star$ SNR=?dB

Figure 9.10: Inpainting with Sobolev and sparsity.

Chapter 10

Theory of Sparse Regularization

We now apply the basics elements of convex analysis from the previous chapter to perform a theoretical analysis of the properties of the Lasso, in particular its performances to recover sparse vectors.

10.1 Existence and Uniqueness

10.1.1 Existence

We consider problems (9.10) and (9.11), that we rewrite here as

$$\min_{x\in\mathbb{R}^N} f_\lambda(x) \overset{\text{def.}}{=} \frac{1}{2\lambda}\|y-Ax\|^2+\lambda\|x\|_1 \tag{$\mathcal{P}_\lambda(y)$}$$

and its limit as $\lambda\to 0$

$$\min_{Ax=y}\|x\|_1 = \min_x f_0(x) \overset{\text{def.}}{=} \iota_{\mathcal{L}_y}(x)+\|x\|_1. \tag{$\mathcal{P}_0(y)$}$$

where $A\in\mathbb{R}^{P\times N}$, and $\mathcal{L}_y \overset{\text{def.}}{=} \{x\in\mathbb{R}^N \;;\; Ax=y\}$.

We recall that the setup is that one observe noise measures

$$y = Ax_0+w$$

and we would like conditions to ensure for x_0 to solution to $(\mathcal{P}_0(Ax_0))$ (i.e. when $w=0$) and to be close (in some sense to be defined, and in some proportion to the noise level $\|w\|$) to the solutions of $(\mathcal{P}_0(y=Ax_0+w))$ when λ is wisely chosen as a function of $\|w\|$.

First let us note that since $(\mathcal{P}_\lambda(y))$ is unconstrained and coercive (because $\|\cdot\|_1$ is), this problem always has solutions. Since A might have a kernel and $\|\cdot\|_1$ is not strongly convex, it might have non-unique solutions. If $y\in\mathrm{Im}(A)$, the constraint set of $(\mathcal{P}_0(y))$ is non-empty, and it also has solutions, which might fail to be unique.

Figure 10.1 gives the intuition of the theory that will be developed in this chapter, regarding the exact or approximated recovery of sparse vectors x_0, and the need for a careful selection of the λ parameter.

10.1.2 Polytope Projection for the Constraint Problem

The following proposition gives a geometric description of those vectors which are recovered by ℓ^1 minimization when there is no noise.

Proposition 28. *We denote* $B \overset{\text{def.}}{=} \{x\in\mathbb{R}^N \;;\; \|x\|_1\leqslant 1\}$. *Then, assuming* $\mathrm{Im}(A)=\mathbb{R}^P$,

$$x_0 \text{ is a solution to } \mathcal{P}_0(Ax_0) \quad\Longleftrightarrow\quad A\frac{x_0}{\|x_0\|_1}\in\partial(AB) \tag{10.1}$$

where "∂" denoted the boundary and $AB=\{Ax \;;\; x\in B\}$.

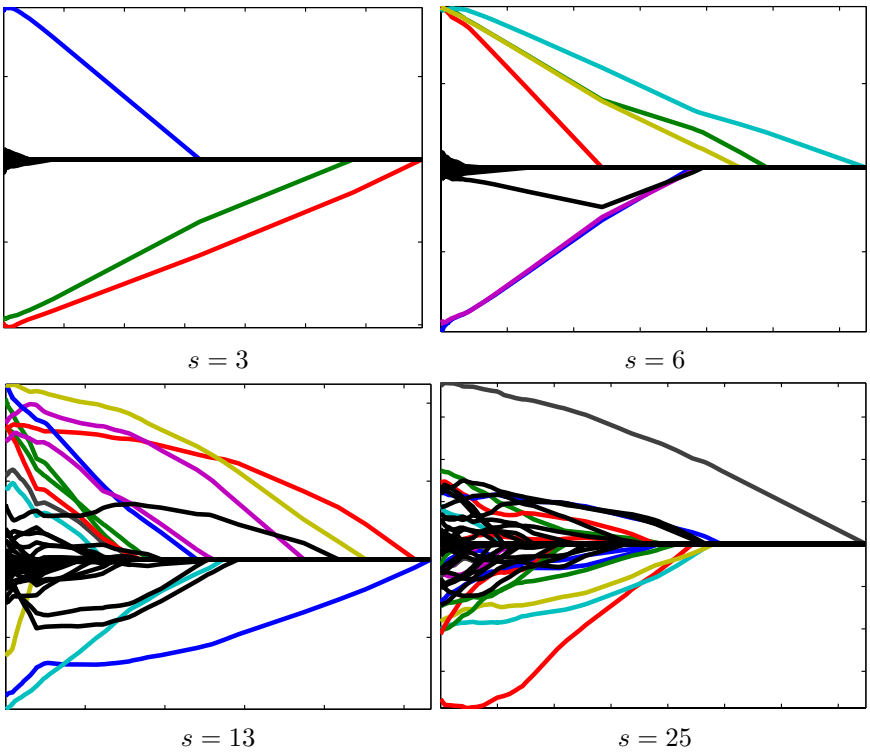


Figure 10.1: Display of the evolution $\lambda \mapsto x_\lambda$ of the solutions of $(\mathcal{P}_\lambda(y))$.

Proof. We first prove " $\Rightarrow$". We suppose that x_0 is not a solution, and aim at showing that $A\frac{x_0}{\|x_0\|_1} \in \mathrm{int}(AB_\rho)$. Since it is not a solution, there exists z such that $Ax_0 = Az$ and $\|z\|_1 = (1-\delta)\|x_0\|_1$ with $\delta > 0$. Then for any displacement $h = A\varepsilon \in \mathrm{Im}(A)$, where one can impose $\varepsilon \in \ker(A)^\perp$, i.e. $\varepsilon = A^+h$, one has $Ax_0 + h = A(z+\varepsilon)$ and

$$\|z+\varepsilon\|_1 \leqslant \|z\|_1 + \|\Phi^+h\| \leqslant (1-\delta)\|x_0\|_1 + \|\Phi^+\|_{1,1}\|h\|_1 < \frac{\delta}{\|A^+\|_{1,1}}\|x_0\|.$$

This means that choosing $\|h\|_1 < \frac{\delta}{\|A^+\|_{1,1}}\|x_0\|_1$ implies that $A\frac{x_0}{\|x_0\|_1} \in \mathrm{int}(AB)$.

We now prove "$\Leftarrow$". We suppose that $A\frac{x_0}{\|x_0\|_1} \in \mathrm{int}(AB)$. Then there exists z such that $Ax_0 = (1-\delta)Az$ and $\|z\|_1 < \|x_0\|_1$. This implies $\|(1-\delta)z\|_1 < \|x_0\|_1$ so that $(1-\delta)z$ is better than x_0 which is thus not a solution. □

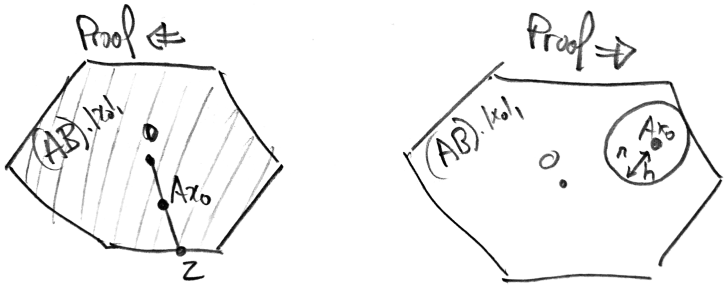


Figure 10.2: Graphical display of the proof for the polytope analysis of ℓ^1 exact recovery.

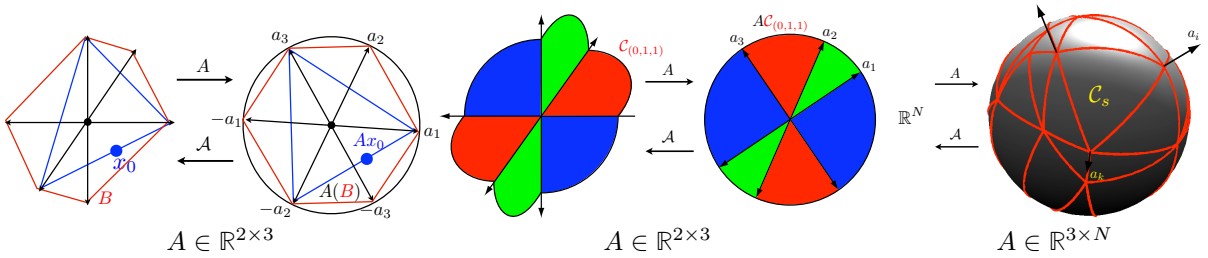


Figure 10.3: Display of the action of the linear map A on the ℓ^1 ball B, and of the inverse non-linear map $\mathcal{A}$ defined by the solution of $(\mathcal{P}_0(y))$.

This results state that "friendly" identifiable vectors (those recovered by ℓ^1) are those who gets projected by A on the boundary of the polytope $\|x_0\|_1 AB$. Intuitively, if P is small in comparison to N, then this projected polytope is small, and most vector will failed to be reconstructed by solving ℓ^1 minimization. This also suggests why using random projections as in Chapter 11, because somehow they results in a low distortion embedding of the ℓ^1 ball from $\mathbb{R}^N$ to $\mathbb{R}^P$.

Note that if x_0 is identifiable, so is λx_0 for ρx_0 for $\rho > 0$, and in fact, since the recovery condition only depends on the geometry of the faces of B, the obtained condition (10.1) only depends on $\mathrm{sign}(x_0)$. We denote $\mathcal{A} : y \mapsto x^\star$ the map from y to a solution of $(\mathcal{P}_0(y))$, which we assume is unique for simplicity of exposition. Condition (10.1) thus shows that A and $\mathcal{A}$ are inverse bijection on a family of cones $\mathcal{C}_s = \{x \; ; \; \mathrm{sign}(x) = s\}$ and $A\mathcal{C}_s$ for certain "friendly" sign patterns s. These cones $A\mathcal{C}_s$ form a partition of the image space $\mathbb{R}^P$. Assuming for simplicity that the columns $(a_j)_j$ of A have unit norm, for $P = 3$, the interaction of these $A\mathcal{C}_s$ with the unit sphere of $\mathbb{R}^3$ for a so-called Delaunay triangulation of the sphere (this construction extends to higher dimension by replacing triangle by simplexes), see also Figure 10.7. Such Delaunay triangulation is characterized by the empty spherical cap property (each circumcircle associated to a triangle should not contains any columns vector a_j of the matrix). Figure 10.3 illustrate these conclusions in $\mathbb{R}^2$ and $\mathbb{R}^3$.

10.1.3 Optimality Conditions

In the following, given an index set $I \subset \{1, \dots, N\}$, denoting $A = (a_i)_{i=1}^N$ the columns of A, we denote $A_I \overset{\text{def.}}{=} (a_i)_{i\in I} \in \mathbb{R}^{P\times |I|}$ the extracted sub-matrix. Similarly, for $x \in \mathbb{R}^N$, we denote $x_I \overset{\text{def.}}{=} (x_i)_{i\in I} \in \mathbb{R}^{|I|}$.

The following proposition rephrases the first order optimality conditions in a handy way.

Proposition 29. *x_λ is a solution to $(\mathcal{P}_\lambda(y))$ for $\lambda > 0$ if and only if*

$$\eta_{\lambda,I} = \mathrm{sign}(x_{\lambda,I}) \quad \text{and} \quad \|\eta_{\lambda,I^c}\| \leqslant \lambda$$

where we define

$$I \overset{\text{def.}}{=} \mathrm{supp}(x_\lambda) \overset{\text{def.}}{=} \{i \; ; \; x_{\lambda,i} \neq 0\}, \quad \text{and} \quad \eta_\lambda \overset{\text{def.}}{=} \frac{1}{\lambda} A^*(y - Ax_\lambda). \tag{10.2}$$

Proof. Since $(\mathcal{P}_\lambda(y))$ involves a sum of a smooth and a continuous function, its sub-differential reads

$$\partial f_\lambda(x) = \frac{1}{\lambda} A^*(Ax - y) + \lambda \partial \|\cdot\|_1(x).$$

Thus x_λ is solution to $(\mathcal{P}_\lambda(y))$ if and only if $0 \in \partial f_\lambda(x_\lambda)$, which gives the desired result. □

Note that one has in particular that $\mathrm{supp}(x_\lambda) \subset \mathrm{sat}(\eta_\lambda)$.

The following proposition studies the limit case $\lambda = 0$ and introduces the crucial concept of "dual certificates", which are the Lagrange multipliers of the constraint $\mathcal{L}_y$.

Proposition 30. *$x^\star$ being a solution to $(\mathcal{P}_0(y))$ is equivalent to having $Ax^\star = y$ and that*

$$\exists \eta \in \mathcal{D}_0(y, x^\star) \overset{\text{def.}}{=} \mathrm{Im}(A^*) \cap \partial \|\cdot\|_1(x^\star). \tag{10.3}$$

Proof. Since ($\mathcal{P}_0(y)$) involves a sum with a continuous function, one can also computes it sub-differential as

$$\partial f_0(x) = \partial \iota_{\mathcal{L}_y}(x) + \partial \|\cdot\|_1(x).$$

If $x \in \mathcal{L}_y$, then $\partial \iota_{\mathcal{L}_y}(x)$ is the linear space orthogonal to $\mathcal{L}_y$, i.e. $\ker(A)^\perp = \mathrm{Im}(A^*)$. □

Note that one has in particular that $\mathrm{supp}(x^\star) \subset \mathrm{sat}(\eta)$ for any valid vector $\eta \in \mathcal{D}_0(y, x^\star)$.
Writing $I = \mathrm{supp}(x^\star)$, one thus has

$$\mathcal{D}_0(y, x^\star) = \{\eta = A^* p \;;\; \eta_I = \mathrm{sign}(x^\star_I),\ \|\eta\|_\infty \leqslant 1\}.$$

Although it looks like the definition of $\mathcal{D}_0(y, x^\star)$ depends on the choice of a solution $x^\star$, convex duality (studied in the next chapter) shows that it is not the case (it is the same set for all solutions).

10.1.4 Uniqueness

The following proposition shows that the Lasso selects a set of linearly independent regressors (columns of A). This is why this method is also often called "basis pursuit".

Proposition 31. *For $\lambda \geqslant 0$, there is always a solution $x^\star$ to ($\mathcal{P}_\lambda(y)$) with $I = \mathrm{supp}(x^\star)$ such that $\ker(A_I) = \{0\}$*

Proof. Let x be a solution and denote $I = \mathrm{supp}(x)$. If $\ker(A_I) \neq \{0\}$, one selects $h_I \in \ker(A_I)$ and define for $t \in \mathbb{R}$ the vector $x_t \overset{\text{def.}}{=} x + th$. We denote t_0 the smallest $|t|$ such that $\mathrm{sign}(x_t) \neq \mathrm{sign}(x)$, i.e. $\mathrm{supp}(x_t)$ is strictly included in I. For $t < t_0$, since $Ax_t = Ax$ and $\mathrm{sign}(x_t) = \mathrm{sign}(x)$, x_t still satisfies the same first order condition as x_0, and one can apply either Proposition 30 (for $\lambda = 0$) or Proposition 29 (for $\lambda > 0$), so that x_t is a solution of ($\mathcal{P}_\lambda(y)$). Since the minimized function are lower semi continuous, $x_t \to x_{t_0}$ is still a solution. If $\ker(A_J) \neq \{0\}$ with $J = \mathrm{supp}(x_{t_0})$, one is over, otherwise one can iterate this argument on x_{t_0} in place of x and have a sequence of supports which is strictly decaying in size, so it must terminate. □

This results in particular that if columns of A_I are not independent, then the solution of ($\mathcal{P}_\lambda(y)$) is necessarily non-unique.

Assuming that x_λ is a solution such that $\ker(A_I) = \{0\}$, then from ($\mathcal{P}_\lambda(y)$), one obtains the following implicit expression for the solution

$$x_{\lambda,I} = A_I^+ y - \lambda (A_I^* A_I)^{-1} \mathrm{sign}(x_{\lambda,I}). \tag{10.4}$$

This expression can be understood as a form of generalized soft thresholding (one retrieve the soft thresholding when $A = \mathrm{Id}_N$).

The following useful lemma shows that while solutions x_λ to ($\mathcal{P}_\lambda(y)$) are not necessarily unique, the associated "predictor" (i.e. denoised version of y) Ax_λ is however always uniquely defined. Note that according to (10.5), one has

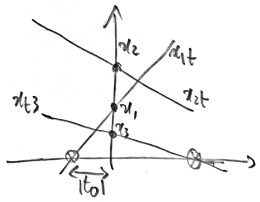


Figure 10.4: Trajectory $(x_t)_t$.

$$\Phi x_\lambda = \mathrm{Proj}_{\mathrm{Im}(A_I)} y - \lambda A_I (A_I^* A_I)^{-1} \mathrm{sign}(x_{\lambda,I}). \tag{10.5}$$

so up to a $O(\lambda)$ bias, this predictor is an orthogonal projection on a low dimensional subspace indexed by I.

Lemma 3. *For $\lambda \geqslant 0$, if (x_1, x_2) are solution to ($\mathcal{P}_\lambda(y)$), then $Ax_1 = Ax_2$.*

Proof. For $\lambda = 0$, this is trivial because $Ax_1 = Ax_2 = y$. Otherwise, let us assume $Ax_1 \neq Ax_2$. Then for $x = (x_1 + x_2)/2$, one has

$$\|x\|_1 \leqslant \frac{\|x_1\|_1 + \|x_2\|_2}{2} \quad \text{and} \quad \|Ax - y\|^2 < \frac{\|Ax_1 - y\|^2 + \|Ax_2 - y\|^2}{2}$$

where the second inequality follows from the strict convexity of the square. This shows that

$$\frac{1}{2\lambda}\|Ax - y\|^2 + \|x\|_1 < \frac{1}{2\lambda}\|Ax_1 - y\|^2 + \|x_1\|_1,$$

which is a contradiction to the optimality of x_1. □

Proposition 32. *For $\lambda > 0$, let x_λ be a solution to $(\mathcal{P}_\lambda(y))$ and denote $\eta_\lambda \overset{\text{def.}}{=} \frac{1}{\lambda} A^*(y - Ax_\lambda)$. We define the "extended support" as*

$$J \overset{\text{def.}}{=} \mathrm{sat}(\eta_\lambda) \overset{\text{def.}}{=} \{i \,;\, |\eta_{\lambda,i}| = 1\}\,.$$

If $\ker(A_J) = \{0\}$ then x_λ is the unique solution of $(\mathcal{P}_\lambda(y))$.

Proof. If $\tilde{x}_\lambda$ is also a minimizer, then by Lemma 3, $Ax_\lambda = A\tilde{x}_\lambda$, so that in particular they share the same dual certificate

$$\eta_\lambda = \frac{1}{\lambda} A^*(y - Ax_\lambda) = \frac{1}{\lambda} A^*(y - A\tilde{x}_\lambda).$$

The first order condition, Proposition 29, shows that necessarily $\mathrm{supp}(x_\lambda) \subset J$ and $\mathrm{supp}(\tilde{x}_\lambda) \subset J$. Since $A_J x_{\lambda,J} = A_J \tilde{x}_{\lambda,J}$, and since $\ker(A_J) = \{0\}$, one has $x_{\lambda,J} = \tilde{x}_{\lambda,J}$, and thus $x_\lambda = \tilde{x}_\lambda$ because of their supports are included in J. □

Proposition 33. *Let $x^\star$ be a solution to $(\mathcal{P}_0(y))$. If there exists $\eta \in \mathcal{D}_0(y, x^\star)$ such that $\ker(A_J) = \{0\}$ where $J \overset{\text{def.}}{=} \mathrm{sat}(\eta)$ then $x^\star$ is the unique solution of $(\mathcal{P}_0(y))$.*

Proof. The proof is the same as for Proposition 32, replacing η_λ by η. □

These propositions can be used to show that if A is drawn according to a distribution having a density over $\mathbb{R}^{p\times n}$, then with probability 1 on the matrix A, the solution to $(\mathcal{P}_\lambda(y))$ is unique. Note that this results is not true if A is non random but y is.

10.1.5 Duality

We now related the first order conditions and "dual certificate" introduced above to the duality theory detailed in Section 16.2. This is not strictly needed to derive the theory of sparse regularization, but this offers an alternative point of view and allows to better grasp the role played by the certificates.

Theorem 11. *For any $\lambda \geqslant 0$ (i.e. including $\lambda = 0$), one has strong duality between $(\mathcal{P}_\lambda(y))$ and*

$$\sup_{p\in\mathbb{R}^P} \left\{ \langle y,\, p\rangle - \frac{\lambda}{2}\|p\|^2 \,;\, \|A^*p\|_\infty \leqslant 1 \right\}. \tag{10.6}$$

One has for any $\lambda \geqslant 0$ that $(x^\star, p^\star)$ are primal and dual solutions if and only if

$$A^*p^\star \in \partial\|\cdot\|_1(x^\star) \quad \Leftrightarrow \quad (I \subset \mathrm{sat}(A^*p) \quad \text{and} \quad \mathrm{sign}(x^\star_I) = A^*_I p)\,, \tag{10.7}$$

where we denoted $I = \mathrm{supp}(x^\star)$, and furthermore, for $\lambda > 0$,

$$p^\star = \frac{y - Ax^\star}{\lambda}.$$

while for $\lambda = 0$, $Ax^\star = y$.

Proof. There are several ways to derive the same dual. One can for instance directly use the Fenchel-Rockafeller formula (**??**). But it is instructive to do the computations using Lagrange duality. One can first consider the following re-writing of the primal problem

$$\min_{x\in\mathbb{R}^N} \{f(z) + \|x\|_1 \,;\, Ax = z\} = \min_{x\in\mathbb{R}^N} \sup_{p\in\mathbb{R}^p} \mathcal{L}(x, z, p) \overset{\text{def.}}{=} f_\lambda(z) + \|x\|_1 + \langle z - Ax,\, p\rangle$$

where $f_\lambda(z) \overset{\text{def.}}{=} \frac{1}{2\lambda}\|z - y\|^2$ if $\lambda > 0$ and $f(z) = \iota_{\{y\}}(z)$ if $\lambda = 0$. For $\lambda > 0$ since f_λ and $\|\cdot\|_1$ are continuous, strong duality holds. For $\lambda = 0$, since the constraint appearing in f_0 is linear (actually a singleton), strong duality holds also. Thus using Theorem 26, one can exchange the min and the max and obtains

$$\max_{p\in\mathbb{R}^P} \;(\min z\; \langle z,\, p\rangle + f_\lambda(z)) + \;(\min x\; \|x\|_1 - \langle x,\, A^*p\rangle) = \max_{p\in\mathbb{R}^P} \; - f^*_\lambda(-p) - (\|\cdot\|_1)^*(A^*p).$$

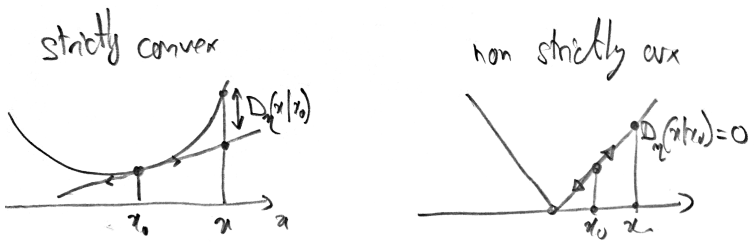


Figure 10.5: Visualization of Bregman divergences.

Using (54), one has that $(\|\cdot\|_1^* = \iota_{\|\cdot\|_\infty \leqslant 1}$. For $\lambda > 0$, one has using Proposition 55 that

$$f_\lambda^* = (\frac{1}{2\lambda}\|\cdot - y\|^2)^* = \frac{1}{\lambda}(\frac{1}{2}\|\cdot - y\|^2)^*(\lambda\cdot) = \frac{1}{2\lambda}\|\lambda\cdot\|^2 + \langle\cdot, y\rangle$$

which gives the desired dual problem. The first order optimality conditions read $Ax^\star = z^\star$ and

$$0 \in \partial\|\cdot\|_1(x^\star) - A^*p^\star \quad \text{and} \quad 0 \in \partial f_\lambda(z^\star) + p^\star.$$

The first condition is equivalent to (10.7). For $\lambda > 0$, f_λ is smooth, and the second condition is equivalent to

$$p^\star = \frac{y - A^*x^\star}{\lambda} \quad \text{and} \quad A^*p^\star \in \partial\|\cdot\|_1(x^\star)$$

which are the desired formula. For $\lambda = 0$, the second condition holds as soon as $z^\star = Ax^\star = y$. □

Note that in the case $\lambda > 0$, (10.6) is strongly convex, and in fact the optimal solution p_λ is computed as an orthogonal projection

$$p_\lambda \in \operatorname*{argmin}_{p\in\mathbb{R}^P} \{\|p - y/\lambda\| \;;\; \|A^*p\|_\infty \leqslant 1\}.$$

The sup in (10.6) is thus actually a max if $\lambda > 0$. If $\lambda > 0$, in case $\ker(A^*) = \mathrm{Im}(A)^\perp = \{0\}$, the constraint set of the dual is bounded, so that the sup is also a max.

10.2 Consistency and Sparsitency

10.2.1 Bregman Divergence Rates for General Regularizations

Here we consider the case of a general regularization of the form

$$\min_{x\in\mathbb{R}^N} \frac{1}{2\lambda}\|Ax - y\|^2 + J(x) \tag{10.8}$$

for a convex regularizer J.

For any $\eta \in \partial J(x_0)$, we define the associated Bregman divergence as

$$D_\eta(x|x_0) \overset{\text{def.}}{=} J(x) - J(x_0) - \langle\eta, x - x_0\rangle.$$

One has $D_\eta(x_0|x_0)$, and since J is convex, one has $D_\eta(x|x_0) \geqslant 0$ **[ToDo: put here drawings]**.

In the case where J is differentiable, since $\partial J(x_0) = \{\nabla J(x_0)\}$, this divergence simply reads

$$D(x|x_0) \overset{\text{def.}}{=} J(x) - J(x_0) - \langle\nabla J(x_0), x - x_0\rangle.$$

If furthermore J is strictly convex, then $D(x|x_0) = 0$ if and only if $x = x_0$, so that $D(\cdot|\cdot)$ is similar to a distance function (but it does not necessarily satisfies the triangular inequality.

If $J = \|\cdot\|^2$, then $D(x|x_0) = \|x - x_0\|^2$ is the Euclidean norm. If $J(x) = \sum_i x_i(\log(x_i) - 1) + \iota_{\mathbb{R}^+}(x_i)$ is the entropy, then

$$D(x|x_0) = \sum_i x_i \log\left(\frac{x_i}{x_{0,i}}\right) + x_{0,i} - x_i$$

is the so-called Kulback-Leibler divergence on $\mathbb{R}^N_+$.

The following theorem, which is due to Burger-Osher, state a linear rate in term of this Bregman divergence.

Theorem 12. *If there exists*

$$\eta = A^*p \in \mathrm{Im}(A^*) \cap \partial J(x_0), \tag{10.9}$$

then one has for any x_λ solution of (10.8)

$$D_\eta(x_\lambda|x_0) \leqslant \frac{1}{2}\left(\frac{\|w\|}{\sqrt{\lambda}} + \sqrt{\lambda}\|p\|\right)^2. \tag{10.10}$$

Futhermore, one has the useful bound

$$\|Ax_\lambda - y\| \leqslant \|w\| + (\sqrt{2}+1)\|p\|\lambda. \tag{10.11}$$

Proof. The optimality of x_λ for (10.8) implies

$$\frac{1}{2\lambda}\|Ax_\lambda - y\|^2 + J(x_\lambda) \leqslant \frac{1}{2\lambda}\|Ax_0 - y\|^2 + J(x_0) = \frac{1}{2\lambda}\|w\|^2 + J(x_0).$$

Hence, using $\langle \eta,\, x_\lambda - x_0\rangle = \langle p,\, Ax_\lambda - Ax_0\rangle = \langle p,\, Ax_\lambda - y + w\rangle$, one has

$$\begin{aligned} D_\eta(x_\lambda|x_0) &= J(x_\lambda) - J(x_0) - \langle \eta,\, x_\lambda - x_0\rangle \leqslant \frac{1}{2\lambda}\|w\|^2 - \frac{1}{2\lambda}\|Ax_\lambda - y\|^2 - \langle p,\, Ax_\lambda - y\rangle - \langle p,\, w\rangle \\ &= \frac{1}{2\lambda}\|w\|^2 - \frac{1}{2\lambda}\|Ax_\lambda - y + \lambda p\|^2 + \lambda\|p\|^2 - \langle p,\, w\rangle \\ &\leqslant \frac{1}{2\lambda}\|w\|^2 + \frac{\lambda}{2}\|p\|^2 + \|p\|\|w\| = \frac{1}{2}\left(\frac{\|w\|}{\sqrt{\lambda}} + \sqrt{\lambda}\|p\|\right)^2. \end{aligned}$$

From the second line above, since $D_\eta(x_\lambda|x_0) \geqslant 0$, one has using Cauchy-Schwartz

$$\|Ax_\lambda - y + \lambda p\|^2 \leqslant \|w\|^2 + 2\lambda^2\|p\|^2 + 2\lambda\|p\|\|w\| \leqslant \|w\|^2 + 2\sqrt{2}\|p\|\|w\|\lambda + 2\lambda^2\|p\|^2 = \left(\|w\| + \sqrt{2}\lambda\|p\|\right)^2.$$

Hence

$$\|Ax_\lambda - y\| \leqslant \|Ax_\lambda - y + \lambda p\| + \lambda\|p\| \leqslant \|w\| + \sqrt{2}\lambda\|p\| + \lambda\|p\|.$$

□

Choosing $\lambda = \|w\|/\|p\|$ in (10.10), one thus obtain a linear rate in term of Bregman divergence $D_\eta(x_\lambda|x_0) \leqslant 2\|w\|\|p\|$. For the simple case of a quadratic regularized $J(x) = \|x\|^2/2$, as used in Section **??**, one sees that the source conditions (10.9) simply reads

$$x_0 \in \mathrm{Im}(A^*)$$

which is equivalent to (8.12) with exponent $\beta = \frac{1}{2}$, and under this condition, (10.10) gives the following sub-linear rate in term of the ℓ^2 norm

$$\|x_0 - x_\lambda\| \leqslant 2\sqrt{\|w\|\|p\|}.$$

[ToDo: This seems inconsistent, this should corresponds to $\beta = 1$ to obtain the same rates in both theorems!]

Note that the "source condition" (10.9) is equivalent to x_0 such that $Ax_0 = y$ is a solution to the constraint problem

$$\min_{Ax=y} J(x).$$

So simply being a solution of the constraint noiseless problem thus implies a linear rate for the resolution of the noisy problem in term of the Bregman divergence.

10.2.2 Linear Rates in Norms for ℓ^1 Regularization

The issue with the control (10.10) of the error in term of Bregman divergence is that it is not "distance-like" for regularizers J which are not strictly convex. This is in particular the case for the ℓ^1 norm $J = \|\cdot\|_1$ which we now study.

The following fundamental lemma shows however that this Bregman divergence for ℓ^1 behave like a distance (and in fact controls the ℓ^1 norm) on the indexes where η does not saturate.

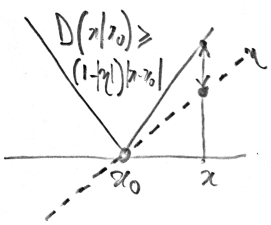


Figure 10.6: Controlling Bregman divergence with the ℓ^1 norm when η is not saturating.

Lemma 4. *For $\eta \in \partial\|\cdot\|_1(x_0)$, let $J \overset{\text{def.}}{=} \mathrm{sat}(\eta)$. Then*

$$D_\eta(x|x_0) \geqslant (1 - \|\eta_{J^c}\|_\infty)\|(x - x_0)_{J^c}\|_1. \tag{10.12}$$

Proof. Note that $x_{0,J^c} = 0$ since $\mathrm{supp}(x_0) \subset \mathrm{sat}(\eta)$ by definition of the sub-differential of the ℓ^1 norm. Since the ℓ^1 norm is separable, each term in the sum defining $D_\eta(x|x_0)$ is positive, hence

$$\begin{aligned} D_\eta(x|x_0) &= \sum_i |x_i| - |x_{0,i}| - \eta_i(x_i - x_{0,i}) \geqslant \sum_{i\in J^c} |x| - |x_0| - \eta_i(x_i - x_{0,i}) \\ &= \sum_{i\in J^c} |x_i| - \eta_i x_i \geqslant \sum_{i\in J^c} (1 - |\eta_i|)|x_i| \geqslant (1 - \|\eta_{J^c}\|_\infty) \sum_{i\in J^c} |x_i| = (1 - \|\eta_{J^c}\|_\infty) \sum_{i\in J^c} |x_i - x_{0,i}|. \end{aligned}$$

□

The quantity $1 - \|\eta_{J^c}\|_\infty > 0$ controls how much η is "inside" the sub-differential. The larger this coefficients, the better is the control of the Bregman divergence.

The following theorem uses this lemma to state the convergence rate of the sparse regularized solution, under the same hypothesis has Proposition 33 (with $x^\star = x_0$).

Theorem 13. *If there exists*

$$\eta \in \mathcal{D}_0(Ax_0, x_0) \tag{10.13}$$

and $\ker(A_J) = \{0\}$ where $J \overset{\text{def.}}{=} \mathrm{sat}(\eta)$ then choosing $\lambda = c\|w\|$, there exists C (depending on c) such that any solution x_λ of $\mathcal{P}(Ax_0 + w)$ satisfies

$$\|x_\lambda - x_0\| \leqslant C\|w\|. \tag{10.14}$$

Proof. We denote $y = Ax_0 + w$. The optimality of x_λ in ($\mathcal{P}_\lambda(y)$) implies

$$\frac{1}{2\lambda}\|Ax_\lambda - y\|^2 + \|x_\lambda\|_1 \leqslant \frac{1}{2\lambda}\|Ax_0 - y\|^2 + \|x_0\|_1 = \frac{1}{2\lambda}\|w\|^2 + \|x_0\|_1$$

and hence

$$\|Ax_\lambda - y\|^2 \leqslant \|w\|^2 + 2\lambda\|x_0\|_1$$

Using the fact that A_J is injective, one has $A_J^+ A_J = \mathrm{Id}_J$, so that

$$\begin{aligned} \|(x_\lambda - x_0)_J\|_1 &= \|A_J^+ A_J (x_\lambda - x_0)_J\|_1 \leqslant \|A_J^+\|_{1,2}\|A_J x_{\lambda,J} - y + w\| \leqslant \|A_J^+\|_{1,2}\left(\|A_J x_{\lambda,J} - y\| + \|w\|\right) \\ &\leqslant \|A_J^+\|_{1,2}\left(\|Ax_\lambda - y\| + \|A_{J^c} x_{\lambda,J^c}\| + \|w\|\right) \\ &\leqslant \|A_J^+\|_{1,2}\left(\|Ax_\lambda - y\| + \|A_{J^c}\|_{2,1}\|x_{\lambda,J^c} - x_{0,J^c}\|_1 + \|w\|\right) \\ &\leqslant \|A_J^+\|_{1,2}\left(\|w\| + (\sqrt{2}+1)\|p\|\lambda + \|A_{J^c}\|_{2,1}\|x_{\lambda,J^c} - x_{0,J^c}\|_1 + \|w\|\right) \end{aligned}$$

where we used $x_{0,J^c} = 0$ and (10.11). One plug this bound in the decomposition, and using (10.12) and (10.10)

$$\begin{aligned}
\|x_\lambda - x_0\|_1 &= \|(x_\lambda - x_0)_J\|_1 + \|(x_\lambda - x_0)_{J^c}\|_1 \\
&\leqslant \|(x_\lambda - x_0)_{J^c}\|_1 \left(1 + \|A_J^+\|_{1,2}\|A_{J^c}\|_{2,1}\right) + \|A_J^+\|_{1,2}\left((\sqrt{2}+1)\|p\|\lambda + 2\|w\|\right) \\
&\leqslant \frac{D_\eta(x|x_0)}{1 - \|\eta_{J^c}\|_\infty}\left(1 + \|A_J^+\|_{1,2}\|A_{J^c}\|_{2,1}\right) + \|A_J^+\|_{1,2}\left((\sqrt{2}+1)\|p\|\lambda + 2\|w\|\right) \\
&\leqslant \frac{\frac{1}{2}\left(\frac{\|w\|}{\sqrt{\lambda}} + \sqrt{\lambda}\|p\|\right)^2}{1 - \|\eta_{J^c}\|_\infty}\left(1 + \|A_J^+\|_{1,2}\|A_{J^c}\|_{2,1}\right) + \|A_J^+\|_{1,2}\left((\sqrt{2}+1)\|p\|\lambda + 2\|w\|\right).
\end{aligned}$$

Thus setting $\lambda = c\|w\|$, one obtains the constant

$$C \overset{\text{def.}}{=} \frac{\frac{1}{2}\left(\frac{1}{\sqrt{c}} + \sqrt{c}\|p\|\right)^2}{1 - \|\eta_{J^c}\|_\infty}\left(1 + \|A_J^+\|_{1,2}\|A_{J^c}\|_{2,1}\right) + \|A_J^+\|_{1,2}\left((\sqrt{2}+1)\|p\|c + 2\right).$$

□

Note that this theorem does not imply that x_λ is a unique solution, only x_0 is unique in general. The condition (10.13) is often called a "source condition", and is strengthen by imposing a non-degeneracy $\ker(A_J) = \{0\}$. This non-degeneracy imply some stability in ℓ^2 sense (10.14). The result (10.14) shows a linear rate, i.e. the (possibly multi-valued) inverse map $y \mapsto x_\lambda$ is Lipschitz continuous.

It should be compared with Theorem 10 on linear methods for inverse problem regularization, which only gives sub-linear rate. The sources conditions in the linear (8.12) and non-linear (10.13) cases are however very different. In the linear case, for $\beta = 1/2$, it reads $x_0 \in \mathrm{Im}(A^*) = \ker(A)^\perp$, which is mandatory because linear method cannot recover anything in $\ker(A)$. On contrary, the non-linear source condition only requires that η to be in $\mathrm{Im}(A^*)$, and is able (in the favorable cases of course) to recover information in $\ker(A)$.

10.2.3 Sparsistency for Low Noise

Theorem 13 is abstract in the sense that it rely on hypotheses which are hard to check. The crux of the problem, to be able to apply this theorem, is to be able to "construct" a valid certificate (10.13). We now give a powerful "recipe" which – when it works – not only give a sufficient condition for linear rate, but also provides "support stability".

For any solution x_λ of $(\mathcal{P}_\lambda(y))$, as already done in (10.2), we define the (unique, independent of the chosen solution) dual certificate

$$\eta_\lambda \overset{\text{def.}}{=} A^* p_\lambda \quad \text{where} \quad p_\lambda \overset{\text{def.}}{=} \frac{y - Ax_\lambda}{\lambda}.$$

The following proposition shows that p_λ converge to a very specific dual certificate of the constrained problem, which we coined "minimal norm" certificate.

Proposition 34. *If $y = Ax_0$ where x_0 is a solution to $(\mathcal{P}_\lambda(y = Ax_0))$, one has*

$$p_\lambda \to p_0 \overset{\text{def.}}{=} \operatorname*{argmin}_{p \in \mathbb{R}^P} \left\{\|p\| \;;\; A^*p \in \mathcal{D}_0(y, x_0)\right\}. \tag{10.15}$$

The vector $\eta_0 \overset{\text{def.}}{=} A^*p_0$ is called the "minimum norm certificate".

Proof. This follows from the fact that p_λ is the unique solution to (10.6) and then applying the same proof as the one done in Proposition 25 to study the small λ limit of penalized problems. □

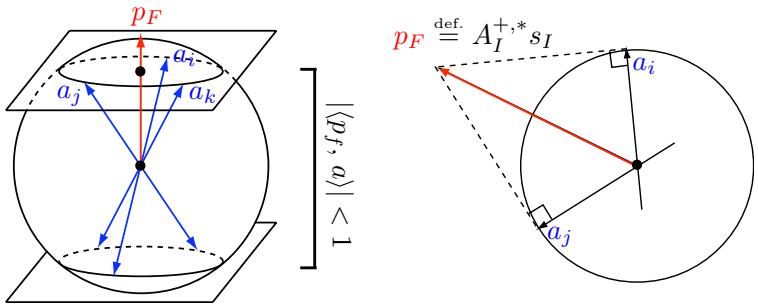


Figure 10.7: Visualization of the condition that $\|\eta_F\|_\infty \leqslant 1$ as a spherical Delaunay triangulation constraint that all Delaunay spherical caps indexes by identifiable vector should be empty of $(\pm a_i)_i$.

This proposition shows that, while dual certificate $\mathcal{D}_0(y, x_0)$ for $\lambda = 0$ are non-unique, taking the limit as $\lambda \to 0$ singles-out a specific one, which is of paramount importance to study stability of the support when the noise and λ are small.

A major difficulty in computing (10.24) is that it should satisfy the non-linear constraint $\|\eta_0\|_\infty$. One thus can "simplify" this definition by removing this ℓ^∞ constraint and define the so-called "minimum norm certificate"

$$\eta_F \stackrel{\text{def.}}{=} A^* p_F \quad \text{where} \quad p_F \stackrel{\text{def.}}{=} \operatorname*{argmin}_{p \in \mathbb{R}^P} \{\|p\| \;;\; A_I^* p = \operatorname{sign}(x_{0,I})\}. \tag{10.16}$$

The notation "η_F" refers to the "Fuchs" certificate, which we named in honour of J-J. Fuchs who first used it to study ℓ^1 minimization.

We insists that p_F is not necessarily a valid certificate (hence the naming "pre-certificate") since one does not have in general $\|\eta_F\|_\infty \leqslant 1$. The vector p_F is a least square solution to the linear system $A_I^* p = \operatorname{sign}(x_{0,I})$, and it can thus be compute in closed form using the pseudo-inverse $p_F = A_I^{*,+} \operatorname{sign}(x_{0,I})$ (see Proposition (23)). In case $\ker(A_I) = \{0\}$, one has the simple formula

$$p_F = A_I (A_I^* A_I)^{-1} \operatorname{sign}(x_{0,I}).$$

Denoting $C \stackrel{\text{def.}}{=} A^* A$ the "correlation" matrix, one has the nice formula

$$\eta_F = C_{\cdot,I} C_{I,I}^{-1} \operatorname{sign}(x_{0,I}). \tag{10.17}$$

The following proposition relates η_F to η_0, and shows that η_F can be used as a "proxy" for η_0

Proposition 35. *If $\|\eta_F\|_\infty \leqslant 1$, then $p_F = p_0$ and $\eta_F = \eta_0$.*

The condition $\|\eta_F\|_\infty \leqslant 1$ implies that x_0 is solution to ($\mathcal{P}_0(y)$). The following theorem shows that if one strengthen this condition to impose a non-degeneracy on η_F, then one has linear rate with a stable support in the small noise regime.

Before proceeding to the proof, let us note that the constraint $\|\eta_F\|_\infty \leqslant 1$ corresponds to the definition of the spherical Delaunay triangulation, as highlighted by Figure 10.7. This remark was made to us by Charles Dossal.

Remark 1 (Operator norm). In the proof, we use the $\ell^p - \ell^q$ matrix operator norm, which is defined as

$$\|B\|_{p,q} \stackrel{\text{def.}}{=} \max \{\|Bu\|_q \;;\; \|u\|_p \leqslant 1\}.$$

For $p = q$, we denote $\|B\|_p \stackrel{\text{def.}}{=} \|B\|_{p,p}$. For $p = 2$, $\|B\|_2$ is the maximum singular value, and one has

$$\|B\|_1 = \max_j \sum_i |B_{i,j}| \quad \text{and} \quad \|B\|_\infty = \max_i \sum_j |B_{i,j}|.$$

Theorem 14. *If*

$$\|\eta_F\|_\infty \leqslant 1 \quad \textit{and} \quad \|\eta_{F,I^c}\|_\infty < 1,$$

and $\ker(A_I) = \{0\}$*, then there exists* C, C' *such that if* $\max(\|w\|, \|w\|/\lambda) \leqslant C$*, then the solution* x_λ *of* ($\mathcal{P}_\lambda(y)$) *is unique, is supported in* I*, and in fact*

$$x_{\lambda,I} = x_{0,I} + A_I^+ w - \lambda (A_I^* A_I)^{-1} \operatorname{sign}(x_{0,I}^\star). \tag{10.18}$$

In particular, $\|x_\lambda - x_0\| = O(\|A^* w\|_\infty) = O(\|w\|)$.

Proof. In the following we denote $T \overset{\text{def.}}{=} \min_{i \in I} |x_{0,i}|$ the signal level, and $\delta \overset{\text{def.}}{=} \|A^* w\|_\infty$ which is the natural way to measure the noise amplitude in the sparse setting. We define $s \overset{\text{def.}}{=} \operatorname{sign}(x_0)$, and consider the "ansatz" (10.18) and thus define the following candidate solution

$$\hat{x}_I \overset{\text{def.}}{=} x_{0,I} + A_I^+ w - \lambda (A_I^* A_I)^{-1} s_I, \tag{10.19}$$

and $\hat{x}_{I^c} = 0$. The goal is to show that $\hat{x}$ is indeed the unique solution of ($\mathcal{P}_\lambda(y)$).
Step 1. The first step is to show sign consistency, i.e. that $\operatorname{sign}(\hat{x}) = s$. This is true if $\|x_{0,I} - \hat{x}_I\|_\infty < T$, and is thus implied by

$$\|x_{0,I} - \hat{x}_I\|_\infty \leqslant K \|A_I^* w\|_\infty + K\lambda < T \quad \text{where} \quad K \overset{\text{def.}}{=} \|(A_I^* A_I)^{-1}\|_\infty, \tag{10.20}$$

where we used the fact that $A_I^+ = (A_I^* A_I)^{-1} A_I^*$.
Step 2. The second step is to check the first order condition of Proposition 32, i.e. $\|\hat{\eta}_{I^c}\|_\infty < 1$, where $\lambda \hat{\eta} = A^*(y - A\hat{x})$. This implies indeed that $\hat{x}$ is the unique solution of ($\mathcal{P}_\lambda(y)$). One has

$$\begin{aligned} \lambda \hat{\eta} &= A^*(A_I x_{0,I} + w - A_I \left(x_{0,I} + A_I^+ w - \lambda (A_I^* A_I)^{-1} s_I \right)) \\ &= A^*(A_I A_I^+ - \mathrm{Id}) w + \lambda \eta_F. \end{aligned}$$

The condition $\|\hat{\eta}_{I^c}\|_\infty < 1$ is thus implied by

$$\|A_{I^c}^* A_I (A_I^* A_I)^{-1}\|_\infty \|A_I^* w\|_\infty + \|A_{I^c}^* w\|_\infty + \lambda \|\eta_{F,I^c}\|_\infty \leqslant R \|A_I^* w\|_\infty - S\lambda < 0 \tag{10.21}$$

$$R \overset{\text{def.}}{=} KL + 1 \quad \text{and} \quad S \overset{\text{def.}}{=} 1 - \|\eta_{F,I^c}\|_\infty > 0$$

where we denoted $L \overset{\text{def.}}{=} \|A_{I^c}^* A_I\|_\infty$, and also we used the hypothesis $\|\eta_{F,I^c}\|_\infty < 1$.
Conclusion. Putting (10.20) and (10.21) together shows that $\hat{x}$ is the unique solution if (λ, w) are such that the two linear inequations are satisfies

$$\mathcal{R} = \left\{ (\delta, \lambda) \; ; \; \delta + \lambda < \frac{T}{K} \quad \text{and} \quad R\delta - S\lambda < 0 \right\}$$

This region $\mathcal{R}$ is trianglular-shaped, and includes the following "smaller" simpler triangle

$$\tilde{\mathcal{R}} = \left\{ (\delta, \lambda) \; ; \; \frac{\delta}{\lambda} < \frac{S}{R} \quad \text{and} \quad \lambda < \lambda_{\max} \right\} \quad \text{where} \quad \lambda_{\max} \overset{\text{def.}}{=} \frac{T(KL+1)}{K(R+S)}. \tag{10.22}$$

□

It is important to realize that Theorem 14 operates in a "small noise" regime, i.e. $\|w\|$ (and hence λ) needs to be small enough for the support to be identifiable (otherwise small amplitude comment of x_0 will be killed by the regularization). In contrast, Theorem 13 is "global" and holds for any noise level $\|w\|$. The price to pay is that one has no controls about the support (and one does not even knows whether x_λ is unique) and than the constant involved are more pessimistic.

A nice feature of this proof is that it gives access to explicit constant, involving the three key parameter K, L, S, which controls:

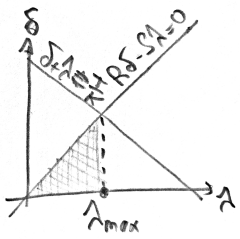


Figure 10.8: Zone in the (λ, δ) where sign consistency occurs.

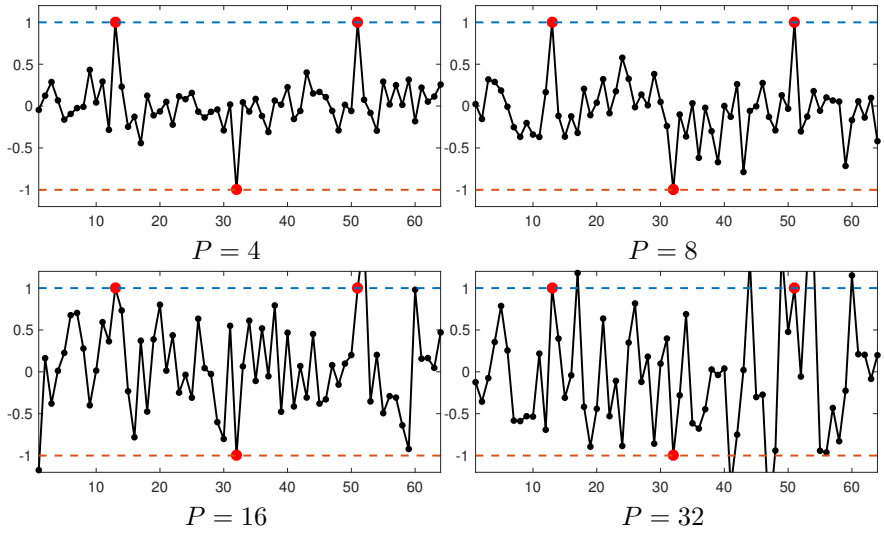


Figure 10.9: Display of certificate η_F for a $A \in \mathbb{R}^{p \times n}$, $n = 64$, with independent Gaussian entries.

– K accounts for the conditionning of the operator on the support I ;
– L accounts for the worse correlation between atoms inside and outside the support ;
– S accounts for how much the certificates η_F is non-degenerate.

The constant on $\|A^* w\|/\lambda$ and on λ are given by (10.22). Choosing (which is in practice impossible, because it requires knowledge about the solution) the smallest possible λ gives $\lambda = \delta \frac{S}{R}$ and in this regime the error is bounded in ℓ^∞ (using other error norms would simply leads to using other matrix norm)

$$\|x_0 - x_\lambda\|_\infty \leqslant \left(1 + \frac{KL+1}{S}\right) K\delta.$$

The crux of the analysis of the performance (in term of support stability) of ℓ^1 regularization is to be able to say wether, for some class of signal x_0 of interest, η_F is a valid certificate, i.e. $\|\eta_F\|_\infty \leqslant 1$. Figure 10.9 displays numerically what one obtains when A is random. One see that η_F is non-degenerate when P is large enough. Section 11.2 performs a mathematical analysis of this phenomena.

10.2.4 Sparsistency for Arbitrary Noise

A flow of the previous approach is that it provides only asymptotic sparsistency when the noise is small enough with respect to the signal. In particular, it cannot be used to asses the performance of sparse recovery for approximately sparse (e.g. compressible signal), for which the residual error is of the error of the signal itself (and thus not small).

This can be aleviated by controlling all possible certificate associated to all the sign pattern of a given support. This is equivalent to the ERC condition of Tropp. **[ToDo: write me]**

The proof proceeds by restricting the optimization to the support, which is still a convex program, and then showing that this candidate solution is indeed the correct one. Since one does not know in advance the sign of this candidate, this is why one needs to control all possible certificates.

10.3 Sparse Deconvolution Case Study

Chapter 11 studies the particular case where A is random, in which case it is possible to make very precise statement about wether η_F is a valid certificate.

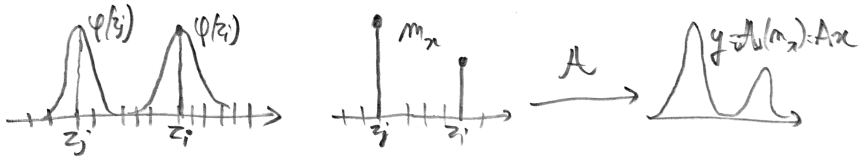


Figure 10.10: Convolution operator.

Another interesting case study, which shows the limitation of this approach, is the case of "super-resolution". It corresponds to inverse problems where the columns $(a_i)_i$ of A are highly correlated, since typically they are obtained by sampling a smooth kernel.

We thus consider the case where $a_i = \varphi(z_i)$ where the $(z_i)_i \subset \mathbb{X}$ is a sampling grid of a domain $\mathbb{X}$ and $\varphi : \mathbb{X} \to \mathcal{H}$ is a smooth map. One has

$$Ax = \sum_i x_i \varphi(z_i).$$

Since we seek for sparse x, one can view x as representing the weights of a discrete measure $m_x \overset{\text{def.}}{=} \sum_{i=1}^N x_i \delta_{z_i}$ where the dirac masses are constraint to be on the sampling grid.

The matrix A is a discretized version of an infinite dimensional operator mapping Radon measures to vectors of observations $\mathcal{A} : m \in \mathcal{M}(\mathbb{X}) \mapsto y = \mathcal{A}m \in \mathcal{H}$

$$\mathcal{A}(m) \overset{\text{def.}}{=} \int_{\mathbb{X}} \varphi(x) \mathrm{d}m(x).$$

Indeed, one has for discrete measure $\mathcal{A}(m_x) = Ax$.

A typical example is when using $\mathcal{H} = L^2(\mathbb{X})$ with $\mathbb{X} = \mathbb{R}^d$ or $\mathbb{X} = \mathbb{T}^d$ and $\varphi(z) = \tilde{\varphi}(z - \cdot)$, which corresponds to a convolution

$$(\mathcal{A}m)(z) = \int \tilde{\varphi}(z - x) \mathrm{d}m(x) = (\tilde{\varphi} \star m)(z).$$

Note that here $\mathcal{H}$ is infinite dimensional, and to get finite dimensional observations, it suffices to sample the output and consider $\varphi(z) = (\varphi(z - r_j))_{j=1}^P$ (note that the observation grid $r \in \mathbb{X}^P$ can be different from the recovery grid $z \in \mathbb{X}^N$).

Another example, actually very much related, is when using $\varphi(z) = (e^{\mathrm{i}kz})_{k=-f_c}^{f_c}$ on $\mathbb{X} = \mathbb{T}$, so that $\mathcal{A}$ corresponds to computing the f_c low-frequencies of the Fourier transform of the measure

$$\mathcal{A}(m) = \left(\int_{\mathbb{T}} e^{\mathrm{i}kx} \mathrm{d}m(x) \right)_{k=-f_c}^{f_c}.$$

The operator $\mathcal{A}^*\mathcal{A}$ is a convolution against an ideal low pass (Dirichlet) kernel. By weighting the Fourier coefficients, one can this way model any low pass filtering on the torus.

Yet another interesting example on $\mathbb{X} = \mathbb{R}^+$ is the Laplace transform

$$\mathcal{A}(m) = z \mapsto \int_{\mathbb{R}^+} e^{-xz} \mathrm{d}m(x).$$

We denote the "continuous" covariance as

$$\forall\, (z, z') \in \mathbb{X}^2, \quad \mathcal{C}(z, z') \overset{\text{def.}}{=} \langle \varphi(z), \varphi(z') \rangle_{\mathcal{H}}.$$

Note that this $\mathcal{C}$ is the kernel associated to the operator $\mathcal{A}^*\mathcal{A}$. The discrete covariance, defined on the computational grid is $C = (\mathcal{C}(z_i, z_i')))_{(i,i')} \in \mathbb{R}^{N \times N}$, while its restriction to some support set I is $C_{I,I} = (\mathcal{C}(z_i, z_i')))_{(i,i') \in I^2} \in \mathbb{R}^{I \times I}$.

Using (10.17), one sees that η_F is obtained as a sampling on the grid of a "continuous' " certificate $\tilde{\eta}_F$

$$\eta_F = (\tilde{\eta}_F(z_i))_{i=1}^N \in \mathbb{R}^N,$$

$$\text{where} \quad \tilde{\eta}_F(x) = \sum_{i \in I} b_i \mathcal{C}(x, z_i) \quad \text{where} \quad b_I = C_{I,I}^{-1} \operatorname{sign}(x_{0,I}), \tag{10.23}$$

so that η_F is a linear combination of I basis functions $(\mathcal{C}(x, z_i))_{i \in I}$.

The question is wether $\|\eta_F\|_{\ell^\infty} \leqslant 1$. If the gris is fine enough, i.e. N large enough, this can only hold if $\|\tilde{\eta}_F\|_{L^\infty} \leqslant 1$. The major issue is that $\tilde{\eta}_F$ is only constrained by construction to interpolate $\operatorname{sign}(x_{0,i})$ are points $z_{0,i}$ for $i \in I$. So nothing prevents $\tilde{\eta}_F$ to go outside $[-1, 1]$ around each interpolation point. Figure 10.11 illustrates this fact.

In order to guarantee this property of "local" non-degeneracy around the support, one has to impose on the certificate the additional constraint $\eta'(z_i) = 0$ for $i \in I$. This leads to consider a minimum pre-certificate with vanishing derivatives

$$\eta_V \overset{\text{def.}}{=} A^* p_V \quad \text{where} \quad p_V \in \operatorname*{argmin}_{p \in L^2(\mathbb{R})} \left\{ \|p\|_{L^2(\mathbb{R})} \;;\; \tilde{\eta}(z_I) = \operatorname{sign}(x_{0,I}), \tilde{\eta}'(z_I) = \mathbf{0}_I \right\}. \tag{10.24}$$

where we denoted $\tilde{\eta} = \bar{\psi} \star p$. Similarly to (10.23), this vanishing pre-certificate can be written as a linear combination, but this time of $2|I|$ basis functions

$$\tilde{\eta}_V(x) = \sum_{i \in I} b_i \mathcal{C}(x, z_i) + c_i \partial_2 \mathcal{C}(x, z_i),$$

where $\partial_2 \mathcal{C}$ is the derivative of $\mathcal{C}$ with respect to the second variable, and (b, c) are solution of a $2|I| \times 2|I|$ linear system

$$\begin{pmatrix} b \\ c \end{pmatrix} = \begin{pmatrix} (\mathcal{C}(x_i, x_{i'}))_{i,i' \in I^2} & (\partial_2 \mathcal{C}(x_i, x_{i'}))_{i,i' \in I^2} \\ (\partial_1 \mathcal{C}(x_i, x_{i'}))_{i,i' \in I^2} & (\partial_1 \partial_2 \mathcal{C}(x_i, x_{i'}))_{i,i' \in I^2} \end{pmatrix}^{-1} \begin{pmatrix} \operatorname{sign}(x_{0,I}) \\ \mathbf{0}_I \end{pmatrix}.$$

The associated continuous pre-certificate is $\tilde{\eta}_V = \bar{\psi} \star p_V$, and η_V is a sampling on the grid of $\tilde{\eta}_V$. Figure 10.9 shows that this pre-certificate η_V is much better behaved than η_F. If $\|\eta_V\|_\infty \leqslant 1$, one can apply (13) and thus obtain a linear convergence rate with respect to the ℓ^2 norm on the grid. But for very fine grid, since one is interested in sparse solution, the ℓ^2 norm becomes meaningless (because the L^2 norm is not defined on measures). Since η_V is different from η_F, one cannot directly applies Theorem 14: the support is not stable on discrete grids, which is a fundamental property of super-resolution problems (as opposed to compressed sensing problems). The way to recover interesting results is to use and analyze methods without grids. Indeed, after removing the grid, one can show that η_V becomes the minimum norm certificate (and is the limit of η_λ).

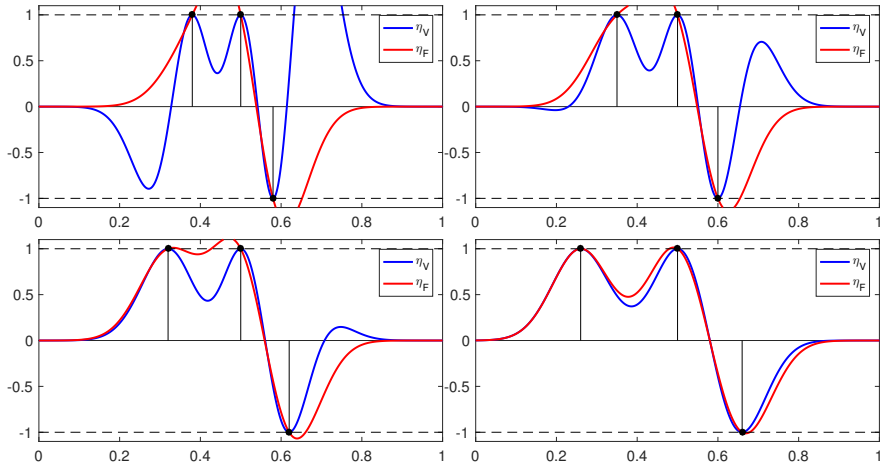


Figure 10.11: Display of "continuous" certificate η_F and η_V for A being a convolution operator.

Chapter 11

Compressed Sensing

This chapter details an important class of inverse problems, which corresponds to using "random" forward operators Φ. This is interesting from an applicative point of view since it allows to model a novel class of imaging devices which can potentially have improved resolution with respect to traditional operators (e.g. low-pass filters for usual cameras) when using in conjunction with sparse regularization technics. This is also interesting from a theoretical point of view, since the mathematical analysis becomes much simpler than with deterministic operators, and one can have good recovery and stability performances. Let us however stress that the "physical" creation of hardware that fulfils the theoretical hypothesis, in particular in medical imaging, is still largely open (put aside some restricted areas), although the theory gives many insightful design guides to improve imaging devices.

The main references for this chapter are [17, 16, 23].

11.1 Motivation and Potential Applications

11.1.1 Single Pixel Camera

In order to illustrate the exposition, we will discuss the "single pixel camera" prototype developed at Rice University [**?**], and which is illustrated by the figure 11.1 (left). It is an important research problem of developing a new class of cameras allowing to obtain both the sampling and the compression of the image. Instead of first sampling very finely (ie with very large Q) the analog signal $\tilde{f}$ to obtain a $f \in \mathbb{R}^Q$ image then compressing enormously (ie with M small) using (**??**), we would like to dispose directly of an economic representation $y \in \mathbb{R}^P$ of the image, with a budget P as close to M and such that one is able to "decompress" y to obtain a good approximation of the image f_0.

The "single-pixel" hardware performs the compressed sampling of an observed scene $\tilde{f}_0$ (the letter "R" in Figure 11.1), which is a continuous function indicating the amount of light $\tilde{f}_0(s)$ reaching each point $s \in \mathbb{R}^2$ of the focal plane of the camera. To do this, the light is focused against a set of Q micro-mirrors aligned on the focal plane. These micro-mirrors are not sensors. Unlike conventional sampling (described in Section **??**), they do not record any information, but they can each be positioned to reflect or absorb light. To obtain the complete sampling/compression process, one very quickly changes P times the configurations of the micro-mirrors. For $p = 1, \ldots, P$, one sets $\Phi_{p,q} \in \{0, 1\}$, depending on whether the micromirror at position q has been placed in the absorbing (0) or reflective (value 1) position at step p of the acquisition. The total light reflected at step p is then accumulated into a single sensor (hence the name "single pixel", in fact it is rather a "single sensor"), which achieves a linear sum of the reflected intensities to obtain the recorded $y_p \in \mathbb{R}$ value. In the end, if the light intensity arriving on the surface c_q of the mirror indexed by $f_q = \int_{c_q} \tilde{f}_0(s)\mathrm{d}s$ is denoted (as in the **??** section) as q, the equation that links the discrete image $f \in \mathbb{R}^Q$

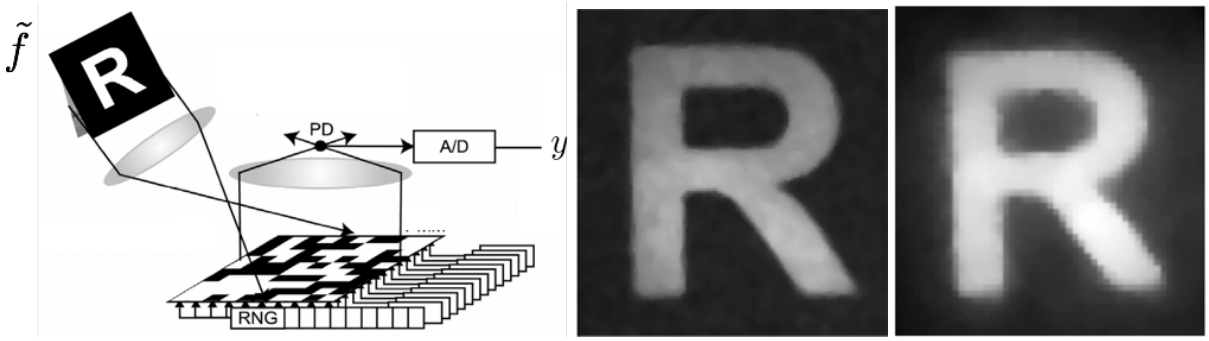


Diagram of the device f $f^\star$, $P/Q = 6$

Figure 11.1: Left: diagram of the single-pixel acquisition method. Center: image $f_0 \in \mathbb{R}^Q$ "ideal" observed in the focal plane of the micro-mirrors. Right: image $f_0^\star = \Psi x^\star$ reconstructed from observation $y \in \mathbb{R}^P$ with a compression factor $P/Q = 6$ using ℓ^1-type regularization.

"seen through the mirrors" to the P measures $y \in \mathbb{R}^P$ is

$$\forall\, p = 1, \ldots, P, \quad y_p \approx \sum_q \Phi_{p,n} \int_{c_n} \tilde{f}_0(s)\mathrm{d}s = (\Phi f_0)_p,$$

(here $\approx$ accounts for some noise), which corresponds to the usual forward model of inverse problems

$$y = \Phi f_0 + w \in \mathbb{R}^P$$

where w is the noise vector. It is important to note that the mirrors do not record anything, so in particular the f_0 discrete image is never calculated or recorded, since the device directly calculates the compressed representation y from the analog signal $\tilde{f}_0$. The term w models here the acquisition imperfections (measurement noise). The compressed sampling therefore corresponds to the transition from the observed scene $\tilde{f}_0$ to the compressed vector y. The "decompression" corresponds to the resolution of an inverse problem, whose goal is to find a good approximation of f_0 (the discrete image "ideal" as seen by the micro-mirrors) from y.

11.1.2 Sparse Recovery

In order to reconstruct an approximation of the (unknown) image f_0, following Section 9.2, we assume it is sparse in some dictionary Ψ. Denoting $A \overset{\text{def.}}{=} \Psi\Phi \in \mathbb{R}^{P\times N}$, this leads us to consider the usual ℓ^1 regularized poblem (9.10)

$$x_\lambda \in \operatorname*{argmin}_{x\in\mathbb{R}^N} \frac{1}{2\lambda}\|y - Ax\|^2 + \|x\|_1, \qquad (\mathcal{P}_\lambda(y))$$

so that the reconstructed image is $f_\lambda = \Psi x_\lambda$. We also sometimes consider the constraint problem

$$x_\varepsilon \in \operatorname*{argmin}_{\|Ax-y\|\leqslant\varepsilon} \|x\|_1, \qquad (\mathcal{P}^\varepsilon(y))$$

where, for the sake of simplicity, we set $\varepsilon = \|w\|$ (which we assume is known). From a mathematical point of view, these problem are equivalent in the sense that there exists a bijection between λ and ε which links it solution. But in practice, this bijection is not explicitly known and depends on y.

Here, it is important to remember that A is drawn from a random matrix ensemble. For an arbitrary Ψ, it is hard to analyze this random distribution. If Ψ is orthogonal, and the distribution of the columns of Φ are invariant by rotation (which is the case if the entries are i.i.d. Gaussian), then A has the same distribution as Φ. In the following, we thus directly models the distribution of A and assumes it has some nice property (typically it is close to being Gaussian i.i.d.).

11.2 Dual Certificate Theory and Non-Uniform Guarantees

11.2.1 Random Projection of Polytopes

When there is no noise, $w = 0$ a way to tackle the problem is to use the caracterization of solutions of $(\mathcal{P}^0(Ax_0)) = (\mathcal{P}_0(Ax_0))$ given in Section 10.1.2. According to Proposition 28, identifiable vectors with sparsity $\|x_0\|_0 = s$ corresponds to s-dimensional faces of the ℓ^1 balls B_1 which are mapped to face of the projected polytope AB_1. This leads to a combinatorial problems to count the number of face of random polytope. Donoho and Tanner were able to perform a sharp analysis of this problem. They showed the existence of two regimes, using two functions C_A, C_M so that, with high probability (i.e. a probability converging exponentially fast to 1 with (n, p)) on the matrix A

– All x_0 so that $\|x_0\|_0 \leqslant C_A(P/N)P$ are identifiable.

– Most x_0 so that $\|x_0\|_0 \leqslant C_M(P/N)P$ are identifiable.

For instance, they show that $C_A(1/4) = 0.065$ and $C_M(1/4) = 0.25$. Figure 11.5 illustrates numerically these two phase transitions. This analysis can be shown to be sharp in high dimension, i.e. when $\|x_0\|_0 > C_M(P/N)P$, then x_0 is not identifiable with high probability (this corresponds to a phase transition phenomena). For large dimensions (N, P), the scaling given by C_M describe very well what one observe in practice. For $P = N/4$ (compression of a factor 4), one retrieve in practice all vector with sparsity smaller than P/N. The function C_M can be computed numerically, and it can be shown to have a logarithmic grows $C_M(r) \sim \log(r)$ for small r. This suggests that for high compression regime, one recovers with ℓ^1 minimization almost all vector with a sparsity $\|x_0\|_0$ proportional (up to log factor) to the number of measurements P.

11.2.2 Random Matrices

The analysis of the performance ℓ^1 minimization to solve compressed sensing problem is made possible because of the very precise understanding of the distribution of the singular values of certain random ensemble let us illustrate this in the Gaussian case, which is the simplest, and is very illustrative.

An important part of the recovery proof relies on controlling the correlation matrix $A_I^* A_I$ of selected columns, and even more importantly, its inverse $(A_I A_I)^{-1}$. These random matrices are called Wishart matrices and inverse Wishart matrices. Such a matrix $B = A_I$ is of size (P, s) and is also drawn from the Gaussian ensemble. Fixing s and letting $P \to +\infty$, one has thanks to the low of large numbers $B^* B \to \mathrm{Id}_s$ almost surely. This is however not a very realistic setting, since in general, one hope to have s almost equal, up to log factor, to P.

Linear growth $P = s/\beta$. A quite extreme setting is when s grows proportionally to P, and impose $s/P = \beta$. In this case, the eigenvalues of $B^* B$ are, for large p, essentially contained in the interval $[\lambda_-, \lambda_+]$ where $\lambda_\pm = (1 \pm \sqrt{\beta})^2$, $\beta \overset{\text{def.}}{=} s/p$, in the sense that the probability distribution of eigenvalues converges (in the weak sense of measures) toward the Marcenko-Pastur law

$$\forall\, (u, v) \in \mathbb{R}_+^2, \quad \mathbb{P}(\mathrm{eig}(B^\top B) \in [u, v]) \overset{p \to +\infty}{\longrightarrow} \int_u^v f_\beta(\lambda) \mathrm{d}\lambda$$

where one fix the ratio $\beta = s/P$, and the Marcenko-Pastur law is

$$f_\beta(\lambda) \overset{\text{def.}}{=} \frac{1}{2\pi\beta\lambda} \sqrt{(\lambda - \lambda_-)_+ (\lambda_+ - \lambda)}\, 1_{[\lambda_-, \lambda_+]}(\lambda).$$

Figure (11.3) illustrates this convergence.

Super-linear grows $P = s \log(\ldots)$. In order to have a better concentration of the singular values of $A_I^* A_I$ around 1, one needs to have a slightly super-linear growth of P with s. In this setting one has that $A_I^* A_I$. In order to derive non-asymptotic (i.e. with explicit constants) results, one can use a celebrated

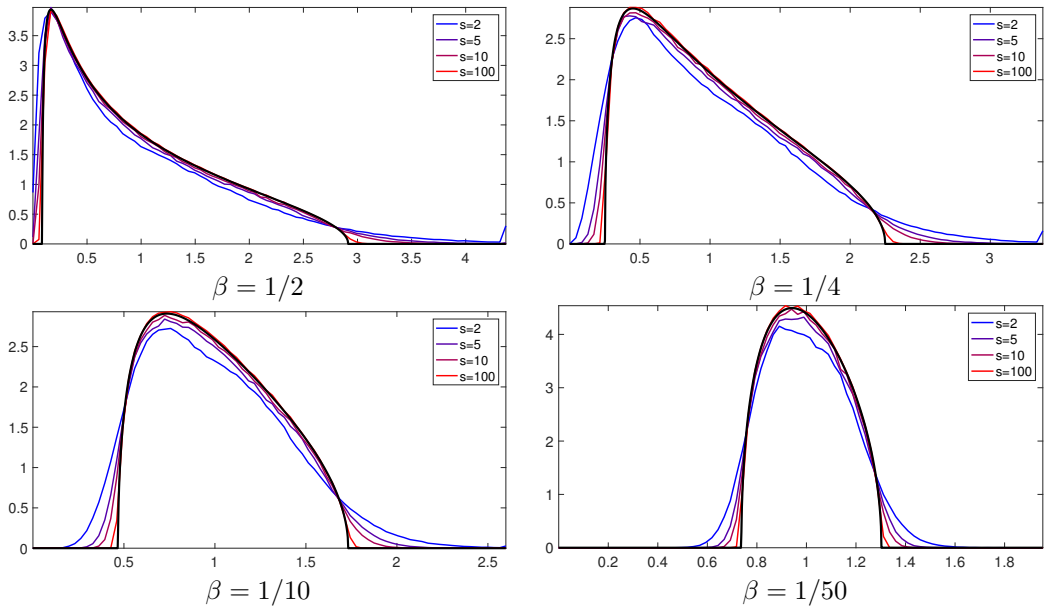


Figure 11.2: Illustration of the convergence toward the Marcenko-Pastur law.

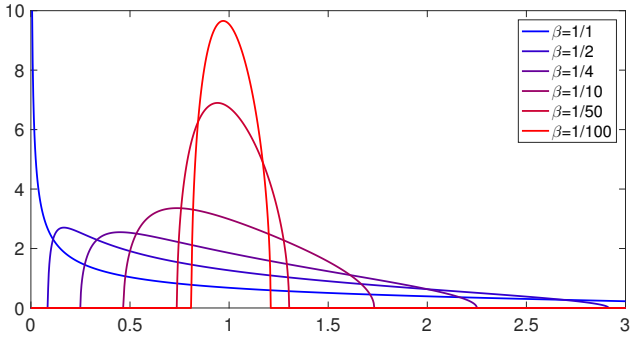


Figure 11.3: Display of the Marcenko-Pastur distribution f_β for various β.

concentration inequality due to Talagrand, which assert that one has a fast concentration of this randomized covariance $A_I^* A_I$ toward its expectation Id_s.

$$\mathbb{P}\left(\|A_I^* A_I - \mathrm{Id}_s\|_{\mathrm{op}} \geqslant t + \sqrt{\frac{s}{P}} \right) \leqslant e^{-\frac{t^2 s}{2}}. \tag{11.1}$$

11.2.3 Dual Certificates

In order to analyze recovery performance, one can looks not only for ℓ^2 stability ($\|x_\lambda - x_0\| \sim \|w\|$) but also that x_λ has the same support as x_0 when $\|w\|$ is small. As detailed in Section 10.2.3, this requires to ensure that the pre-certificate η_F defined in (10.24) is non-degenerated, i.e.

$$\|\eta_F\|_\infty \leqslant 1 \quad \text{where} \quad \eta_F = A^* A_I (A_I^* A_I)^{-1} \operatorname{sign}(x_{0,I}). \tag{11.2}$$

Figure 10.9 suggests that this should be the case if P is large enough with respect to $\|x_0\|$. This theorem backup this observation.

Coherence-based analysis. We first perform a crude analysis using the so-called coherence of the matrix $A = (a_j)_{j=1}^N$ where the $a_j \in \mathbb{R}^P$ are the columns of A, which we assume to be normalized $\|a_j\| = 1$

$$\mu \overset{\text{def.}}{=} \max_{i \neq j} |\langle a_i,\, a_j \rangle| = \|A^* A - \mathrm{Id}_N\|_\infty \tag{11.3}$$

where $\|C\|_\infty = \max_{i,j} |C_{i,j}|$. The coherence is 0 for an orthogonal matrix, and is always smaller than 1, $\mu \in [0,1]$. The smaller the coherence, the better conditioned the inverse problem $Ax = y$ is, and the more likely is the certificate η_F to be non-degenerate, as shown by the following proposition.

Proposition 36. *One has, denoting $s = \|x_0\|_0 = |I|$ where $I = \operatorname{supp}(x_0)$, for $\mu < \frac{1}{s-1}$,*

$$\|\eta_{F,I^c}\|_\infty \leqslant \frac{s\mu}{1-(s-1)\mu} \quad \textit{and} \quad \|p_F\|^2 \leqslant \frac{s}{1-(s-1)\mu}. \tag{11.4}$$

In particular, if $s < \frac{1}{2}\left(1 + \frac{1}{\mu}\right)$, $\|\eta_{F,I^c}\|_\infty < 1$ and one can thus apply the recovery Theorem 14.

Proof. We recall that the ℓ^∞ operator norm (see Remark 1) is

$$\|B\|_\infty = \max_i \sum_j |B_{i,j}|.$$

We denote $C = A^* A$. One has

$$\|A_{I^c}^* A_I\|_\infty = \max_{j \in I^c} \sum_{i \in I} C_{i,j} \leqslant s\mu \quad \text{and} \quad \|\mathrm{Id}_s - A_I^* A_I\|_\infty = \max_{j \in I} \sum_{i \in I, i \neq j} C_{i,j} \leqslant (s-1)\mu$$

One also has

$$\begin{aligned} \|(A_I^* A_I)^{-1}\|_\infty &= \|((\mathrm{Id}_s - A_I^* A_I) - \mathrm{Id}_s)^{-1}\|_\infty = \|\sum_{k \geqslant 0} (\mathrm{Id}_s - A_I^* A_I)^k\|_\infty \\ &\leqslant \sum_{k \geqslant 0} \|\mathrm{Id}_s - A_I^* A_I\|_\infty^k \leqslant \sum_{k \geqslant 0} ((s-1)\mu)^k = \frac{1}{1-(s-1)\mu} \end{aligned}$$

which is legit because the matrix series indeed converge since $(s-1)\mu < 1$. Using these two bounds, one has

$$\|\eta_{F,I^c}\|_\infty = \|A_{I^c}^* A_I (A_I^* A_I)^{-1} \operatorname{sign}(x_{0,I})\|_\infty \leqslant \|A_{I^c}^* A_I\|_\infty \|(A_I^* A_I)^{-1}\|_\infty \|\operatorname{sign}(x_{0,I})\|_\infty \leqslant (s\mu) \times \frac{1}{1-(s-1)\mu} \times 1.$$

One has

$$\frac{s\mu}{1-(s-1)\mu} \quad \Longleftrightarrow \quad 2s\mu < 1+\mu$$

which gives the last statement. One also has

$$\|p_F\|^2 = \langle (A_I^* A_I)^{-1} s_I,\, s_I \rangle \leqslant \|(A_I^* A_I)^{-1}\|_\infty \|s\|_1 \leqslant \frac{s}{1-(s-1)\mu}$$

□

Note that this proof actually shows that if $s < \frac{1}{2}\left(1+\frac{1}{\mu}\right)$, all certificate η_F are valid, for any sign pattern $\mathrm{sign}(x_0)$. This actually implies a much stronger stability in the sense that whatever the noise w (which might not be small), the support of x_λ is included (not necessarily equal) in the one of x_0.

One can show that one always has

$$\mu \geqslant \sqrt{\frac{N-P}{P(N-1)}} \tag{11.5}$$

which is equivalent to $1/\sqrt{P}$ for $N \gg P$. For Gaussian matrix $A \in \mathbb{R}^{P\times N}$, one has for large $(N,P) \to +\infty$

$$\mu \sim \sqrt{\log(PN)/P}$$

which shows that Gaussian matrix are close to being optimal for the bound (11.5) if $N \gg P$. Applying Proposition 36 thus shows that ℓ^1 regularization is able to recover with a stable support vector with less than $s \sim O(\sqrt{P})$ (ignoring log terms) non-zero coefficients. In face, we will show now that it does much better and recover a proportional number $s \sim O(P)$. This is because the coherence bound (11.4) is very pessimistic.

Randomized analysis of the Fuchs certificate. We consider here a class of sufficiently "random" matrices.

Definition 1 (sub-Gaussian random matrix)**.** *A random matrix $\sqrt{P}\,A$ is said to be sub-Gaussian if its entries are independent such that $\mathbb{E}(A_{i,j}) = 0$ (zero mean) $\mathbb{E}(A_{i,j}^2) = 1/P$ and*

$$\mathbb{P}(|\sqrt{P}\,A_{i,j}| \geqslant t) \leqslant \beta e^{-\kappa t^2}.$$

Note that its entries does not needs to be identically distributed, but the sub-Gaussiannity parameter (β,κ) should not depend on (i,j). Note also the presence of the normalization factor $\sqrt{P}$, which is here to ensure $\mathbb{E}(\|a_j\|^2) = 1$ where a_j are the columns of A.

Typical example of sub-Gaussian random matrix are Gaussian or Bernoulli matrices.

Theorem 15. *For a given $x_0 \in \mathbb{R}^N$, denoting $s = \|x_0\|_0$, and assuming A is sub-Gaussian, then for for any $0 < \varepsilon < 1$ provided that*

$$P \geqslant \frac{4c}{1-\delta} s \log(2N/\varepsilon) \quad \text{where} \quad \delta^2 \overset{\text{def.}}{=} \frac{C}{4c}\left(\frac{7}{\log(2N/\varepsilon)} + \frac{2}{s}\right)$$

condition (11.2) holds with probability $1-\varepsilon$, so that x_λ has the same support and sign as x_0 when $(\|w\|, \|w\|/\lambda)$ is small enough. The constant C, c, $C = \frac{2}{3\tilde{c}}$ where $\tilde{c} \overset{\text{def.}}{=} \frac{\kappa^2}{4\beta+2\kappa}$ only depends on the sub-Gaussianity parameter (β,κ) appearing (1), and for Gaussian or Benoulli matrices, $c = 1/2$.

For a Gaussian matrix, the scaling is that one should have $P \geqslant (2+\delta) s \log(N)$ with a high probability.

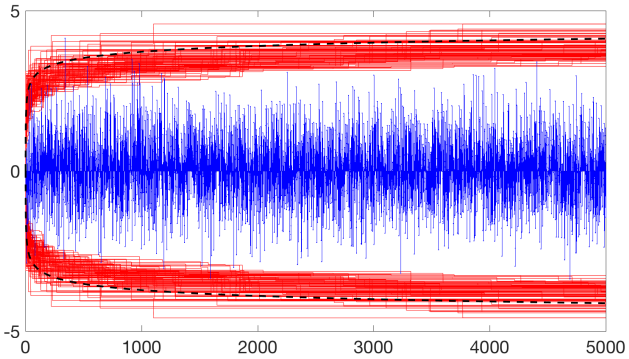


Figure 11.4: Graphical display of how the maximum of N i.i.d. gaussians concentrates tightly just below the $\sqrt{2\log(N)}$ dashed curve.

Proof. We only give the main insight for the proof. Its crux relies on the fact that $\|A_{I^c}^* p_F\|_\infty \leqslant 1$ reads

$$\max_{j \notin I} |\langle a_j, p_F \rangle| \leqslant 1$$

where $p_F = A_I^{*+} s_{0,I}$ is *independent* from the vectors $(a_j)_{j \notin I}$ it is correlated against. This allows one to check this condition by first controlling $\|p_F\|$ and then making as if p_F was a fixed deterministic vector. Noticing

$$\|p_F\|^2 = \langle A_I (A_I^* A_I)^{-1} s_{0,I},\, A_I (A_I^* A_I)^{-1} s_{0,I} \rangle = \langle (A_I^* A_I)^{-1} s_{0,I},\, s_{0,I} \rangle,$$

the heuristic reasoning is that, following what we said in Section (11.2.2), if P grows slightly (logarithmically) faster than s, $A_I^* A_I$ is close to Id_s (see Talagrand inequality (11.1)), so that

$$\|p_F\|^2 \sim \|s_{0,I}\|^2 = s \tag{11.6}$$

For a *fixed* p_F, one has that $\langle a_j, p_F \rangle$ is Gaussian distributed with variance $\|p_F\|^2/P$, and we use the well known fact (already encountered for denoising using thresholding) that the maximum of $P - s$ such vectors concentrates just below the universal threshold $\|p_F\| \sqrt{2\log(N-s)/P}$. Using the estimate (11.6), one sees that $\|p_F\|_\infty \leqslant 1$ is implied by $2\log(N)s/P \leqslant 1$, which gives the sharp scaling $P \geqslant 2\log(N)s$.

In order to get robustness to noise, one needs to impose that $\|A_{I^c}^* p_F\| < 1$, which can be achieve by using a slightly stronger scaling $P \geqslant 2(1+\delta)\log(N)s$ for a small δ.

One can actually make this reasoning very precise, because quite surprisingly, it turns out that $Ps/\|p_F\|^2$ is actually distributed according to a χ^2 variable with $P - s + 1$ degrees of freedom (i.e. the sum of $P - s + 1$ squares of Gaussian variables). Indeed, for $s = 1$, one immediately sees that $P/\|p_F\|^2$ is χ^2 with P degrees of freedom. The general case is more involved, and the proof relies on the fact that the isotropy of the distribution implies that $p - P/\|p_F\|^2$ is the square of the distance between the first column of A_I and the linear space spanned by the other columns (hence $P - s + 1$ degrees of freedom). Since one has a very good understanding of the clustering of such a χ^2 variable around its means, one can thus show that $\|p_F\| \leqslant (1-\delta)\sqrt{s}$ with high precision for arbitrary small $\delta > 0$. □

For $(N, s) \to +\infty$, one has $\delta \to 0$, so that informally, the scaling is

$$P \geqslant 2s \log(2N/\varepsilon). \tag{11.7}$$

This theorem states a non-uniform recovery guarantee, in the sense that one first choose a vector x_0, then draws the matrix A, and the recovery results holds with high probability. This should be contrasted with the RIP theory developed in Section 11.3 which provides stronger uniform guarantees.

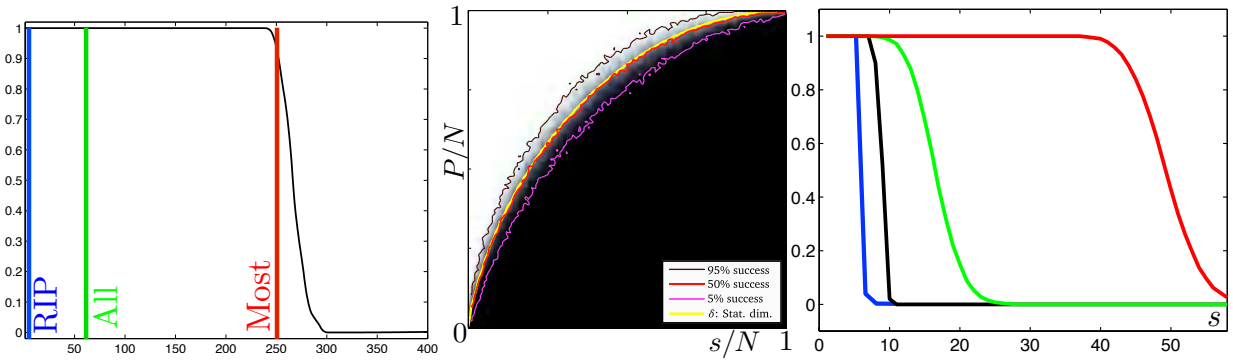


Figure 11.5: Phase transitions. For the figure on the right shows probability as function of sparsity that certain criteria hold true, blue: w-ERC, black: ERC, green $|\eta_F| \leqslant 1$, red: identifiability.

11.3 RIP Theory for Uniform Guarantees

11.3.1 Restricted Isometry Constants

The Restricted Isometry constant δ_s of a matrix $A \in \mathbb{R}^{P \times N}$ is defined as

$$\forall\, z \in \mathbb{R}^N, \quad \|z\|_0 \leqslant s \quad \Longrightarrow \quad (1-\delta_s)\|z\|^2 \leqslant \|Az\|^2 \leqslant (1+\delta_s)\|z\|^2, \tag{11.8}$$

and one usually chose the smallest δ_s so that these relation hold.

A related concept is the (s, s') restricted orthogonality constant $\theta_{s,s'}$, which is such that for all (x, x') with $\|x\|_0 \leqslant s$, $\|x'\|_0 \leqslant s'$ and disjoint support , one has

$$|\langle Ax,\, Ax' \rangle| \leqslant \theta_{s,s'} \|x\| \|x'\|$$

The following lemma shows that RI and RO constants are tightly related.

Lemma 5. *One has*

$$\theta_{s,s'} \leqslant \delta_{s+s'} \leqslant \theta_{s,s'} + \max(\delta_s, \delta_{s'}).$$

Proof. We prove the first inequality (which is the most important). We prove that if z and z' have disjoints supports and $\|z\| \leqslant s$ and $\|z'\|_0 \leqslant s$,

$$|\langle Az,\, Az' \rangle| \leqslant \delta_{2s} \|z\| \|z'\|.$$

Using the RIP (11.8) since $z \pm z'$ has support of size $s + s'$ and the fact that $\|z \pm z'\|^2 = \|z\|^2 + \|z'\|^2$, one has

$$(1-\delta_{s+s'}) \left(\|z\|^2 + \|z'\|^2\right) \leqslant \|Az \pm Az'\|^2 \leqslant (1+\delta_{s+s'}) \left(\|z\|^2 + \|z'\|^2\right).$$

One thus has using the parallelogram equality

$$|\langle Az,\, Az' \rangle| = \frac{1}{4} |\|Az + Az'\|^2 - \|Az - Az'\|^2| \leqslant \delta_{s+s'} \|z\| \|z'\|.$$

□

The following theorem states that for a sub-Gaussian random matrix, these RIP constants grow slowly with s. Let us stress that, although this is an abuse of notation, here we assume that A is a random matrix, and not a deterministic one as previously considered.

Theorem 16. *If A is a sub-Gaussian random matrix, then provided*

$$P \geqslant C \delta^{-2} s \log(eN/s) \tag{11.9}$$

it satisfies $\delta_s \leqslant \delta$ with probability $1 - 2e^{-\delta^2 \frac{m}{2C}}$, where C only depends on the sub-Gaussianity parameters appearing in (1).

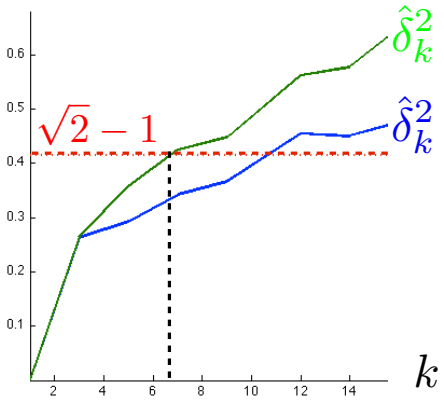


Figure 11.6: Left: evolution of lower bounds $\hat{\delta}_k$ on the RIP constant.

We do not prove this Theorem, and simply give the main intuition. The proof of this theorem relies on results regarding the distribution of the singular values of Gaussian matrices. Indeed, the RIP condition (11.8) is equivalent to having the bound $\mathrm{eig}(A_I^* A_I) \subset [1-\delta_s, 1+\delta_s]$ for all Gaussian matrices A_I extracted from A. In the Gaussian case (actually this holds for any random matrix with i.i.d. entries and unit covariance), one has a very good understanding of the distribution of the singular values of covariance matrices $B^* B \in \mathbb{R}^{s \times s}$ of a Gaussian matrix B of size (P, s), $B \sim \texttt{randn}(P, s)/\sqrt{P}$, as detailed in Section 11.2.2. In particular, using Talagrand concentration inequality (11.1), one obtains the desired controls over the δ_s constants. The intuition is that, if we assume that $s/P = \beta$ is constant and P is large, then one has that the eigenvalue of $A_I^* A_I$, and for instance its smaller one should be of the order of $2\sqrt{s/P} - s/P$, so that δ_s should be a function of s/P, and hence P should scale proportionally to the s. The log term comes from the exponential number of such matrix to control, and one need to use a non-asymptotic lower bound in place of the Marcenko-Pastur asymptotic law.

11.3.2 RIP implies dual certificates

The following theorem ensures that having small enough restricted isometry constant implies the existence of a valid dual certificates. This means that one can apply Theorem 13, which in turn ensures that one has a stable recovery of sparse signals.

Theorem 17. *If $\delta_s + \theta_{s,s} + \theta_{s,2s} < 1$, then for any x_0 with $\|x_0\| \leqslant s$, there exists $\eta \in \mathcal{D}_0(Ax_0, x_0)$, i.e. $\eta \in \mathrm{Im}(A^*) \cap \partial \|x_0\|_1$. More precisely, one has, denoting $I = \mathrm{supp}(x_0)$, $\eta_I = \mathrm{sign}(x_{0,I})$ and*

$$\|\eta_{I^c}\|_\infty \leqslant \frac{\theta_{s,s}}{1-\delta_s-\delta_{s,2s}} < 1 \quad \text{and} \quad \|p\| \leqslant \frac{\delta_{s,s}}{1-\delta_s-\theta_{s,2s}} \frac{\sqrt{s}}{\sqrt{1-\delta_s}}.$$

Note that thanks to Lemma 5, condition

$$\delta_s + \theta_{s,s} + \theta_{s,2s} \leqslant \delta_s + \delta_{2s} + \delta_{3s} \leqslant 3\delta_{3s}$$

so that condition $\delta_s + \theta_{s,s} + \theta_{s,2s} < 1$ is implied by $\delta_{3s} < 1/3$. It is furthermore possible to refine the proof of this theorem to obtain alternative (often much sharper) sufficient condition such as $\delta_{2d} \leqslant \sqrt{2}-1$. These sufficient conditions involving RI constants are often called "Restricted Isometry Properties" (RIP). As we will illustrate next, however, the constant involved on the sufficient sparsity to guarantee that such uniform RIP conditions holds are large, of the order of a few hundred.

To prove this theorem, it is not enough to directly consider the pre-certificate η_F defined in (11.2). Indeed, using such a certificate leads to slightly suboptimal log-factors in the asymptotic. The proof strategy, developed by Candes and Tao (in "Decoding with Linear Programming") consists in iteratively "improving"

this certificate by removing a vector interpolating the largest s entries outside I. In order to study η_F and to perform the improvement, it is crucial the behavior of least square solution of interpolation problems using random matrices.

Lemma 6. *We assume $\delta_s < 1$. Let $c \in \mathbb{R}^n$ with $\|c\|_0 \leqslant s$ and $\mathrm{supp}(c) = J$. Let $\eta = A^* p$ be the least square solution of $\eta_J = A_J^* p = c_J$, i.e.*

$$p \overset{\text{def.}}{=} A_J^{*,+} c_J = A_J (A_J^* A_J)^{-1} c_J$$

(note that A_J is full rank because $\delta_s < 1$). Then, denoting $K \subset J^c$ the s' largest entries in magnitude of η_{J^c}, one has

$$\|\eta_{(J\cup K)^c}\|_\infty \leqslant \frac{\theta_{s,s'}\|c\|}{(1-\delta_s)\sqrt{s}} \quad \text{and} \quad \|\eta_K\| \leqslant \frac{\theta_{s,s'}\|c\|}{1-\delta_s} \quad \text{and} \quad \|p\| \leqslant \frac{\|c\|}{\sqrt{1-\delta_s}}$$

Proof. Since $|J| \leqslant s$, one has that $\lambda_{\min}(A_J^* A_J) \geqslant 1 - \delta_s$ and hence

$$\|(A_J A_J^*)^{-1}\| \leqslant \frac{1}{1-\delta_s}$$

One has

$$\|p\|^2 = \langle A_J (A_J^* A_J)^{-1} c_J,\, A_J (A_J^* A_J)^{-1} c_J \rangle = \langle (A_J^* A_J)^{-1} c_J,\, c_j \rangle \leqslant \frac{\|c\|^2}{1-\delta_s}.$$

Let L be any set with $L \cap J = \emptyset$ and $|L| \leqslant s'$. One has

$$\|\eta_L\|^2 = |\langle \eta_L,\, \eta_L \rangle| = |\langle A_J (A_J^* A_J)^{-1} c_J,\, A_L \eta_L \rangle| \leqslant \theta_{s,s'} \|(A_J^* A_J)^{-1} c_J\| \|\eta_L\| \leqslant \frac{\theta_{s,s'}}{1-\delta_s} \|c_J\| \|\eta_L\|.$$

so that this gives

$$\|\eta_L\| \leqslant \frac{\theta_{s,s'}}{1-\delta_s} \|c_J\|. \tag{11.10}$$

Let us denote $\bar{K} = \{k \in J^c \,;\, |\eta_k| > T\}$ where $T \overset{\text{def.}}{=} \frac{\theta_{s,s'}}{(1-\delta_s)\sqrt{s'}} \|c_J\|$. One necessarily has $|\bar{K}| \leqslant s'$, otherwise one would have, taking $K \subset \bar{K}$ the s' largest entries (in fact any s' entries)

$$\|\eta_K\| > \sqrt{s' T^2} = \frac{\theta_{s,s'}}{1-\delta_s} \|c_J\|$$

which contradicts (11.10). This shows that the entries in η_{J^c} after the rank s' are smaller than T. □

We can now prove the Theorem 17.

Proof. We denote $I_0 = \mathrm{supp}(x_0)$. We first consider $\eta_1 = \eta_F$, $p_1 = p_F$, and we use (6) with $c_I = \sigma_I \overset{\text{def.}}{=} \mathrm{sign}(x_{0,I})$, $J = I$, $s' = s$, to control this first pre-certificate. This lemma defines a second set I_1 (denoted K in the lemma) which are the s largest entries of η_{1,I_1^c}, with

$$I_0 \cap I_1 = \emptyset, \quad |I_1| \leqslant s, \quad \eta_{1,I_0} = s_I, \quad \|\eta_{1,(I_0\cup I_1)^c}\|_\infty \leqslant \frac{\theta_{s,s}}{1-\delta_s}, \quad \|\eta_{I_1}\| \leqslant \frac{\theta_{s,s}\sqrt{s}}{1-\delta_s}, \quad \|p_1\| \leqslant \frac{\sqrt{s}}{\sqrt{1-\delta_s}}.$$

Now we proceed recursively. Having defined the vectors $(\eta_1 = A^* p_1, \ldots, \eta_n = A^* p_n)$ with associated sets $(I_1, \ldots, I_n)$, we define $\eta_{n+1} = A^* p_{n+1}$ and I_{n+1} by applying (6) to $J = I_0 \cup I_n$ with $c = (0_{I_0}, \eta_{n,I_n})$ (and thus $(2s, s)$ in place of (s, s')), which hence satisfy

$$I_{n+1} \cap (I_0 \cup I_n) = \emptyset, \quad |I_{n+1}| \leqslant s, \quad \eta_{n+1,I_0\cup I_n} = (0_{I_0}, \eta_{n,I_n}), \quad \text{and}$$

$$\|\eta_{n+1,(I_0\cup I_n\cup I_{n+1})^c}\|_\infty \leqslant \frac{\theta_{s,s}}{1-\delta_s} Q^n, \quad \|\eta_{n+1,I_{n+1}}\| \leqslant \frac{\theta_{s,s}\sqrt{s}}{1-\delta_s} Q^n, \tag{11.11}$$

where we denoted $Q \stackrel{\text{def.}}{=} \frac{\theta_{s,2s}}{1-\delta_s}$, and we have

$$\|p_{n+1}\| \leqslant \frac{\|\eta_{n,I_n}\|}{\sqrt{1-\delta_s}} \leqslant \frac{1}{\sqrt{1-\delta_s}} \frac{\theta_{s,s}\sqrt{s}}{1-\delta_s} Q^{n-1}.$$

Since $\delta_s + \theta_{s,s} < 1$, one has that $Q < 1$, and thus setting

$$p = \sum_{n=1}^{+\infty} (-1)^{n-1} p_n \quad \text{and} \quad \eta = A^* p$$

defines a convergent series. By construction, since $\eta_{1,I} = \sigma$ and $\eta_{n,I} = 0$ for $n > 1$, one has $\eta_I = \sigma_I$, thus this vector interpolates the sign vector σ. Now consider $j \in I^c$ and define $E_j \stackrel{\text{def.}}{=} \{n \geqslant 1 \,;\, j \in I_n\} = \{n_1 \leqslant n_2 \leqslant n_3 \leqslant \ldots\}$. Since $I_n \cap I_{n+1} = \emptyset$, necessary $n_{k+1} \geqslant n_k + 2$ (j cannot belong to two consecutive I_n). Furthermore, if $n \in E_j \Leftrightarrow j \in I_n$, then by construction

$$\eta_{n,j} = \eta_{n+1,j}$$

so that these two consecutive terms cancels out in the sum defining η **[ToDo: make a drawing]**, which in turn can thus be written in the form

$$\eta = \sum_{n \in H} (-1)^{n-1} \eta_n.$$

The index set H is composed of $n \notin E_j$ such that $n-1 \notin E_j$ (because otherwise one could cancel η_n from the sum). So this means that for $n \in H$, one has $j \notin (I_0 \cup I_n \cup I_{n+1})$, thus applying the property (11.11), one has

$$\forall\, n \in H, \quad |\eta_{j,n}| \leqslant \frac{\theta_{s,s}}{1-\delta_s} Q^{n-1},$$

so that

$$|\eta_j| \leqslant \sum_{n \in H} |\eta_{j,n}| \leqslant \sum_{n=1}^{+\infty} \frac{\theta_{s,s}}{1-\delta_s} Q^{n-1} = \frac{\theta_{s,s}}{1-\delta_s} \frac{1}{1-\theta_{s,2s}(1-\delta_s)^{-1}} = \frac{\theta_{s,s}}{1-\delta_s-\theta_{s,2s}}.$$

Note that one also has the bound

$$\|p\| \leqslant \sum_n \|p_n\| \leqslant \sum_n \frac{1}{\sqrt{1-\delta_s}} \frac{\theta_{s,s}\sqrt{s}}{1-\delta_s} Q^{n-1} = \frac{\delta_{s,s}}{1-\delta_s-\theta_{s,2s}} \frac{\sqrt{s}}{\sqrt{1-\delta_s}}.$$

□

11.3.3 RIP implies stable recovery

Putting together Theorems 16 and **??**, and using the general inverse prolem stability theorem **??**, one obtains the following recovery guarantee.

Theorem 18. *If A is a sub-Gaussian random matrix, then there exists constants (C, C') such that provided*

$$P \geqslant Cs \log(N/s) \tag{11.12}$$

with probability $1 - 2e^{-C'P}$ on the draw of A, one has that for every s-sparse signal x_0,

$$\|x_\lambda - x_0\| = O(\|w\|)$$

where x_λ is the unique solution of (9.10) with measurements $y = Ax_0 + w$ when choosing $\lambda \sim \|w\|$.

It is possible to extend this theorem when x_0 is not exactly s-sparse but only approximately. Defining $x_{0,s}$ the best s-term approximation, the easiest way to go is to write $y = Ax_{0,s} + \tilde{w}$ where $\tilde{w} = w + A(x_0 - x_{0,s})$ and applying the previous result to obtain

$$\|x_\lambda - x_{0,s}\| = O(\|w\| + \|A\|\|x_0 - x_{0,s}\|).$$

It is possible to obtain better scaling in term of the non-linear approximation error $\|x_0 - x_{0,s}\|$ by doing a more careful proof.

This theorem provides a uniform recovery guarantee, in the sense that it means

$$\mathbb{P}(\forall\, s-\text{sparse } x_0, x^\star = x_0) \text{ goes fast to 1 when } P \to +\infty.$$

In contrast, theorem 15 proves a weaker non-uniform guarantee, in the sense that it means

$$\forall\, s-\text{sparse } x_0, \mathbb{P}(x^\star = x_0) \text{ goes fast to 1 when } P \to +\infty.$$

The recovery performance analysis based on RIP constants proves a better scaling in term of log-factors. This is because the analysis using η_F does not only imply stable recovery, it also provides stability of the support (sparsistency). Note however that the constants involved in the RIP analysis are very large (of the order of a few hundreds, as highlighted by Figure 11.6, left, and by Figure 11.5, right), while the constant appearing in (11.7) are small and known to be sharp.

Also, one can show that having small enough RIP constant implies the existence of a valid dual certificate (but this certificate is not necessarily η_F).

11.3.4 Fourier sampling RIP

A practical issue is that doing hardware implementing random operators A is very difficult, specially if this operator is "fully" random, i.e. if its entries are i.i.d. A more practical option is to use structured sampling operator, which are in some sense "less random". A possibility is to consider a random sub-sampling of orthogonal projection of the signal in some ortho-basis $\Xi = (\xi_\omega)_{\omega=1}^N$ of $\mathbb{R}^N$, so that

$$Ax \overset{\text{def.}}{=} (\langle x, \xi_\omega \rangle)_{\omega \in \Omega} \in \mathbb{R}^P \tag{11.13}$$

where $\Omega \subset \{1, \dots, N\}$ is drawn uniformly at random among all sets of size P. The following theorem ensure that such an operator satisfies the RIP properties for large s (proportional to P up to log factors) is the atomes φ_ω are "spread enough", i.e. have a small magnitude, as measured by

$$\rho(\Xi) \overset{\text{def.}}{=} \sqrt{N} \max_{1 \leqslant \omega \leqslant N} \|\xi_\omega\|_\infty.$$

Theorem 19 (Rudelson-Vershinyn). *For any $0 < c < 1$, there exists C, such that provided that*

$$P \geqslant C\rho(\Xi)^2 s \log(N)^4 \tag{11.14}$$

with high probability on Ω, then A defined as in (11.13) satisfies $\delta_{2s} \leqslant c$.

One always has $1 \leqslant \rho(\Xi)^2 \leqslant N$. The worse case is ξ_ω to be Sirac atoms (i.e. $\Xi = \mathrm{Id}_N$), having $\rho(\Xi)^2 = N$, in which case P needs to be as large as N. In sharp contrast, optimal sampling bases are for instance Fourier atoms $\xi_\omega = (e^{\frac{2i\pi}{N}\omega n})_{n=1}^N \in \mathbb{C}^N$, for which $\rho(\Xi) = 1$ (it is also possible to consider a Hadamard basis for instance). In this case, up to log-factors, the scaling (11.14) is similar to the one for sub-Gaussian matrices (11.12).

Theorem 19 deals with cases where the data x_0 to recover is sparse in the Dirac basis. If the data f_0 is sparse in another basis $\Psi = (\psi_m)_{m=1}^N$, one can do a change of variable $x = \Psi^* f$ (x being the coefficients of f in the basis Ψ), in which case $\rho(\Xi)$ appearing in (11.14) should be replaced by the mutual coherence between the sampling and the sparsity bases

$$\rho(\Psi^*\Xi) \overset{\text{def.}}{=} \sqrt{N} \max_{1 \leqslant \omega, m \leqslant N} |\langle \psi_m, \xi_\omega \rangle|.$$

Good recovery performances are thus reached by (sampling,sparsity) pairs which are incoherent. The (Fourier,Dirac) pair is maximally incoherent. In contrast, Wavelet and Fourier are highly coherent. There exists explicit construction of "noiselets" bases which are almost maximally incoherent with wavelets. Note however that in contrast to Gaussian matrices, these structured measurement matrices are not universal, in the sense that there compressed sensing recovery performances depend on the sparsity basis Ψ which is used.

Chapter 12

Basics of Machine Learning

This chapter gives a rapid overview of the main concepts in machine learning. The goal is not to be exhaustive, but to highlight representative problems and insist on the distinction between unsupervised (vizualization and clustering) and supervised (regression and classification) setups. We also shed light on the tight connexions between machine learning and inverse problems.

While imaging science problems are generally concern with processing a single data (e.g. an image), machine learning problem is rather concern with analysing large collection of data. The focus (goal and performance measures) is thus radically different, but quite surprisingly, it uses very similar tools and algorithm (in particular linear models and convex optimization).

12.1 Unsupervised Learning

In unsupervised learning setups, one observes n points $(x_i)_{i=1}^n$. The problem is now to infer some properties for this points, typically for vizualization or unsupervised classication (often called clustering). For simplicity, we assume the data are points in Euclidean space $x_i \in \mathbb{R}^p$ (p is the so-called number of features). These points are conveniently stored as the rows of a matrix $X \in \mathbb{R}^{n \times d}$.

12.1.1 Dimensionality Reduction and PCA

Dimensionality reduction is useful for vizualization. It can also be understood as the problem of feature extraction (determining which are the relevant parameters) and this can be later used for doing other tasks more efficiently (faster and/or with better performances). The simplest method is the Principal Component Analysis (PCA), which performs an orthogonal linear projection on the principal axes (eigenvectors) of the covariance matrix.

Presentation of the method. The empirical mean is defined as

$$\hat{m} \stackrel{\text{def.}}{=} \frac{1}{n} \sum_{i=1}^n x_i \in \mathbb{R}^p$$

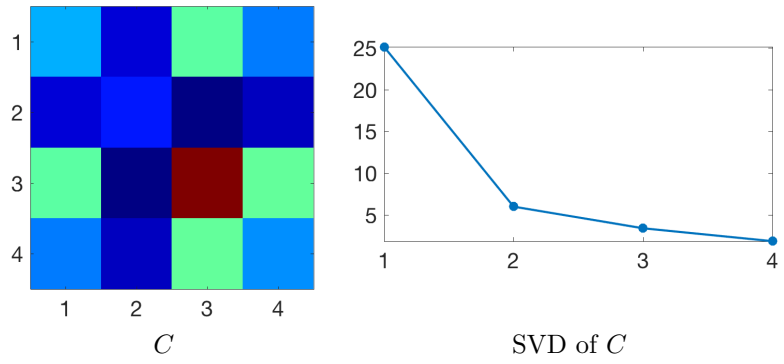
and covariance

$$\hat{C} \stackrel{\text{def.}}{=} \frac{1}{n} \sum_{i=1}^n (x_i - \hat{m})(x_i - \hat{m})^* \in \mathbb{R}^{p \times p}. \tag{12.1}$$

Denoting $\tilde{X} \stackrel{\text{def.}}{=} X - 1_p \hat{m}^*$, one has $\hat{C} = \tilde{X}^* \tilde{X}/n$.

Note that if the points $(x_i)_i$ are modelled as i.i.d. variables, and denoting $\mathbf{x}$ one of these random variables, one has, using the law of large numbers, the almost sure convergence as $n \to +\infty$

$$\hat{m} \to m \stackrel{\text{def.}}{=} \mathbb{E}(\mathbf{x}) \quad \text{and} \quad \hat{C} \to C \stackrel{\text{def.}}{=} \mathbb{E}((\mathbf{x} - m)(\mathbf{x} - m)^*). \tag{12.2}$$



C SVD of C

Figure 12.1: Empirical covariance of the data and its associated singular values.

Denoting μ the distribution (Radon measure) on $\mathbb{R}^p$ of $\mathbf{x}$, one can alternatively write

$$m = \int_{\mathbb{R}^p} x \mathrm{d}\mu(x) \quad \text{and} \quad C = \int_{\mathbb{R}^p} (x-m)(x-m)^* \mathrm{d}\mu(x).$$

The PCA ortho-basis, already introduced in Section 22, corresponds to the right singular vectors of the centred data matrix, as defined using the (reduced) SVD decomposition

$$\tilde{X} = \sqrt{n} U \operatorname{diag}(\sigma) V^*$$

where $U \in \mathbb{R}^{n \times r}$ and $V \in \mathbb{R}^{p \times r}$, and where $r = \operatorname{rank}(\tilde{X}) \leqslant \min(n,p)$. We denote $V = (v_k)_{k=1}^r$ the orthogonal columns (which forms an orthogonal system of eigenvectors of $\hat{C} = V \operatorname{diag}(\sigma^2) V^\top$), $v_k \in \mathbb{R}^p$. The intuition is that they are the main axes of "gravity" of the point cloud $(x_i)_i$ in $\mathbb{R}^p$. We assume the singular values are ordered, $\sigma_1 \geqslant \ldots \geqslant \sigma_r$, so that the first singular values capture most of the variance of the data.

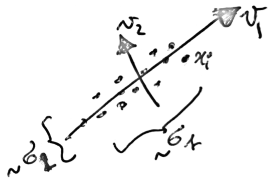


Figure 12.2: PCA main axes capture variance

Figure 12.1 displays an example of covariance and its associated spectrum σ. The points $(x_i)_i$ correspond to the celebrated IRIS dataset[1] of Fisher. This dataset consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). The dimensionality of the features is $p = 4$, and the dimensions corresponds to the length and the width of the sepals and petals.

The PCA dimensionality reduction embedding $x_i \in \mathbb{R}^p \mapsto z_i \in \mathbb{R}^d$ in dimension $d \leqslant p$ is obtained by projecting the data on the first d singular vector

$$z_i \overset{\text{def.}}{=} (\langle x_i - m, v_k \rangle)_{k=1}^d \in \mathbb{R}^d. \tag{12.3}$$

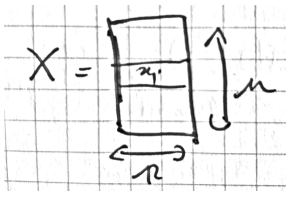
From these low-dimensional embedding, one can reconstruct back an approximation as

$$\tilde{x}_i \overset{\text{def.}}{=} m + \sum_k z_{i,k} v_k \in \mathbb{R}^p. \tag{12.4}$$

One has that $\tilde{x}_i = \operatorname{Proj}_{\tilde{T}}(x_i)$ where $\tilde{T} \overset{\text{def.}}{=} m + \operatorname{Span}_{k=1}^d(v_k)$ is an affine space.

Figure 12.3 shows an example of PCA for 2-D and 3-D vizualization.

Optimality analysis. We now show that among all possible linear dimensionality reduction method, PCA is optimal in sense of ℓ^2 error. To simplify, without loss of generality (since it can be subtracted from the data) we assume that empirical mean is zero $\hat{m} = 0$ so that $X = \tilde{X}$.



[1] https://en.wikipedia.org/wiki/Iris_flower_data_set

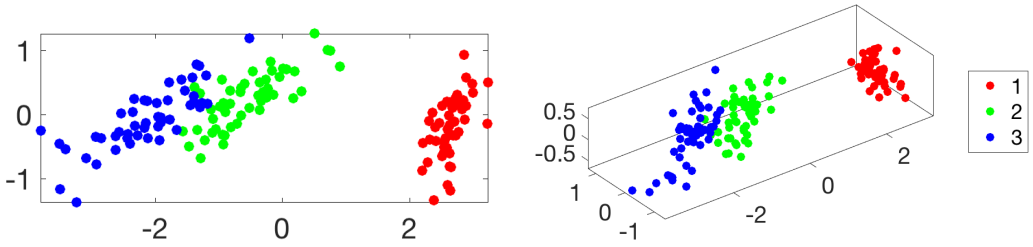


Figure 12.3: 2-D and 3-D PCA vizualization of the input clouds.

We recall that $X = \sqrt{n} U \operatorname{diag}(\sigma) V^\top$ and $\hat{C} = \frac{1}{n} X^\top X = U \Lambda U^\top$ where $\Lambda = \operatorname{diag}(\lambda_i = \sigma_i^2)$, where $\lambda_1 \geqslant \ldots \geqslant \lambda_r$.

The following proposition shows that PCA is optimal in term of ℓ^2 distance if one consider only affine spaces. This means that we consider the following compression/decompression to a dimension k (i.e. dimensionality reduction and then expansion)

$$\min_{R,S} \left\{ f(R,S) \overset{\text{def.}}{=} \sum_i \|x_i - R S^\top x_i\|_{\mathbb{R}^p}^2 \;;\; R, S \in \mathbb{R}^{p \times k} \right\} \tag{12.5}$$

Note that this minimization is a priori not trivial to solve because, although $f(\cdot, S)$ and $f(R, \cdot)$ are convex, f is not jointly convex. So iteratively minimizing on R and S might fail to converge to a global minimizer. This section aims at proving the following theorem.

Theorem 20. *A solution of (12.5) is* $S = R = V_{1:k} \overset{\text{def.}}{=} [v_1, \ldots, v_k]$.

Note that using such a compressor R and decompressor $R = S$ corresponds exactly to the PCA method (12.3) and (12.4).

We first prove that one can restrain its attention to orthogonal projection matrix.

Lemma 7. *One can assume* $S = R$ *and* $S^\top S = \mathrm{Id}_{k \times k}$.

Proof. We consider an arbitrary pair (R, S). Since the matrix $RS^\top$ has rank $k' \leqslant k$, let $W \in \mathbb{R}^{p \times k'}$ be an ortho-basis of $\operatorname{Im}(RS^\top)$, so that $W^\top W = \mathrm{Id}_{k' \times k'}$. We remark that

$$\operatorname*{argmin}_z \|x - Wz\|^2 = W^\top x$$

because the first order condition for this problem reads $W^\top(Wz - x) = 0$. Hence, denoting $RS^\top x_i = W z_i$ for some $z_i \in \mathbb{R}^{k'}$

$$f(R,S) = \sum_i \|x_i - RS^\top x_i\|^2 = \sum_i \|x_i - W z_i\|^2 \geqslant \sum_i \|x_i - WW^\top x_i\|^2 \geqslant f(\tilde{W}, \tilde{W}).$$

were we have extended W in an orthogonal matrix $\tilde{W} \in \mathbb{R}^{p \times k}$ where $\tilde{W}_{1:k'} = W$. □

Lemma 8. *Denoting* $C \overset{\text{def.}}{=} XX^\top \in \mathbb{R}^{p \times p}$, *an optimal* S *is obtained by solving*

$$\max_{S \in \mathbb{R}^{p \times k}} \left\{ \operatorname{tr}(S^\top C S^\top) \;;\; S^\top S = \mathrm{Id}_k \right\}.$$

Proof. Using the previous lemma, one can consider only $R = S$ with $S^\top S = \mathrm{Id}_k$ so that one needs to solve

$$f(S,S) = \sum_i \|x_i S S^\top x_i\|^2 = \sum_i \|x_i\|^2 - 2 x_i^\top S S^\top x_i + x_i^\top S (S^\top S) S^\top x_i.$$

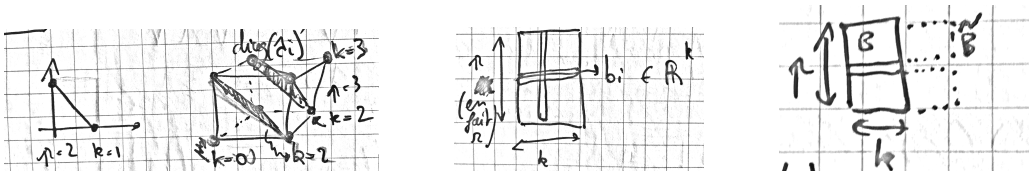


Figure 12.4: Left: proof of the rightmost inequality in (12.6). Middle: matrix B, right: matrix $\tilde{B}$.

Using that $S^\top S = \mathrm{Id}_k$, one has

$$f(S,S) = \mathrm{cst} - \sum_i x_i^\top S S^\top x_i = -\sum_i \mathrm{tr}(x_i^\top S S^\top x_i) = -\sum_i \mathrm{tr}(S^\top x_i x_i^\top S) = -\mathrm{tr}(S^\top (\sum_i x_i x_i^\top) S).$$

□

The next lemma provides an upper bound on the quantity being minimized as the solution of a convex optimization problem (a linear program). The proof of the theorem follows by showing that this upper bound is actually reached, which provides a certificate of optimality.

Lemma 9. *Denoting $C = V\Lambda V^\top$, one has*

$$\mathrm{tr}(S^\top C S) \leqslant \max_{\beta \in \mathbb{R}^p} \left\{ \sum_{i=1}^p \lambda_i \beta_i \;;\; 0 \leqslant \beta \leqslant 1, \sum_i \beta_i \leqslant k \right\} = \sum_{i=1}^k \lambda_i \tag{12.6}$$

i.e. the maximum on the right hand size is $\beta = (1, \ldots, 1, 0, \ldots, 0)$.

Proof. We extend $V \in \mathbb{R}^{p\times k}$ into an orthogonal matrix $\tilde{V} \in \mathbb{R}^{p\times p}$ (i.e. $\tilde{V}_{1:r} = V$) such that $\tilde{V}\tilde{V}^\top = \tilde{V}^\top V = \mathrm{Id}_p$. Similarly we extend Λ into $\tilde{\Lambda}$ by adding zeros, so that $C = \tilde{V}\tilde{\Lambda}\tilde{V}^\top$. One has

$$\mathrm{tr}(S^\top C S) = \mathrm{tr}(S^\top \tilde{V}\tilde{\Lambda}\tilde{V}^\top S) = \mathrm{tr}(B^\top \Lambda B) = \mathrm{tr}(\Lambda B B^\top) = \sum_{i=1}^p \lambda_i \|b_i\|^2 = \sum_i \lambda_i \beta_i$$

where we denoted $B \overset{\text{def.}}{=} V^\top S \in \mathbb{R}^{p\times k}$, $(b_i)_{i=1}^p$ with $b_i \in \mathbb{R}^k$ are the rows of B and $\beta_i \overset{\text{def.}}{=} \|b_i\|^2 \geqslant 0$. One has

$$B^\top B = S^\top \tilde{V}\tilde{V}^\top S = S^\top S = \mathrm{Id}_k,$$

so that the columns of B are orthogonal, and thus

$$\sum_i \beta_i = \sum_i \|b_i\|^2 = \|B\|_{\mathrm{Fro}}^2 = \mathrm{tr}(B^\top B) = \mathrm{tr}(B^\top B) = k.$$

We extend the k columns of b into an orthogonal basis $\tilde{B} \in \mathbb{R}^{p\times p}$ such that $\tilde{B}\tilde{B}^\top = \tilde{B}^\top \tilde{=} \mathrm{Id}_p$, so that

$$0 \leqslant \beta_i = \|b_i\|^2 \leqslant \|\tilde{b}_i\|^2 = 1$$

and hence $(\beta_i)_{i=1}^p$ satisfies the constraint of the considered optimization problem, hence $\mathrm{tr}(S^\top C S)$ is necessarily smaller than the maximum possible value.

For the proof of the second upper bound, we only verify it in 2D and 3D using a drawing, see Figure 12.4, left. □

Proof of Theorem 20. Setting $S = V_{1:k} = [v_1, \ldots, v_k]$, it satisfies $CS = V\Lambda V^\top V_{1:k} = V_{1:k}\,\mathrm{diag}(\lambda_i)_{i=1}^k$ and hence

$$\mathrm{tr}(S^\top C S) = \mathrm{tr}(S^\top S\,\mathrm{diag}(\lambda_i)_{i=1^k}) = \mathrm{tr}(\mathrm{Id}_k\,\mathrm{diag}(\lambda_i)_{i=1^k}) = \sum_{i=1}^k \lambda_i.$$

This value matches the right-most upper bound of Lemma 9, which shows that this S is optimal. □

12.1.2 Clustering and k-means

A typical unsupervised learning task is to infer a class label $y_i \in \{1, \dots, k\}$ for each input point x_i, and this is often called a clustering problem (since the set of points associated to a given label can be thought as a cluster).

k-means A way to infer these labels is by assuming that the clusters are compact, and optimizing some compactness criterion. Assuming for simplicity that the data are in Euclidean space (which can be relaxed to an arbitrary metric space, although the computations become more complicated), the k-means approach minimizes the distance between the points and their class centroids $c = (c_\ell)_{\ell=1}^k$, where each $c_\ell \in \mathbb{R}^p$. The corresponding variational problem becomes

$$\min_{(y,c)} \mathcal{E}(y,c) \stackrel{\text{def.}}{=} \sum_{\ell=1}^{k} \sum_{i:y_i=\ell} \|x_i - c_\ell\|^2.$$

The k-means algorithm can be seen as a block coordinate relaxation, which alternatively updates the class labels and the centroids. The centroids c are first initialized (more on this later), for instance, using a well-spread set of points from the samples. For a given set c of centroids, minimizing $y \mapsto \mathcal{E}(y,c)$ is obtained in closed form by assigning as class label the index of the closest centroids

$$\forall\, i \in \{1, \dots, n\}, \quad y_i \leftarrow \operatorname*{argmin}_{1 \leqslant \ell \leqslant k} \|x_i - c_\ell\|. \tag{12.7}$$

For a given set y of labels, minimizing $c \mapsto \mathcal{E}(y,c)$ is obtained in closed form by computing the barycenter of each class

$$\forall\, \ell \in \{1, \dots, k\}, \quad c_\ell \leftarrow \frac{\sum_{i:y_i=\ell} x_i}{|\{i\ ;\ y_i = \ell\}|} \tag{12.8}$$

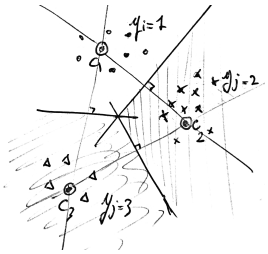


Figure 12.5: k-means clusters according to Vornoi cells.

If during the iterates, one of the cluster associated to some c_ℓ becomes empty, then one can either decide to destroy it and replace k by $k-1$, or try to "teleport" the center c_ℓ to another location (this might increase the objective function $\mathcal{E}$ however).

Since the energy $\mathcal{E}$ is decaying during each of these two steps, it is converging to some limit value. Since there is a finite number of possible labels assignments, it is actually constant after a finite number of iterations, and the algorithm stops.

Of course, since the energy is non-convex, little can be said about the property of the clusters output by k-means. To try to reach lower energy level, it is possible to "teleport" during the iterations centroids c_ℓ associated to clusters with high energy to locations within clusters with lower energy (because optimal solutions should somehow balance the energy).

Figure 12.6 shows an example of k-means iterations on the Iris dataset.

k-means++ To obtain good results when using k-means, it is crucial to have an efficient initialization scheme. In practice, the best results are obtained by seeding them as far as possible from one another (a greedy strategy works great in practice).

Quite surprisingly, there exists a randomized seeding strategy which can be shown to be close to optimal in term of value of $\mathcal{E}$, even without running the k-means iterations (although in practice it still needs to be used to polish the results). The corresponding k-means++ initialization is obtained by selecting c_1 uniformly at random among the x_i, and then assuming c_ℓ has been seeded, drawing $c_{\ell+1}$ among the sample according to the probability $\pi^{(\ell)}$ on $\{1, \dots, n\}$ proportional to the squared inverse of the distance to the previously seeded points

$$\forall\, i \in \{1, \dots, n\}, \quad \pi_i^{(\ell)} \stackrel{\text{def.}}{=} \frac{1/d_i^2}{\sum_{j=1}^n 1/d_j^2} \quad \text{where} \quad d_j \stackrel{\text{def.}}{=} \min_{1 \leqslant r \leqslant \ell-1} \|x_i - c_r\|.$$

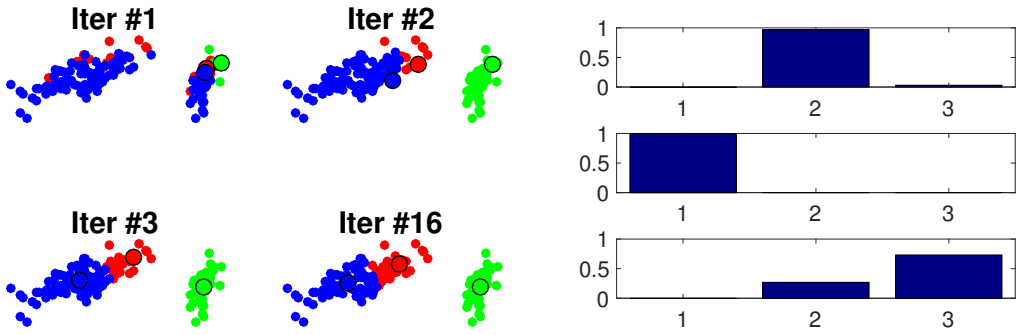


Figure 12.6: Left: iteration of k-means algorithm. Right: histogram of points belonging to each class after the k-means optimization.

This means that points which are located far away from the preciously seeded centers are more likely to be picked.

The following results, due to David Arthur and Sergei Vassilvitskii, shows that this seeding is optimal up to log factor on the energy. Note that finding a global optimum is known to be NP-hard.

Theorem 21. *For the centroids $c^\star$ defined by the k-means++ strategy, denoting $y^\star$ the associated nearest neighbor labels defined as in (12.7), one has*

$$\mathbb{E}(\mathcal{E}(y^\star, c^\star)) \leqslant 8(2+\log(k)) \min_{(y,c)} \mathcal{E}(y,v),$$

where the expectation is on the random draws performed by the algorithm.

Lloyd algorithm and continuous densities. The k-means iterations are also called "Lloyd" algorithm, which also find applications to optimal vector quantization for compression. It can also be used in the "continuous" setting where the empirical samples $(x_i)_i$ are replaced by an arbitrary measure over $\mathbb{R}^p$. The energy to minimize becomes

$$\min_{(\mathcal{V},c)} \sum_{\ell=1}^{k} \int_{\mathcal{V}_\ell} \|x-c_\ell\|^2 \mathrm{d}\mu(x)$$

where $(\mathcal{V}_\ell)_\ell$ is a partition of the domain. Step (12.7) is replaced by the computation of a Voronoi cell

$$\forall\, \ell \in \{1,\dots,k\}, \quad \mathcal{V}_\ell \overset{\text{def.}}{=} \{x \;;\; \forall\, \ell' \neq \ell, \|x-c_\ell\| \leqslant \|x-c_{\ell'}\|\}\,.$$

These Voronoi cells are polyhedra delimited by segments of mediatrix between centroids, and this Voronoi segmentation can be computed efficiently using tools from algorithmic geometry in low dimension. Step (12.8) are then replaced by

$$\forall\, \ell \in \{1,\dots,k\}, \quad c_\ell \leftarrow \frac{\int_{\mathcal{C}_\ell} x \mathrm{d}\mu(x)}{\int_{\mathcal{C}_\ell} \mathrm{d}\mu(x)}.$$

In the case of μ being uniform distribution, optimal solution corresponds to the hexagonal lattice. Figure 12.7 displays two examples of Lloyd iterations on 2-D densities on a square domain.

12.2 Empirical Risk Minimization

Before diving into the specifics of regression and classification problems, let us give describe a generic methodology which can be applied in both case (possibly with minor modification for classification, typically considering class probabilities instead of class labels).

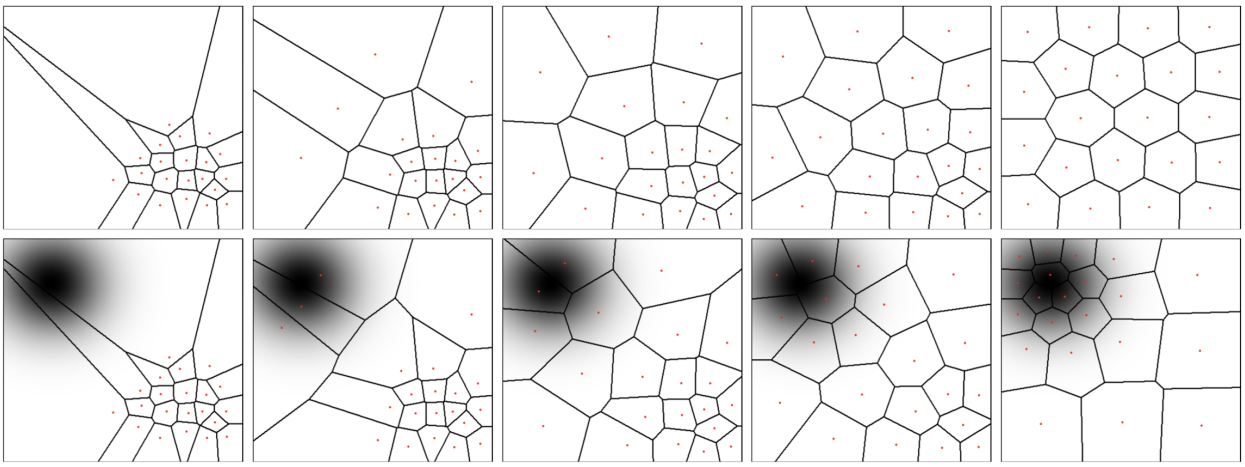


Figure 12.7: Iteration of k-means algorithm (Lloyd algorithm) on continuous densities μ. Top: uniform. Bottom: non-uniform (the densities of μ with respect to the Lebesgue measure is displayed as a grayscale image in the background).

In order to make the problem tractable computationally, and also in order to obtain efficient prediction scores, it is important to restrict the fit to the data $y_i \approx f(x_i)$ using a "small enough" class of functions. Intuitively, in order to avoid overfitting, the "size" of this class of functions should grows with the number n of samples.

12.2.1 Empirical Risk

Denoting $\mathcal{F}_n$ some class of functions (which depends on the number of available samples), one of the most usual way to do the learning is to perform an empirical risk minimization (ERM)

$$\hat{f} \in \operatorname*{argmin}_{f \in \mathcal{F}_n} \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), y_i). \tag{12.9}$$

Here $L : \mathcal{Y}^2 \to \mathbb{R}^+$ is the so-called loss function, and it should typically satisfies $L(y, y') = 0$ if and only if $y = y'$. The specifics of L depend on the application at hand (in particular, one should use different losses for classification and regression tasks). To highlight the dependency of $\hat{f}$ on n, we occasionally write $\hat{f}_n$.

12.2.2 Prediction and Consistency

When doing a mathematically analysis, one usually assumes that (x_i, y_i) are drawn from a distribution π on $\mathcal{X} \times \mathcal{Y}$, and the large n limit defines the ideal estimator

$$\bar{f} \in \operatorname*{argmin}_{f \in \mathcal{F}_\infty} \int_{\mathcal{X} \times \mathcal{Y}} L(f(x), y) \mathrm{d}\pi(x, y) = \mathbb{E}_{(\mathbf{x},\mathbf{y}) \sim \pi}(L(f(\mathbf{x}), \mathbf{y}). \tag{12.10}$$

Intuitively, one should have $\hat{f}_n \to \bar{f}$ as $n \to +\infty$, which can be captured in expectation of the prediction error over the samples $(x_i, y_i)_i$, i.e.

$$E_n \overset{\text{def.}}{=} \mathbb{E}(\tilde{L}(\hat{f}_n(\mathbf{x}), \bar{f}(\mathbf{x}))) \longrightarrow 0.$$

One should be careful that here the expectation is over both $\mathbf{x}$ (distributed according to the marginal $\pi_{\mathcal{X}}$ of π on $\mathcal{X}$), and also the n i.i.d. pairs $(x_i, y_i) \sim \pi$ used to define $\hat{f}_n$ (so a better notation should rather be

$(\mathbf{x}_i, \mathbf{y}_i)_i$. Here $\bar{L}$ is some loss function on $\mathcal{Y}$ (one can use $\bar{L} = L$ for instance). One can also study convergence in probability, i.e.

$$\forall\, \varepsilon > 0, \quad E_{\varepsilon,n} \overset{\text{def.}}{=} \mathbb{P}(\tilde{L}(\hat{f}_n(\mathbf{x}), \bar{f}(\mathbf{x})) > \varepsilon) \to 0.$$

If this holds, then one says that the estimation method is consistent (in expectation or in probability). The question is then to derive convergence rates, i.e. to upper bound E_n or $E_{\varepsilon,n}$ by some explicitly decay rate.

Note that when $\tilde{L}(y, y') = |y - y'|^r$, then convergence in expectation is stronger (implies) than convergence in probability since using Markov's inequality

$$E_{\varepsilon,n} = \mathbb{P}(|\hat{f}_n(\mathbf{x}) - f(\mathbf{x})|^r \geqslant \varepsilon) \leqslant \frac{1}{\varepsilon}\mathbb{E}(|\hat{f}_n(\mathbf{x}) - f(\mathbf{x})|^r) = \frac{E_n}{\varepsilon}.$$

12.2.3 Parametric Approaches and Regularization

Instead of directly defining the class $\mathcal{F}_n$ and using it as a constraint, it is possible to rather use a penalization using some prior to favor "simple" or "regular" functions. A typical way to achieve this is by using a parametric model $y \approx f(x, \beta)$ where $\beta \in \mathcal{B}$ parametrizes the function $f(\cdot, \beta) : \mathcal{X} \to \mathcal{Y}$. The empirical risk minimization procedure (12.9) now becomes

$$\hat{\beta} \in \operatorname*{argmin}_{\beta \in \mathcal{B}} \frac{1}{n}\sum_{i=1}^{n} L(f(x_i, \beta), y_i) + \lambda_n J(\beta). \tag{12.11}$$

where J is some regularization function, for instance $J = \|\cdot\|_2^2$ (to avoid blowing-up of the parameter) or $J = \|\cdot\|_1$ (to perform model selection, i.e. using only a sparse set of feature among a possibly very large pool of p features). Here $\lambda_n > 0$ is a regularization parameter, and it should tend to 0 when $n \to +\infty$.

Then one similarly defines the ideal parameter $\bar{\beta}$ as in (12.10) so that the limiting estimator as $n \to +\infty$ is of the form $\bar{f} = f(\cdot, \bar{\beta})$ for $\bar{\beta}$ defined as

$$\bar{\beta} \in \operatorname*{argmin}_{\beta} \int_{\mathcal{X}\times\mathcal{Y}} L(f(x, \beta), y)\mathrm{d}\pi(x, y) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\pi}(L(f(\mathbf{x}, \beta), \mathbf{y}). \tag{12.12}$$

Prediction vs. estimation risks. In this parametric approach, one could be interested in also studying how close $\hat{\theta}$ is to $\bar{\theta}$. This can be measured by controlling how fast some estimation error $\|\hat{\beta} - \bar{\beta}\|$ (for some norm $\|\cdot\|$) goes to zero. Note however that in most cases, controlling the estimation error is more difficult than doing the same for the prediction error. In general, doing a good parameter estimation implies doing a good prediction, but the converse is not true.

12.2.4 Testing Set and Cross-validation

It is not possible to access E_n or $E_{\varepsilon,n}$ because the optimal $\bar{f}$ is unknown. In order to tune some parameters of the methods (for instance the regularization parameter λ), one rather wants to minimize the risk $\mathbb{E}(L(\hat{f}(\mathbf{x}), \mathbf{y}))$, but this one should not be approximated using the training samples $(x_i, y_i)_i$.

One thus rather ressorts to a second set of data $(\bar{x}_j, \bar{y}_j)_{j=1}^{\bar{n}}$, called "testing set". From a modelling perspective, this set should also be distributed i.i.d. according to π. The validation (or testing) risk is then

$$R_{\bar{n}} = \frac{1}{\bar{n}}\sum_{j=1}^{\bar{n}} L(\hat{f}(\bar{x}_j), \bar{y}_j) \tag{12.13}$$

which converges to $\mathbb{E}(L(\hat{f}(\mathbf{x}), \mathbf{y}))$ for large $\bar{n}$. Minimizing $R_{\bar{n}}$ to setup to some meta-parameter of the method (for instance the regularization parameter λ_n) is called "cross validation" in the literature.

12.3 Supervised Learning: Regression

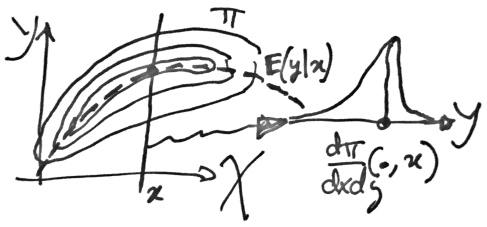


Figure 12.9: Conditional expectation.

In supervised learning, one has access to training data, consisting in pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. Here $\mathcal{X} = \mathbb{R}^p$ for simplicity. The goal is to infer some relationship, typically of the form $y_i \approx f(x_i)$ for some deterministic function $f : \mathcal{X} \to \mathcal{Y}$, in order, when some un-observed data x without associated value in $\mathcal{Y}$ is given, to be able to "predict" the associated value using $y = f(x)$.

If the set $\mathcal{Y}$ is discrete and finite, then this problem is called a supervised classification problem, and this is studied in Section 12.4. The simplest example being the binary classification case, where $\mathcal{Y} = \{0, 1\}$. It finds applications for instance in medical diagnosis, where $y_i = 0$ indicates a healthy subject, why $y_i = 0$ a pathological one. If $\mathcal{Y}$ is continuous (the typical example being $\mathcal{Y} = \mathbb{R}$), then this problem is called a regression problem.

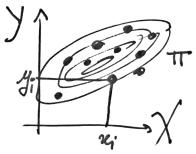


Figure 12.8: Probabilistic modelling.

12.3.1 Linear Regression

We now specialize the empirical risk minimization approach to regression problems, and even more specifically, we consider $\mathcal{Y} = \mathbb{R}$ and use a quadratic loss $L(y, y') = \frac{1}{2}|y - y'|^2$.

Note that non-linear regression can be achieved using approximation in dictionary (e.g. polynomial interpolation), and this is equivalent to using lifting to a higher dimensional space, and is also equivalent to kernelization technics studied in Section 12.5.

Least square and conditional expectation. If one do not put any constraint on f (beside being measurable), then the optimal limit estimator $\bar{f}(x)$ defined in (12.10) is simply averaging the values y sharing the same x, which is the so-called conditional expectation. Assuming for simplicity that π has some density $\frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}$ with respect to a tensor product measure $\mathrm{d}x\mathrm{d}y$ (for instance the Lebegues mesure), one has

$$\forall\, x \in \mathcal{X}, \quad \bar{f}(x) = \mathbb{E}(\mathbf{y}|\mathbf{x} = x) = \frac{\int_{\mathcal{Y}} y \frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}(x, y)\mathrm{d}y}{\int_{\mathcal{Y}} \frac{\mathrm{d}\pi}{\mathrm{d}x\mathrm{d}y}(x, y)\mathrm{d}y}$$

where $(\mathbf{x}, \mathbf{y})$ are distributed according to π.

In the simple case where $\mathcal{X}$ and $\mathcal{Y}$ are discrete, denoting $\pi_{x,y}$ the probability of $(\mathbf{x} = x, \mathbf{y} = y)$, one has

$$\forall\, x \in \mathcal{X}, \quad \bar{f}(x) = \frac{\sum_y y\pi_{x,y}}{\sum_y \pi_{x,y}}$$

and it is unspecified if the marginal of π along $\mathcal{X}$ vanishes at x.

The main issue is that this estimator $\hat{f}$ performs poorly on finite samples, and $f(x)$ is actually undefined if there is no sample x_i equal to x. This is due to the fact that the class of functions is too large, and one should impose some regularity or simplicity on the set of admissible f.

Penalized linear models. A very simple class of models is obtained by imposing that f is linear, and set $f(x,\beta) = \langle x,\, \beta\rangle$, for parameters $\beta \in \mathcal{B} = \mathbb{R}^p$. Note that one can also treat this way affine functions by remarking that $\langle x,\, \beta\rangle + \beta_0 = \langle (x,1),\, (\beta,\beta_0)\rangle$ and replacing x by $(x,1)$. So in the following, without loss of generality, we only treat the vectorial (non-affine) case.

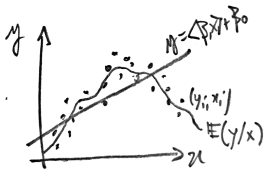


Figure 12.10: Linear regression.

Under the square loss, the regularized ERM (12.11) is conveniently rewritten as

$$\hat\beta \in \operatorname*{argmin}_{\beta\in\mathcal{B}} \frac{1}{2}\langle \hat C\beta,\, \beta\rangle - \langle \hat u,\, \beta\rangle + \lambda_n J(\beta) \tag{12.14}$$

where we introduced the empirical correlation (already introduced in (12.1)) and observations

$$\hat C \overset{\text{def.}}{=} \frac{1}{n}X^*X = \frac{1}{n}\sum_{i=1}^n x_i x_i^* \quad\text{and}\quad \hat u \overset{\text{def.}}{=} \frac{1}{n}\sum_{i=1}^n y_i x_i = \frac{1}{n}X^*y \in \mathbb{R}^p.$$

As $n\to 0$, under weak condition on π, one has with the law of large numbers the almost sure convergence

$$\hat C \to C \overset{\text{def.}}{=} \mathbb{E}(\mathbf{x}^*\mathbf{x}) \quad\text{and}\quad \hat u \to u \overset{\text{def.}}{=} \mathbb{E}(\mathbf{y}\mathbf{x}). \tag{12.15}$$

When considering $\lambda_n \to 0$, in some cases, one can shows that in the limit $n\to+\infty$, one retrieves the following ideal parameter

$$\bar\beta \in \operatorname*{argmin}_{\beta} \left\{ J(\beta)\;;\; C\beta = u \right\}.$$

Problem (12.14) is equivalent to the regularized resolution of inverse problems (8.9), with $\hat C$ in place of Φ and $\hat u$ in place of $\Phi^* y$. The major, and in fact only difference between machine learning and inverse problems is that the linear operator is also noisy since $\hat C$ can be viewed as a noisy version of C. The "noise level", in this setting, is $1/\sqrt{n}$ in the sense that

$$\mathbb{E}(\|\hat C - C\|) \sim \frac{1}{\sqrt{n}} \quad\text{and}\quad \mathbb{E}(\|\hat u - u\|) \sim \frac{1}{\sqrt{n}},$$

under the assumption that $\mathbb{E}(\mathbf{y}^4) < +\infty$, $\mathbb{E}(\|\mathbf{x}\|^4) < +\infty$ so ensure that one can use the central limit theorem on $\mathbf{x}^2$ and $\mathbf{xy}$. Note that, although we use here linear estimator, one does not need to assume a "linear" relation of the form $\mathbf{y} = \langle \mathbf{x},\, \beta\rangle + w$ with a noise w independent from $\mathbf{x}$, but rather hope to do "as best as possible", i.e. estimate a linear model as close as possible to $\bar\beta$.

The general take home message is that it is possible to generalize Theorems 10, 12 and 13 to cope with the noise on the covariance matrix to obtain prediction convergence rates of the form

$$\mathbb{E}(|\langle \hat\beta,\, \mathbf{x}\rangle - \langle \bar\beta,\, \mathbf{x}\rangle|^2) = O(n^{-\kappa})$$

and estimation rates of the form

$$\mathbb{E}(\|\hat\beta - \bar\beta\|^2) = O(n^{-\kappa'}),$$

under some suitable source condition involving C and u. Since the noise level is roughly $n^{-\frac{1}{2}}$, the ideal cases are when $\kappa = \kappa' = 1$, which is the so-called linear rate regime. It is also possible to derive sparsistency theorems by extending theorem 14. For the sake of simplicity, we now focus our attention to quadratic penalization, which is by far the most popular regression technic. It is fair to say that sparse (e.g. ℓ^1 type) methods are not routinely used in machine learning, because they typically do not improve the estimation performances, and are mostly useful to do model selection (isolate a few useful coordinates in the features). This is in sharp contrast with the situation for inverse problems in imaging sciences, where sparsity is a key feature because it corresponds to a modelling assumption on the structure of the data to recover.

Ridge regression (quadratic penalization). For $J = \|\cdot\|^2/2$, the estimator (12.14) is obtained in closed form as

$$\hat{\beta} = (X^*X + n\lambda_n \mathrm{Id}_p)^{-1} X^* y = (\hat{C} + n\lambda_n \mathrm{Id})^{-1} \hat{u}. \tag{12.16}$$

This is often called ridge regression in the literature. Note that thanks to the Woodbury formula, this estimator can also be re-written as

$$\hat{\beta} = X^*(XX^* + n\lambda_n \mathrm{Id}_n)^{-1} y. \tag{12.17}$$

If $n \gg p$ (which is the usual setup in machine learning), then (12.17) is preferable. In some cases however (in particular when using RKHS technics), it makes sense to consider very large p (even infinite dimensional), so that (12.16) must be used.

If $\lambda_n \to 0$, then using (12.15), one has the convergence in expectation and probability

$$\hat{\beta} \to \bar{\beta} = C^+ u.$$

Theorems 10 and 12 can be extended to this setting and one obtains the following result.

Theorem 22. *If*

$$\bar{\beta} = C^\gamma z \quad \textit{where} \quad \|z\| \leqslant \rho \tag{12.18}$$

for $0 < \gamma \leqslant 2$, *then*

$$\mathbb{E}(\|\hat{\beta} - \bar{\beta}\|^2) \leqslant C\rho^{2\frac{1}{\gamma+1}} n^{-\frac{\gamma}{\gamma+1}} \tag{12.19}$$

for a constant C *depending only on* γ.

It is important to note that, since $\bar{\beta} = C^+ u$, the source condition (12.18) is always satisfied. What trully matters here is that the rate (12.19) does not depend on the dimension p of the features, but rather only on ρ, which can be much smaller. This theoretical analysis actually works perfectly fine in infinite dimension $p = \infty$ (which is the setup considered when dealing with RKHS below).

12.4 Supervised Learning: Classification

We now focus on the case of discrete labels $y_i \in \mathcal{Y} = \{1, \ldots, k\}$, which is the classification setup. We now detail two popular classification methods: nearest neighbors and logistic classification. It is faire to say that a significant part of successful applications of machine learning technics consists in using one of these two approaches, which should be considered as baselines. Note that the nearest neighbors approach, while popular for classification could as well be used for regression.

12.4.1 Nearest Neighbors Classification

Probably the simplest method for supervised classification is R nearest neighbors (R-NN), where R is a parameter indexing the number of neighbors. Increasing R is important to cope with noise and obtain smoother decision boundaries, and hence better generalization performances. It should typically decreases as the number of training samples n increases. Despite its simplicity, k-NN is surprisingly successful in practice, specially in low dimension p.

The class $\hat{f}(x) \in \mathcal{Y}$ predicted for a point x is the one which is the most represented among the R points $(x_i)_i$ which are the closed to x. This is a non-parametric method, and $\hat{f}$ depends on the numbers n of samples (its "complexity" increases with n).

One first compute the Euclidean distance between this x and all other x_i in the training set. Sorting the distances generates an indexing σ (a permutation of $\{1, \ldots, n\}$) such that

$$\|x - x_{\sigma(1)}\| \leqslant \|x - x_{\sigma(2)}\| \leqslant \ldots \leqslant \|x - x_{\sigma(n)}\|.$$

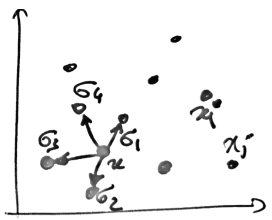
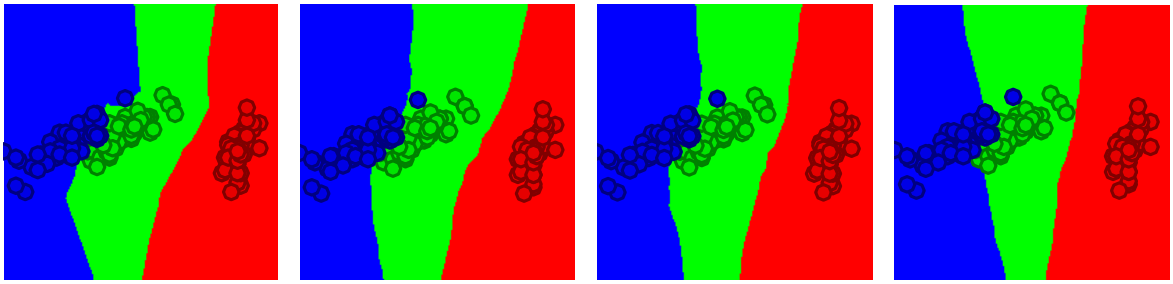


Figure 12.11: Nearest neighbors.



$k = 1$ $k = 5$ $k = 10$ $k = 40$

Figure 12.12: k-nearest-neighbor classification boundary function.

For a given R, one can compute the "local" histogram of classes around x

$$h_\ell(x) \stackrel{\text{def.}}{=} \frac{1}{R} \big\{ i \; ; \; y_{\sigma(i)} \in \{1, \ldots, R\} \big\}.$$

The decision class for x is then a maximum of the histogram

$$\hat{f}(x) \in \operatorname*{argmax}_\ell h_\ell(x).$$

In practice, the parameter R can be setup through cross-validation, by minimizing the testing risk $R_{\bar{n}}$ defined in (12.13), which typically uses a 0-1 loss for counting the number of mis-classifications

$$R_{\bar{n}} \stackrel{\text{def.}}{=} \sum_{j=1}^{\bar{n}} \delta(\bar{y}_j - \hat{f}(x_i))$$

where $\delta(0) = 0$ and $\delta(s) = 1$ if $s \neq 0$. Of course the method extends to arbitrary metric space in place of Euclidean space $\mathbb{R}^p$ for the features. Note also that instead of explicitly sorting all the Euclidean distance, one can use fast nearest neighbor search methods.

Figure 12.12 shows, for the IRIS dataset, the classification domains (i.e. $\{x \; ; \; f(x) = \ell\}$ for $\ell = 1, \ldots, k$) using a 2-D projection for vizualization. Increasing R leads to smoother class boundaries.

12.4.2 Two Classes Logistic Classification

The logistic classification method (for 2 classes and multi-classes) is one of the most popular (maybe "the" most) popular machine learning technics. This is due in large part of both its simplicity and because it also outputs a probability of belonging to each class (in place of just a class membership), which is useful to (somehow ...) quantify the "uncertainty" of the estimation. Note that logistic classification is actually called "logistic regression" in the literature, but it is in fact a classification method.

Another very popular (and very similar) approach is support vector machine (SVM). SVM is both more difficult to train (because the loss is non-smooth) and does not give class membership probability, so the general rule of thumb is that logistic classification is preferable.

To simplify the expression, classes indexes are set to $y_i \in \mathcal{Y} = \{-1, 1\}$ in the following. Note that for logistic classification, the prediction function $f(\cdot, \beta) \in [0, 1]$ outputs the probability of belonging to the first class, and not the class indexes. With a slight abuse of notation, we still denote it as f.

Approximate risk minimization. The hard classifier is defines from a linear predictor $\langle x, \beta \rangle$ as $\operatorname{sign}(\langle x, \beta \rangle) \in \{-1, +1\}$. The 0-1 loss error function (somehow the "ideal" loss) counts the number of miss-classifications, and can ideal classifier be computed as

$$\min_\beta \sum_{i=1}^{n} \ell_0(-y_i \langle x_i, \beta \rangle) \tag{12.20}$$

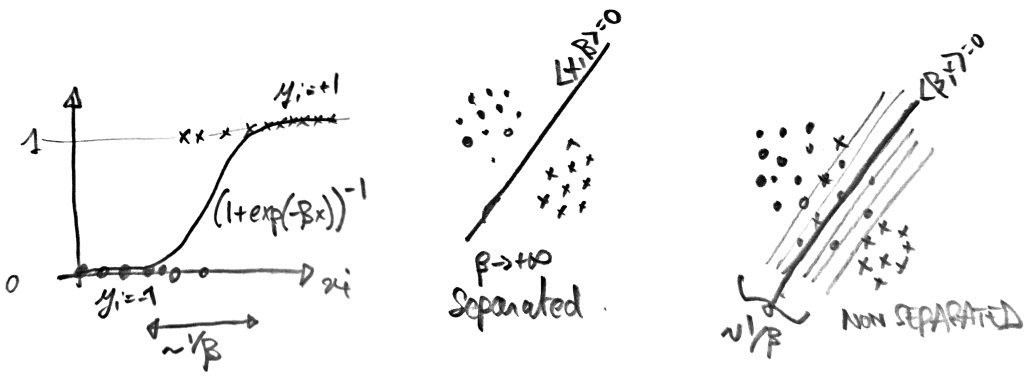


Figure 12.13: 1-D and 2-D logistic classification, showing the impact of $\|\beta\|$ on the sharpness of the classification boundary.

where $\ell_0 = 1_{\mathbb{R}^+}$. Indeed, miss classification corresponds to $\langle x_i, w\rangle$ and y_i having different signs, so that in this case $\ell_0(-y_i\langle x_i, w\rangle) = 1$ (and 0 otherwise for correct classification).

The function ℓ_0 is non-convex and hence problem (12.20) is itself non-convex, and in full generality, can be shown to be NP-hard to solve. One thus relies on some proxy, which are functions which upper-bounds ℓ_0 and are convex (and sometimes differentiable).

The most celebrated proxy are

$$\ell(u) = (1+u)_+ \quad \text{and} \quad \ell(u) = \log(1+\exp(u))/\log(2)$$

which are respectively the hinge loss corresponds to support vector machine (SVM, and is non-smooth) and the logistic loss (which is smooth). The $1/\log(2)$ is just a constant which makes $\ell_0 \leqslant \ell$. AdaBoost is using $\ell(u) = e^u$. Note that least square corresponds to using $\ell(u) = (1+u)^2$, but this is a poor proxy for ℓ_0 for negative values, although it might works well in practice. Note that SVM is a non-smooth problem, which can be cast as a linear program minimizing the co-called classification margin

$$\min_{u\geqslant 0,\beta} \left\{ \sum_i u_i \;;\; 1+u_i = y_i\langle x_i, \beta\rangle \right\}.$$

Logistic loss probabilistic interpretation. Logistic classification can be understood as a linear model as introduced in Section 12.3.1, although the decision function $f(\cdot,\beta)$ is not linear. Indeed, one needs to "remap" the linear value $\langle x, \beta\rangle$ in the interval $[0,1]$. In logistic classification, we define the predicted probability of x belonging to class with label -1 as

$$f(x,\beta) \overset{\text{def.}}{=} \theta(\langle x, \beta\rangle) \quad \text{where} \quad \theta(s) \overset{\text{def.}}{=} \frac{e^s}{1+e^s} = (1+e^{-s})^{-1}, \tag{12.21}$$

which is often called the "logit" model. Using a linear decision model might seems overly simplistic, but in high dimension p, the number of degrees of freedom is actually enough to reach surprisingly good classification performances. Note that the probability of belonging to the second class is $1 - f(x,\beta) = \theta(-s)$. This symmetry of the θ function is important because it means that both classes are treated equally, which makes sense for "balanced" problem (where the total mass of each class are roughly equal).

Intuitively, $\beta/\|\beta\|$ controls the separating hyperplane direction, while $1/\|\beta\|$ is roughly the fuzziness of the the separation. As $\|\beta\| \to +\infty$, one obtains sharp devision boundary, and logistic classification ressembles SVM.

Note that $f(x,\beta)$ can be interpreted as a single layer perceptron with a logistic (sigmoid) rectifying unit, more details on this in Chapter 15.

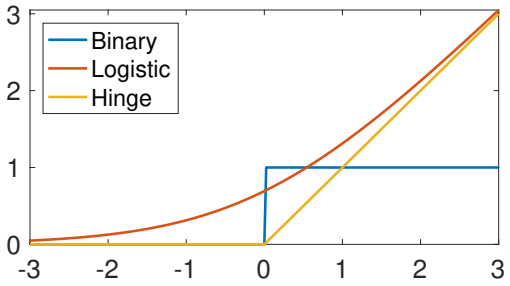


Figure 12.14: Comparison of loss functions. **[ToDo: Re-do the figure, it is not correct, they should upper bound ℓ_0]**

Since the (x_i, y_i) are modeled as i.i.d. variables, it makes sense to define $\hat{\beta}$ from the observation using a maximum likelihood, assuming that each y_i conditioned on x_i is a Bernoulli variable with associated probability $(p_i, 1-p_i)$ with $p_i = f(x_i, \beta)$. The probability of observing $y_i \in \{0,1\}$ is thus, denoting $s_i = \langle x_i, \beta \rangle$

$$\mathbb{P}(\mathbf{y} = y_i | \mathbf{x} = x_i) = p_i^{1-\bar{y}_i}(1-p_i)^{\bar{y}_i} = \left(\frac{e^{s_i}}{1+e^{s_i}}\right)^{1-\bar{y}_i}\left(\frac{1}{1+e^{s_i}}\right)^{\bar{y}_i}$$

where we denoted $\bar{y}_i = \frac{y_i+1}{2} \in \{0,1\}$.

One can then minimize minus the sum of the log of the likelihoods, which reads

$$\hat{\beta} \in \operatorname*{argmin}_{\beta \in \mathbb{R}^p} -\sum_{i=1}^n \log(\mathbb{P}(\mathbf{y} = y_i | \mathbf{x} = x_i)) = \sum_{i=1}^n -(1-\bar{y}_i)\log\frac{e^{s_i}}{1+e^{s_i}} - \bar{y}_i \log\frac{1}{1+e^{s_i}}$$

Some algebraic manipulations shows that this is equivalent to an ERM-type form (12.11) with a logistic loss function

$$\hat{\beta} \in \operatorname*{argmin}_{\beta \in \mathbb{R}^p} E(\beta) = \frac{1}{n}\sum_{i=1}^n L(\langle x_i, \beta \rangle, y_i) \tag{12.22}$$

where the logistic loss reads

$$L(s,y) \overset{\text{def.}}{=} \log(1+\exp(-sy)). \tag{12.23}$$

Problem (12.22) is a smooth convex minimization. If X is injective, E is also strictly convex, hence it has a single global minimum.

Figure (12.14) compares the binary (ideal) 0-1 loss, the logistic loss and the hinge loss (the one used for SVM).

Gradient descent method. Re-writing the energy to minimize

$$E(\beta) = \mathcal{L}(X\beta, y) \quad \text{where} \quad \mathcal{L}(s,y) = \frac{1}{n}\sum_i L(s_i, y_i),$$

its gradient reads

$$\nabla E(\beta) = X^* \nabla \mathcal{L}(X\beta, y) \quad \text{where} \quad \nabla \mathcal{L}(s,y) = \frac{y}{n} \odot \theta(-y \odot s),$$

where $\odot$ is the pointwise multiplication operator, i.e. `.*` in Matlab. Once $\beta^{(\ell=0)} \in \mathbb{R}^p$ is initialized (for instance at 0_p), one step of gradient descent (17.2) reads

$$\beta^{(\ell+1)} = \beta^{(\ell)} - \tau_\ell \nabla E(\beta^{(\ell)}).$$

To understand the behavior of the method, in Figure 12.15 we generate synthetic data distributed according to a mixture of Gaussian with an overlap governed by an offset ω. One can display the data overlaid on top of the classification probability, this highlight the separating hyperplane $\{x \; ; \; \langle \beta, x \rangle = 0\}$.

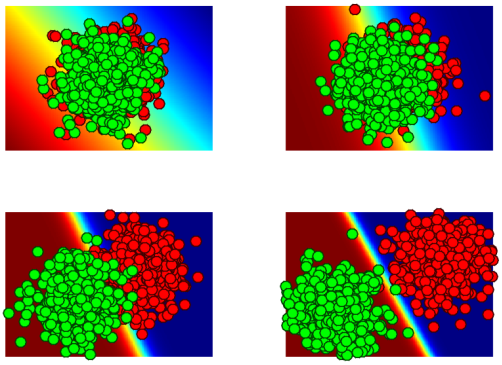


Figure 12.15: Influence on the separation distance between the class on the classification probability.

12.4.3 Multi-Classes Logistic Classification

The logistic classification method is extended to an arbitrary number k of classes by considering a family of weight vectors $\beta = (\beta_\ell)_{\ell=1}^k$, which are conveniently stored as columns of a matrix $\beta \in \mathbb{R}^{p \times k}$.

This allows one to model probabilistically the belonging of a point $x \in \mathbb{R}^p$ to the classes using the logit model

$$f(x, \beta) = \left(\frac{e^{-\langle x, \beta_\ell \rangle}}{\sum_m e^{-\langle x, \beta_m \rangle}} \right)_\ell$$

This vector $h(x) \in [0,1]^k$ describes the probability of x belonging to the different classes, and $\sum_\ell h(x)_\ell = 1$.

The computation of β is obtained by solving a maximum likelihood estimator

$$\max_{\beta \in \mathbb{R}^{p \times k}} \frac{1}{n} \sum_{i=1}^n \log(f(x_i, \beta)_{y_i})$$

where we recall that $y_i \in \mathcal{Y} = \{1, \ldots, k\}$ is the class index of the point x_i.

This is conveniently rewritten as

$$\min_{\beta \in \mathbb{R}^{p \times k}} \mathcal{E}(\beta) \stackrel{\text{def.}}{=} \sum_i \mathrm{LSE}(X\beta)_i - \langle X\beta, D \rangle$$

where $D \in \{0,1\}^{n \times k}$ is the binary class index matrices

$$D_{i,\ell} = \begin{cases} 1 \quad \text{if} \quad y_i = \ell, \\ 0 \text{otherwise.} \end{cases}$$

and LSE is the log-sum-exp operator

$$\mathrm{LSE}(S) = \log \left(\sum_\ell \exp(S_{i,\ell}) \right) \in \mathbb{R}^n.$$

Note that in the case of $k = 2$ classes $\mathcal{Y} = \{-1, 1\}$, this model can be shown to be equivalent to the two-classes logistic classifications methods exposed in Section (12.4.2), with a solution vector being equal to $\beta_1 - \beta_2$ (so it is computationally more efficient to only consider a single vector as we did).

The computation of the LSE operator is unstable for large value of $S_{i,\ell}$ (numerical overflow, producing NaN), but this can be fixed by subtracting the largest element in each row, since

$$\mathrm{LSE}(S + a) = \mathrm{LSE}(S) + a$$

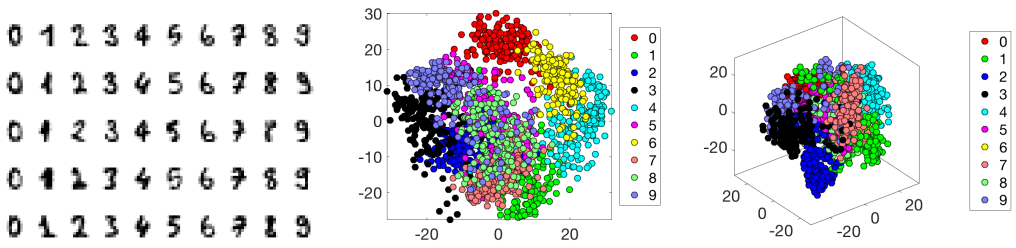


Figure 12.16: 2-D and 3-D PCA vizualization of the digits images.

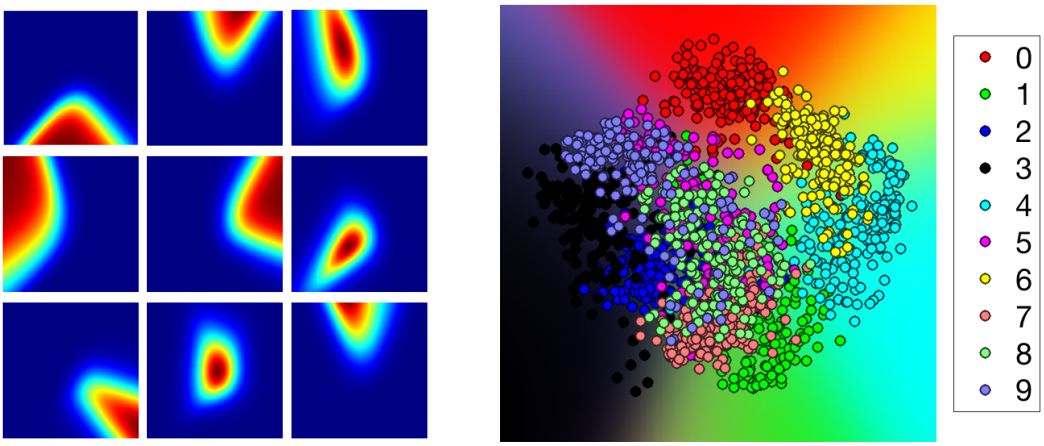


Figure 12.17: Results of digit classification Left: probability $h(x)_\ell$ of belonging to each of the 9 first classes (displayed over a 2-D PCA space). Right: colors reflect probability $h(x)$ of belonging to classes.

if a is constant along the rows. This is often referred to as the "LSE trick" and is very important to use in practice (in particular if some classes are well separated, since the corresponding β_ℓ vector might become large).

The gradient of the LSE operator is the soft-max operator

$$\nabla \mathrm{LSE}(S) = \mathrm{SM}(S) \stackrel{\text{def.}}{=} \left(\frac{e^{S_{i,\ell}}}{\sum_m e^{S_{i,m}}} \right)$$

Similarly to the LSE, it needs to be stabilized by subtracting the maximum value along rows before computation.

Once D matrix is computed, the gradient of $\mathcal{E}$ is computed as

$$\nabla \mathcal{E}(\beta) = \frac{1}{n} X^* (\mathrm{SM}(X\beta) - D).$$

and one can minimize $\mathcal{E}$ using for instance a gradient descent scheme.

To illustrate the method, we use a dataset of n images of size $p = 8 \times 8$, representing digits from 0 to 9 (so there are $k = 10$ classes). Figure 12.16 displays a few representative examples as well as 2-D and 3-D PCA projections. Figure (12.17) displays the "fuzzy" decision boundaries by vizualizing the value of $h(x)$ using colors on an image regular grid.

12.5 Kernel Methods

Linear methods are parametric and cannot generate complex regression or decision functions. The linearity assumption is often too restrictive and in some case the geometry of the input functions or classes is

not well capture by these models. In many cases (e.g. for text data) the input data is not even in a linear space, so one cannot even apply these model.

Kernel method is a simple yet surprisingly powerful remedy for these issues. By lifting the features to a high dimensional embedding space, it allows to generate non-linear decision and regression functions, but still re-use the machinery (linear system solvers or convex optimization algorithm) of linear models. Also, by the use of the so-called "kernel-trick", the computation cost does not depends on the dimension of the embedding space, but of the number n of points. It is the perfect example of so-called "non-parametric" methods, where the number of degrees of freedom (number of variables involved when fitting the model) grows with the number of samples. This is often desirable when one wants the precisions of the result to improve with n, and also to mathematically model the data using "continuous" models (e.g. functional spaces such as Sobolev).

The general rule of thumb is that any machine learning algorithm which only makes use of inner products (and not directly of the features x_i themselves) can be "kernelized" to obtain a non-parametric algorithm. This is for instance the case for linear and nearest neighbor regression, SVM classification, logistic classification and PCA dimensionality reduction. We first explain the general machinery, and instantiate this in two representative setup (ridge regression, nearest-neighbor regression and logistic classification)

12.5.1 Reproducing Kernel Hilbert Space

We consider a general lifting $\varphi : x \in \mathbb{R}^p \to \bar{x} = \varphi(x) \in \mathcal{H}$ where $\mathcal{H}$ is a Hilbert space. A typical example of lift for $1-D$ values $p = 1$ is $\varphi(x) = (1, x, x^2, \ldots, x^k) \in \mathbb{R}^k$ to perform polynomial regression (this can be extended to any dimension p using higher dimensional polynomials). We denote $\bar{X} = (\bar{x}_i^* \overset{\text{def.}}{=} \varphi(x_i)^*)_{i=1}^n$ the "matrix" where each row is a lifted feature $\varphi(x_i)$. For instance, if $\mathcal{H} = \mathbb{R}^{\bar{p}}$ is finite dimensional, one can view this as a matrix $\bar{X} \in \mathbb{R}^{n \times \bar{p}}$, but the rows of the matrix can be infinite dimensional vectors.

The following proposition is the crux of the RKHS approaches. When using a regularization which is a squared Euclidean norm, $\|\cdot\|_{\mathcal{H}}^2$, it states that the solutions actually belongs to a data-driven linear sub-space of dimension n. Although the proof is straightforward, its implications are very profound, since it leads to tractable algorithms even when using an infinite dimensional lifting space $\mathcal{H}$. as we elaborate next. It is often called the "representer" theorem in RKHS theory.

Proposition 37. *The solution $\beta^\star \in \mathcal{H}$ of*

$$\min_{\beta \in \mathcal{H}} \mathcal{L}(\bar{X}\beta, y) + \frac{\lambda}{2}\|\beta\|_{\mathcal{H}}^2 \tag{12.24}$$

is unique and can be written as

$$\beta = \bar{X}^* q^\star = \sum_i q_i^\star \varphi(x_i) \in \mathcal{H} \tag{12.25}$$

where $q \in \mathbb{R}^N$ is a solution of

$$\min_{q \in \mathbb{R}^N} \mathcal{L}(Kp, y) + \frac{\lambda}{2}\langle Kq,\, q\rangle_{\mathbb{R}^n} \tag{12.26}$$

where we defined

$$K \overset{\text{def.}}{=} \bar{X}^*\bar{X} = (\langle \varphi(x_i),\, \varphi(x_j)\rangle_{\mathcal{H}})_{i,j=1}^n \in \mathbb{R}^{n \times n}.$$

Proof. The first order condition of (12.24) reads

$$0 \in \bar{X}^* \partial \mathcal{L}(\bar{X}^*\beta^\star, y) + \lambda\beta^\star = 0$$

i.e. there exists $u^\star \in \partial\mathcal{L}(\bar{X}^*\beta^\star, y)$ such that

$$\beta^\star = -\frac{1}{\lambda}\bar{X}^* u^\star \in \mathrm{Im}(\bar{X}^*)$$

which is the desired result. □

Equation (12.25) expresses the fact that the solution only lives in the n dimensional space spanned by the lifted observed points $\varphi(x_i)$. A crucial by product of this results is that all the computations as well as the prediction procedure can be expressed using the so-called kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ associated to φ

$$\forall\, (x, x') \in \mathcal{X}^2, \quad \kappa(x, x') \overset{\text{def.}}{=} \langle \varphi(x'), \varphi(x') \rangle_{\mathcal{H}}.$$

Indeed, one has $K = (\kappa(x_i, x_j))_{i,j}$ and the prediction operator, as a function of x and not $\varphi(x)$ (which makes it non-linear) is a weighted sum of kernel functions centered at the x_i

$$\langle \bar{x},\, \beta^\star \rangle_{\mathcal{H}} = \sum_{i=1}^{n} p_i^\star \langle \varphi(x),\, \varphi(x_i) \rangle_{\mathcal{H}} = \sum_{i=1}^{n} p_i^\star \kappa(x_i, x). \tag{12.27}$$

This means that one actually never needs to manipulate quantities in $\mathcal{H}$ (which can be infinite dimensional).

But more importantly, one can reverse the process, and instead of starting from a lifting φ, directly consider a kernel $\kappa(x, x')$. This is actually the way this is done in practice, since it is easier to design kernel and think in term of their geometrical properties (for instance, one can sum kernels). In order for this to make sense, the kernel needs to be positive definite, i.e. one should have that $(\kappa(x_i, x_j))_{i,j}$ should be symmetric positive definite for any choice of sampling points $(x_i)_i$. This can be shown to be equivalent to the existence of a lifting function φ generating the kernel. Note that such a kernel can be defined on arbitrary space (not necessarily Euclidean).

When using the linear kernel $\kappa(x, y) = \langle x,\, y \rangle$, one retrieves the linear models studied in the previous section, and the lifting is trivial $\varphi(x) = x$. A family of popular kernels are polynomial ones, $\kappa(x, x') = (\langle x,\, y \rangle + c)^a$ for $a \in \mathbb{N}^*$ and $c > 0$, which corresponds to a lifting in finite dimension. For instance, for $a = 2$ and $p = 2$, one has a lifting in dimension 6

$$\kappa(x, x') = (x_1 x_1' + x_1 x_1' + c)^2 = \langle \varphi(x),\, \varphi(x') \rangle \quad \text{where} \quad \varphi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2, \sqrt{2c} x_1, \sqrt{2c} x_2, c)^* \in \mathbb{R}^6.$$

In Euclidean spaces, the gaussian kernel is the most well known and used kernel

$$\kappa(x, y) \overset{\text{def.}}{=} e^{-\frac{\|x-y\|^2}{2\sigma^2}}. \tag{12.28}$$

The bandwidth parameter $\sigma > 0$ is crucial and controls the locality of the model. It is typically tuned through cross validation. It corresponds to an infinite dimensional lifting $x \mapsto e^{-\frac{\|x-\cdot\|^2}{2(\sigma/2)^2}} \in L^2(\mathbb{R}^p)$. Another related popular kernel is the Laplacian kernel $\exp(-\|x-y\|/\sigma)$. More generally, when considering translation invariant kernels $\kappa(x, x') = k(x - x')$ on $\mathbb{R}^p$, being positive definite is equivalent to $\hat{k}(\omega) > 0$ where $\hat{k}$ is the Fourier transform, and the associated lifting is obtained by considering $\hat{h} = \sqrt{\hat{\kappa}}$ and $\varphi(x) = h(x - \cdot) \in L^2(\mathbb{R}^p)$.

12.5.2 Examples of Kernelized Algorithms

We illustrate this general machinery by applying it to three typical problems.

Kernelized ridge regression. The simplest instantiation of this kernelization approach is when using the square loss $L(y, y') = \frac{1}{2}|y - y'|^2$, which is the ridge regression problem studied in Section 12.3.1. The obtain regression model (12.27) corresponds to approximating the data using a weighted sum of data-centered kernel function $\kappa(x_i, \cdot)$. When using a Gaussian kernel (12.28), the bandwidth σ controls the smoothness of the approximation. This is illustrated in Figure 12.18.

In this special case of a square loss, one can solve in closed form (12.26) by solving a $n \times n$ linear system

$$q^\star = (KK + \lambda K)^{-1} K y = (K + \lambda \mathrm{Id}_N)^{-1} y$$

This expression matches exactly (12.17) when using K in place of $\hat{C}$

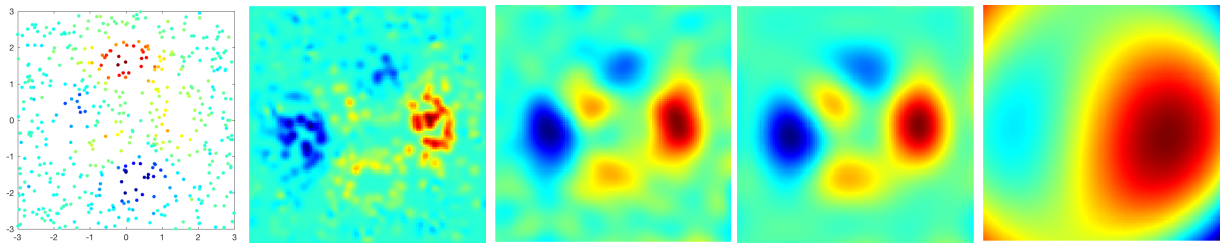


Figure 12.18: Regression using a Gaussian kernel.

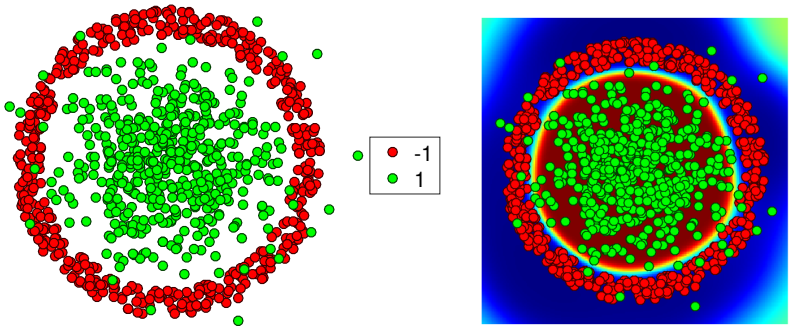


Figure 12.19: Non-linear classification using a Gaussian kernel.

Kernelized logistic classification. Logistic classification tries to separate the classes using a linear separating hyperplane $\{x\ ;\ \langle \beta,\ x\rangle = 0\}$. In order to generate a non-linear decision boundary, one can replace the parametric linear model by a non-linear non-parametric model, thanks to kernelization. This allows in particular to generate decision boundaries of arbitrary complexity.

In the two class problem, as detailed in Section 12.4.2, one solves (12.26) using the logistic loss (12.23). This can be for instance achieved by a gradient descent method. Once the solution $q^\star$ is obtained, the probability of x belonging to the first class is then

$$\theta(\sum_{i=1}^{n} q_i^\star \kappa(x_i, x)).$$

Figure 12.19 illustrate such a non-linear decision function on a simple 2-D problem.

Kernelized nearest-neihbors. It is also possible to extend nearest neighbor classification (as detailed in Section 12.4.1) and regression over a lifted space by making use only of kernel evaluation, simply noticing that

$$\|\varphi(x_i) - \varphi(x_j)\|_{\mathcal{H}}^2 = \kappa(x_i, x_i) + \kappa(x_j, x_j) - 2\kappa(x_i, x_j).$$

Kernel on strings. **[ToDo: write me]**

Chapter 13

Optimization & Machine Learning: Smooth Optimization

13.1 Motivation in Machine Learning

13.1.1 Unconstraint optimization

In most part of this Chapter, we consider unconstrained convex optimization problems of the form

$$\inf_{x \in \mathbb{R}^p} f(x), \tag{13.1}$$

and try to devise "cheap" algorithms with a low computational cost per iteration to approximate a minimizer when it exists. The class of algorithms considered are first order, i.e. they make use of gradient information. In the following, we denote

$$\operatorname*{argmin}_x f(x) \stackrel{\text{def.}}{=} \{x \in \mathbb{R}^p \,;\, f(x) = \inf f\},$$

to indicate the set of points (it is not necessarily a singleton since the minimizer might be non-unique) that achieve the minimum of the function f. One might have $\operatorname{argmin} f = \emptyset$ (this situation is discussed below), but in case a minimizer exists, we denote the optimization problem as

$$\min_{x \in \mathbb{R}^p} f(x). \tag{13.2}$$

In typical learning scenario, $f(x)$ is the empirical risk for regression or classification, and p is the number of parameter. For instance, in the simplest case of linear models, we denote $(a_i, y_i)_{i=1}^n$ where $a_i \in \mathbb{R}^p$ are the features. In the following, we denote $A \in \mathbb{R}^{n \times p}$ the matrix whose rows are the a_i.

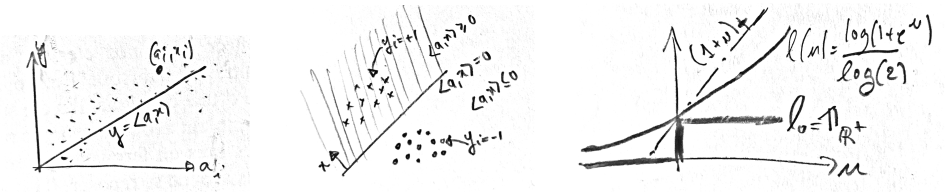


Figure 13.1: Left: linear regression, middle: linear classifier, right: loss function for classification.

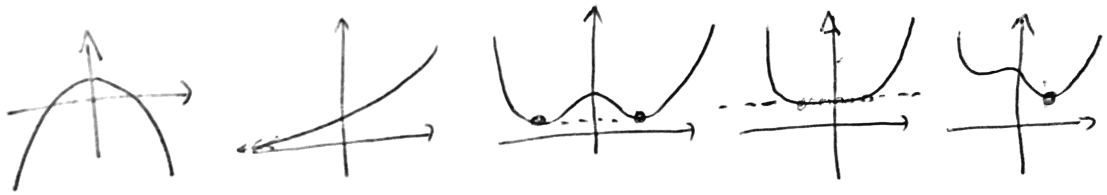


Figure 13.2: Left: non-existence of minimizer, middle: multiple minimizers, right: uniqueness.

13.1.2 Regression

For regression, $y_i \in \mathbb{R}$, in which case

$$f(x) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \langle x, a_i\rangle)^2 = \frac{1}{2}\|Ax - y\|^2, \tag{13.3}$$

is the least square quadratic risk function (see Fig. 13.1). Here $\langle u, v\rangle = \sum_{i=1}^{p} u_i v_i$ is the canonical inner product in $\mathbb{R}^p$ and $\|\cdot\|^2 = \langle \cdot, \cdot\rangle$.

13.1.3 Classification

For classification, $y_i \in \{-1, 1\}$, in which case

$$f(x) = \sum_{i=1}^{n} \ell(-y_i\langle x, a_i\rangle) = L(-\operatorname{diag}(y)Ax) \tag{13.4}$$

where ℓ is a smooth approximation of the 0-1 loss $1_{\mathbb{R}^+}$. For instance $\ell(u) = \log(1 + \exp(u))$, and $\operatorname{diag}(y) \in \mathbb{R}^{n\times n}$ is the diagonal matrix with y_i along the diagonal (see Fig. 13.1, right). Here the separable loss function $L = \mathbb{R}^n \to \mathbb{R}$ is, for $z \in \mathbb{R}^n$, $L(z) = \sum_i \ell(z_i)$.

13.2 Basics of Convex Analysis

13.2.1 Existence of Solutions

In general, there might be no solution to the optimization (17.1). This is of course the case if f is unbounded by below, for instance $f(x) = -x^2$ in which case the value of the minimum is $-\infty$. But this might also happen if f does not grow at infinity, for instance $f(x) = e^{-x}$, for which $\min f = 0$ but there is no minimizer.

In order to show existence of a minimizer, and that the set of minimizer is bounded (otherwise one can have problems with optimization algorithm that could escape to infinity), one needs to show that one can replace the whole space $\mathbb{R}^p$ by a compact sub-set $\Omega \subset \mathbb{R}^p$ (i.e. Ω is bounded and close) and that f is continuous on Ω (one can replace this by a weaker condition, that f is lower-semi-continuous, but we ignore this here). A way to show that one can consider only a bounded set is to show that $f(x) \to +\infty$ when $x \to +\infty$. Such a function is called coercive. In this case, one can choose any $x_0 \in \mathbb{R}^p$ and consider its associated lower-level set

$$\Omega = \{x \in \mathbb{R}^p \;;\; f(x) \leqslant f(x_0)\}$$

which is bounded because of coercivity, and closed because f is continuous. One can actually show that for convex function, having a bounded set of minimizer is equivalent to the function being coercive (this is not the case for non-convex function, for instance $f(x) = \min(1, x^2)$ has a single minimum but is not coercive).

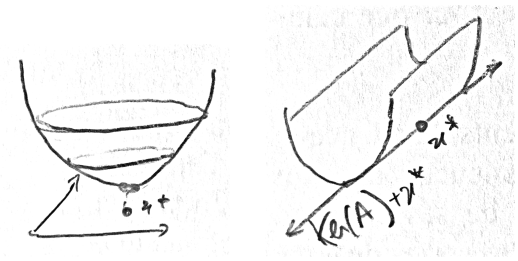


Figure 13.3: Coercivity condition for least squares.

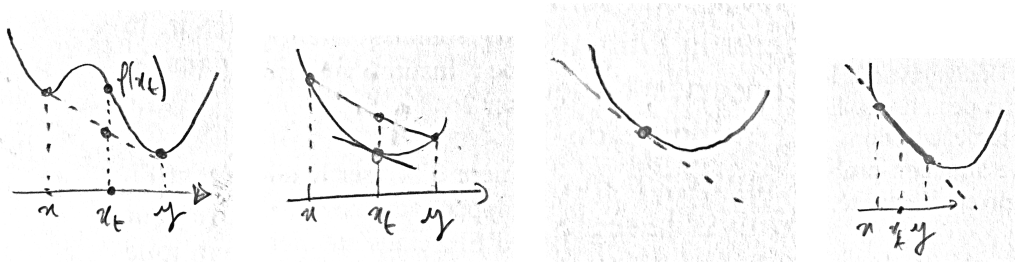


Figure 13.4: Convex vs. non-convex functions ; Strictly convex vs. non strictly convex functions.

Example 1 (Least squares). For instance, for the quadratic loss function $f(x) = \frac{1}{2}\|Ax - y\|^2$, coercivity holds if and only if $\ker(A) = \{0\}$ (this corresponds to the overdetermined setting). Indeed, if $\ker(A) \neq \{0\}$ if $x^\star$ is a solution, then $x^\star + u$ is also solution for any $u \in \ker(A)$, so that the set of minimizer is unbounded. On contrary, if $\ker(A) = \{0\}$, we will show later that the set of minimizer is unique, see Fig. 13.3. If ℓ is strictly convex, the same conclusion holds in the case of classification.

13.2.2 Convexity

Convex functions define the main class of functions which are somehow "simple" to optimize, in the sense that all minimizers are global minimizers, and that there are often efficient methods to find these minimizers (at least for smooth convex functions). A convex function is such that for any pair of point $(x, y) \in (\mathbb{R}^p)^2$,

$$\forall\, t \in [0,1], \quad f((1-t)x + ty) \leqslant (1-t)f(x) + tf(y) \tag{13.5}$$

which means that the function is below its secant (and actually also above its tangent when this is well defined), see Fig. 13.4. If $x^\star$ is a local minimizer of a convex f, then $x^\star$ is a global minimizer, i.e. $x^\star \in \operatorname{argmin} f$.

Convex function are very convenient because they are stable under lots of transformation. In particular, if f, g are convex and a, b are positive, $af + bg$ is convex (the set of convex function is itself an infinite dimensional convex cone!) and so is $\max(f, g)$. If $g : \mathbb{R}^q \to \mathbb{R}$ is convex and $B \in \mathbb{R}^{q\times p}, b \in \mathbb{R}^q$ then $f(x) = g(Bx+b)$ is convex. This shows immediately that the square loss appearing in (13.3) is convex, since $\|\cdot\|^2/2$ is convex (as a sum of squares). Also, similarly, if ℓ and hence L is convex, then the classification loss function (13.4) is itself convex.

Strict convexity. When f is convex, one can strengthen the condition (13.5) and impose that the inequality is strict for $t \in]0,1[$ (see Fig. 13.4, right), i.e.

$$\forall\, t \in]0,1[, \quad f((1-t)x + ty) < (1-t)f(x) + tf(y). \tag{13.6}$$

In this case, if a minimum $x^\star$ exists, then it is unique. Indeed, if $x_1^\star \neq x_2^\star$ were two different minimizer, one would have by strict convexity $f(\frac{x_1^\star + x_2^\star}{2}) < f(x_1^\star)$ which is impossible.

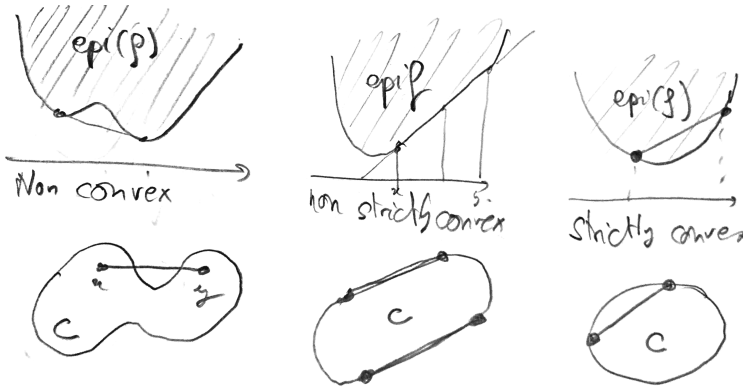


Figure 13.5: Comparison of convex functions $f : \mathbb{R}^p \to \mathbb{R}$ (for $p = 1$) and convex sets $C \subset \mathbb{R}^p$ (for $p = 2$).

Example 2 (Least squares). For the quadratic loss function $f(x) = \frac{1}{2}\|Ax - y\|^2$, strict convexity is equivalent to $\ker(A) = \{0\}$. Indeed, we see later that its second derivative is $\partial^2 f(x) = A^\top A$ and that strict convexity is implied by the eigenvalues of $A^\top A$ being strictly positive. The eigenvalues of $A^\top A$ being positive, it is equivalent to $\ker(A^\top A) = \{0\}$ (no vanishing eigenvalue), and $A^\top Az = 0$ implies $\langle A^\top Az,\, z\rangle = \|Az\|^2 = 0$ i.e. $z \in \ker(A)$.

13.2.3 Convex Sets

A set $\Omega \subset \mathbb{R}^p$ is said to be convex if for any $(x, y) \in \Omega^2$, $(1-t)x + ty \in \Omega$ for $t \in [0, 1]$. The connexion between convex function and convex sets is that a function f is convex if and only if its epigraph $\mathrm{epi}(f) \overset{\text{def.}}{=} \big\{(x, t) \in \mathbb{R}^{p+1} \;;\; t \geqslant f(x)\big\}$ is a convex set.

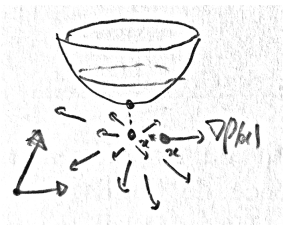
Remark 2 (Convexity of the set of minimizers). In general, minimizers $x^\star$ might be non-unique, as shown on Figure 13.3. When f is convex, the set $\mathrm{argmin}(f)$ of minimizers is itself a convex set. Indeed, if $x_1^\star$ and $x_2^\star$ are minimizers, so that in particular $f(x_1^\star) = f(x_2^\star) = \min(f)$, then $f((1-t)x_1^\star + tx_2^\star) \leqslant (1-t)f(x_1^\star) + tf(x_2^\star) = f(x_1^\star) = \min(f)$, so that $(1-t)x_1^\star + tx_2^\star$ is itself a minimizer. Figure 13.5 shows convex and non-convex sets.

13.3 Derivative and gradient

13.3.1 Gradient

If f is differentiable along each axis, we denote

$$\nabla f(x) \overset{\text{def.}}{=} \left(\frac{\partial f(x)}{\partial x_1}, \ldots, \frac{\partial f(x)}{\partial x_p}\right)^\top \in \mathbb{R}^p$$



the gradient vector, so that $\nabla f : \mathbb{R}^p \to \mathbb{R}^p$ is a vector field. Here the partial derivative (when they exits) are defined as

$$\frac{\partial f(x)}{\partial x_k} \overset{\text{def.}}{=} \lim_{\eta \to 0} \frac{f(x + \eta\delta_k) - f(x)}{\eta}$$

where $\delta_k = (0, \ldots, 0, 1, 0, \ldots, 0)^\top \in \mathbb{R}^p$ is the k^{th} canonical basis vector.

Beware that $\nabla f(x)$ can exist without f being differentiable. Differentiability of f at each reads

$$f(x + \varepsilon) = f(x) + \langle \varepsilon,\, \nabla f(x)\rangle + o(\|\varepsilon\|). \tag{13.7}$$

Here $R(\varepsilon) = o(\|\varepsilon\|)$ denotes a quantity which decays faster than ε toward 0, i.e. $\frac{R(\varepsilon)}{\|\varepsilon\|} \to 0$ as $\varepsilon \to 0$. Existence of partial derivative corresponds to f being differentiable along the axes, while differentiability should hold

for any converging sequence of $\varepsilon \to 0$ (i.e. not along along a fixed direction). A counter example in 2-D is $f(x) = \frac{2x_1x_2(x_1+x_2)}{x_1^2+x_2^2}$ with $f(0) = 0$, which is affine with different slope along each radial lines.

Also, $\nabla f(x)$ is the only vector such that the relation (13.7). This means that a possible strategy to both prove that f is differentiable and to obtain a formula for $\nabla f(x)$ is to show a relation of the form

$$f(x+\varepsilon) = f(x) + \langle \varepsilon,\, g\rangle + o(\|\varepsilon\|),$$

in which case one necessarily has $\nabla f(x) = g$.

The following proposition shows that convexity is equivalent to the graph of the function being above its tangents.

Proposition 38. *If f is differentiable, then*

$$f \text{ convex} \quad \Leftrightarrow \quad \forall (x,x'),\ f(x) \geqslant f(x') + \langle \nabla f(x'),\, x - x'\rangle.$$

Proof. One can write the convexity condition as

$$f((1-t)x + tx') \leqslant (1-t)f(x) + tf(x') \quad \Longrightarrow \quad \frac{f(x+t(x'-x)) - f(x)}{t} \leqslant f(x') - f(x)$$

hence, taking the limit $t \to 0$ one obtains

$$\langle \nabla f(x),\, x' - x\rangle \leqslant f(x') - f(x).$$

For the other implication, we apply the right condition replacing (x, x') by $(x, x_t \overset{\text{def.}}{=} (1-t)x + tx')$ and $(x', (1-t)x + tx')$

$$\begin{aligned} f(x) &\geqslant f(x_t) + \langle \nabla f(x_t),\, x - x_t\rangle = f(x_t) - t\langle \nabla f(x_t),\, x - x'\rangle \\ f(x') &\geqslant f(x_t) + \langle \nabla f(x_t),\, x' - x_t\rangle = f(x_t) + (1-t)\langle \nabla f(x_t),\, x - x'\rangle, \end{aligned}$$

multiplying these inequality by respectively $1-t$ and t, and summing them, gives

$$(1-t)f(x) + tf(x') \geqslant f(x_t).$$

□

13.3.2 First Order Conditions

The main theoretical interest (we will see later that it also have algorithmic interest) of the gradient vector is that it is a necessarily condition for optimality, as stated below.

Proposition 39. *If $x^\star$ is a local minimum of the function f (i.e. that $f(x^\star) \leqslant f(x)$ for all x in some ball around $x^\star$) then*

$$\nabla f(x^\star) = 0.$$

Proof. One has for ε small enough and u fixed

$$f(x^\star) \leqslant f(x^\star + \varepsilon u) = f(x^\star) + \varepsilon\langle \nabla f(x^\star),\, u\rangle + o(\varepsilon) \quad \Longrightarrow \quad \langle \nabla f(x^\star),\, u\rangle \geqslant o(1) \quad \Longrightarrow \quad \langle \nabla f(x^\star),\, u\rangle \geqslant 0.$$

So applying this for u and $-u$ in the previous equation shows that $\langle \nabla f(x^\star),\, u\rangle = 0$ for all u, and hence $\nabla f(x^\star) = 0$. □

Note that the converse is not true in general, since one might have $\nabla f(x) = 0$ but x is not a local mininimum. For instance $x = 0$ for $f(x) = -x^2$ (here x is a maximizer) or $f(x) = x^3$ (here x is neither a maximizer or a minimizer, it is a saddle point), see Fig. 13.6. Note however that in practice, if $\nabla f(x^\star) = 0$ but x is not a local minimum, then $x^\star$ tends to be an unstable equilibrium. Thus most often a gradient-based algorithm will converge to points with $\nabla f(x^\star) = 0$ that are local minimizers. The following proposition shows that a much strong result holds if f is convex.

Figure 13.6: Function with local maxima/minima (left), saddle point (middle) and global minimum (right).

Proposition 40. *If f is convex and $x^\star$ a local minimum, then $x^\star$ is also a global minimum. If f is differentiable and convex,*

$$x^\star \in \operatorname*{argmin}_x f(x) \quad \Longleftrightarrow \quad \nabla f(x^\star) = 0.$$

Proof. For any x, there exist $0 < t < 1$ small enough such that $tx + (1-t)x^\star$ is close enough to $x^\star$, and so since it is a local minimizer

$$f(x^\star) \leqslant f(tx + (1-t)x^\star) \leqslant tf(x) + (1-t)f(x^\star) \quad \Longrightarrow \quad f(x^\star) \leqslant f(x)$$

and thus $x^\star$ is a global minimum.

For the second part, we already saw in (39) the $\Leftarrow$ part. We assume that $\nabla f(x^\star) = 0$. Since the graph of x is above its tangent by convexity (as stated in Proposition 38),

$$f(x) \geqslant f(x^\star) + \langle \nabla f(x^\star),\, x - x^\star \rangle = f(x^\star).$$

□

Thus in this case, optimizing a function is the same a solving an equation $\nabla f(x) = 0$ (actually p equations in p unknown). In most case it is impossible to solve this equation, but it often provides interesting information about solutions $x^\star$.

13.3.3 Least Squares

The most important gradient formula is the one of the square loss (13.3), which can be obtained by expanding the norm

$$\begin{aligned} f(x+\varepsilon) &= \frac{1}{2}\|Ax - y + A\varepsilon\|^2 = \frac{1}{2}\|Ax-y\| + \langle Ax - y,\, A\varepsilon\rangle + \frac{1}{2}\|A\varepsilon\|^2 \\ &= f(x) + \langle \varepsilon,\, A^\top(Ax-y)\rangle + o(\|\varepsilon\|). \end{aligned}$$

Here, we have used the fact that $\|A\varepsilon\|^2 = o(\|\varepsilon\|)$ and use the transpose matrix $A^\top$. This matrix is obtained by exchanging the rows and the columns, i.e. $A^\top = (A_{j,i})_{i=1,\dots,n}^{j=1,\dots,p}$, but the way it should be remember and used is that it obeys the following swapping rule of the inner product,

$$\forall\, (u,v) \in \mathbb{R}^p \times \mathbb{R}^n, \quad \langle Au,\, v\rangle_{\mathbb{R}^n} = \langle u,\, A^\top v\rangle_{\mathbb{R}^p}.$$

Computing gradient for function involving linear operator will necessarily requires such a transposition step. This computation shows that

$$\nabla f(x) = A^\top(Ax - y). \tag{13.8}$$

This implies that solutions $x^\star$ minimizing $f(x)$ satisfies the linear system $(A^\top A)x^\star = A^\top y$. If $A^\star A \in \mathbb{R}^{p\times p}$ is invertible, then f has a single minimizer, namely

$$x^\star = (A^\top A)^{-1} A^\top y. \tag{13.9}$$

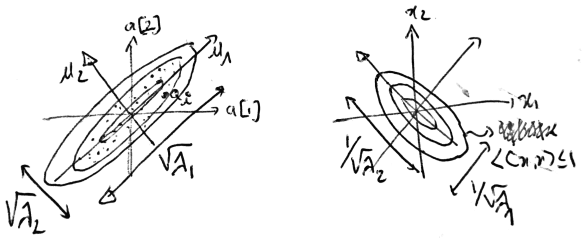


Figure 13.7: Left: point clouds $(a_i)_i$ with associated PCA directions, right: quadratic part of $f(x)$.

This shows that in this case, $x^\star$ depends linearly on the data y, and the corresponding linear operator $(A^\top A)^{-1}A^\star$ is often called the Moore-Penrose pseudo-inverse of A (which is not invertible in general, since typically $p \neq n$). The condition that $A^\top A$ is invertible is equivalent to $\ker(A) = \{0\}$, since

$$A^\top Ax = 0 \quad \Longrightarrow \quad \|Ax\|^2 = \langle A^\top Ax,\, x\rangle = 0 \quad \Longrightarrow \quad Ax = 0.$$

In particular, if $n < p$ (under-determined regime, there is too much parameter or too few data) this can never holds. If $n \geqslant p$ and the features x_i are "random" then $\ker(A) = \{0\}$ with probability one. In this overdetermined situation $n \geqslant p$, $\ker(A) = \{0\}$ only holds if the features $\{a_i\}_{i=1}^n$ spans a linear space $\mathrm{Im}(A^\top)$ of dimension strictly smaller than the ambient dimension p.

13.3.4 Link with PCA

Let us assume the $(a_i)_{i=1}^n$ are centered, i.e. $\sum_i a_i = 0$. If this is not the case, one needs to replace a_i by $a_i - m$ where $m \overset{\text{def.}}{=} \frac{1}{n}\sum_{i=1}^n a_i \in \mathbb{R}^p$ is the empirical mean. In this case, $\frac{C}{n} = A^\top A/n \in \mathbb{R}^{p\times p}$ is the empirical covariance of the point cloud $(a_i)_i$, it encodes the covariances between the coordinates of the points. Denoting $a_i = (a_{i,1}, \ldots, a_{i,p})^\top \in \mathbb{R}^p$ (so that $A = (a_{i,j})_{i,j}$) the coordinates, one has

$$\forall\, (k,\ell) \in \{1,\ldots,p\}^2, \quad \frac{C_{k,\ell}}{n} = \frac{1}{n}\sum_{i=1}^n a_{i,k}a_{i,\ell}.$$

In particular, $C_{k,k}/n$ is the variance along the axis k. More generally, for any unit vector $u \in \mathbb{R}^p$, $\langle Cu,\, u\rangle/n \geqslant 0$ is the variance along the axis u.

For instance, in dimension $p = 2$,

$$\frac{C}{n} = \frac{1}{n}\begin{pmatrix} \sum_{i=1}^n a_{i,1}^2 & \sum_{i=1}^n a_{i,1}a_{i,2} \\ \sum_{i=1}^n a_{i,1}a_{i,2} & \sum_{i=1}^n a_{i,2}^2 \end{pmatrix}.$$

Since C is a symmetric, it diagonalizes in an ortho-basis $U = (u_1, \ldots, u_p) \in \mathbb{R}^{p\times p}$. Here, the vectors $u_k \in \mathbb{R}^p$ are stored in the columns of the matrix U. The diagonalization means that there exist scalars (the eigenvalues) $(\lambda_1, \ldots, \lambda_p)$ so that $(\frac{1}{n}C)u_k = \lambda_k u_k$. Since the matrix is orthogononal, $UU^\top = U^\top U = \mathrm{Id}_p$, and equivalently $U^{-1} = U^\top$. The diagonalization property can be conveniently written as $\frac{1}{n}C = U\,\mathrm{diag}(\lambda_k)U^\top$. One can thus re-write the covariance quadratic form in the basis U as being a separable sum of p squares

$$\frac{1}{n}\langle Cx,\, x\rangle = \langle U\,\mathrm{diag}(\lambda_k)U^\top x,\, x\rangle = \langle \mathrm{diag}(\lambda_k)(U^\top x),\, (U^\top x)\rangle = \sum_{k=1}^p \lambda_k \langle x,\, u_k\rangle^2. \tag{13.10}$$

Here $(U^\top x)_k = \langle x,\, u_k\rangle$ is the coordinate k of x in the basis U. Since $\langle Cx,\, x\rangle = \|Ax\|^2$, this shows that all the eigenvalues $\lambda_k \geqslant 0$ are positive.

If one assumes that the eigenvalues are ordered $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_p$, then projecting the points a_i on the first m eigenvectors can be shown to be in some sense the best linear dimensionality reduction possible (see

next paragraph), and it is called Principal Component Analysis (PCA). It is useful to perform compression or dimensionality reduction, but in practice, it is mostly used for data visualization in 2-D ($m = 2$) and 3-D ($m = 3$).

The matrix C/n encodes the covariance, so one can approximate the point cloud by an ellipsoid whose main axes are the $(u_k)_k$ and the width along each axis is $\propto \sqrt{\lambda_k}$ (the standard deviations). If the data are approximately drawn from a Gaussian distribution, whose density is proportional to $\exp(\frac{-1}{2}\langle C^{-1}a,\, a\rangle)$, then the fit is good. This should be contrasted with the shape of quadratic part $\frac{1}{2}\langle Cx,\, x\rangle$ of $f(x)$, since the ellipsoid $\left\{x\,;\, \frac{1}{n}\langle Cx,\, x\rangle \leqslant 1\right\}$ has the same main axes, but the widths are the inverse $1/\sqrt{\lambda_k}$. Figure 13.7 shows this in dimension $p = 2$.

13.3.5 Classification

We can do a similar computation for the gradient of the classification loss (13.4). Assuming that L is differentiable, and using the Taylor expansion (13.7) at point $-\operatorname{diag}(y)Ax$, one has

$$\begin{aligned} f(x+\varepsilon) &= L(-\operatorname{diag}(y)Ax - \operatorname{diag}(y)A\varepsilon) \\ &= L(-\operatorname{diag}(y)Ax) + \langle \nabla L(-\operatorname{diag}(y)Ax),\, -\operatorname{diag}(y)A\varepsilon\rangle + o(\|\operatorname{diag}(y)A\varepsilon\|). \end{aligned}$$

Using the fact that $o(\|\operatorname{diag}(y)A\varepsilon\|) = o(\|\varepsilon\|)$, one obtains

$$\begin{aligned} f(x+\varepsilon) &= f(x) + \langle \nabla L(-\operatorname{diag}(y)Ax),\, -\operatorname{diag}(y)A\varepsilon\rangle + o(\|\varepsilon\|) \\ &= f(x) + \langle -A^\top \operatorname{diag}(y)\nabla L(-\operatorname{diag}(y)Ax),\, \varepsilon\rangle + o(\|\varepsilon\|), \end{aligned}$$

where we have used the fact that $(AB)^\top = B^\top A^\top$ and that $\operatorname{diag}(y)^\top = \operatorname{diag}(y)$. This shows that

$$\nabla f(x) = -A^\top \operatorname{diag}(y)\nabla L(-\operatorname{diag}(y)Ax).$$

Since $L(z) = \sum_i \ell(z_i)$, one has $\nabla L(z) = (\ell'(z_i))_{i=1}^n$. For instance, for the logistic classification method, $\ell(u) = \log(1+\exp(u))$ so that $\ell'(u) = \frac{e^u}{1+e^u} \in [0,1]$ (which can be interpreted as a probability of predicting $+1$).

13.3.6 Chain Rule

One can formalize the previous computation, if $f(x) = g(Bx)$ with $B \in \mathbb{R}^{q\times p}$ and $g : \mathbb{R}^q \to \mathbb{R}$, then

$$f(x+\varepsilon) = g(Bx+B\varepsilon) = g(Bx) + \langle \nabla g(Bx),\, B\varepsilon\rangle + o(\|B\varepsilon\|) = f(x) + \langle \varepsilon,\, B^\top \nabla g(Bx)\rangle + o(\|\varepsilon\|),$$

which shows that

$$\nabla(g \circ B) = B^\top \circ \nabla g \circ B \tag{13.11}$$

where "$\circ$" denotes the composition of functions.

To generalize this to composition of possibly non-linear functions, one needs to use the notion of differential. For a function $F : \mathbb{R}^p \to \mathbb{R}^q$, its differentiable at x is a linear operator $\partial F(x) : \mathbb{R}^p \to \mathbb{R}^q$, i.e. it can be represented as a matrix (still denoted $\partial F(x)$) $\partial F(x) \in \mathbb{R}^{q\times p}$. The entries of this matrix are the partial differential, denoting $F(x) = (F_1(x), \ldots, F_q(x))$,

$$\forall\, (i,j) \in \{1,\ldots,q\} \times \{1,\ldots,p\}, \quad [\partial F(x)]_{i,j} \overset{\text{def.}}{=} \frac{\partial F_i(x)}{\partial x_j}.$$

The function F is then said to be differentiable at x if and only if one has the following Taylor expansion

$$F(x+\varepsilon) = F(x) + [\partial F(x)](\varepsilon) + o(\|\varepsilon\|). \tag{13.12}$$

where $[\partial F(x)](\varepsilon)$ is the matrix-vector multiplication. As for the definition of the gradient, this matrix is the only one that satisfies this expansion, so it can be used as a way to compute this differential in practice.

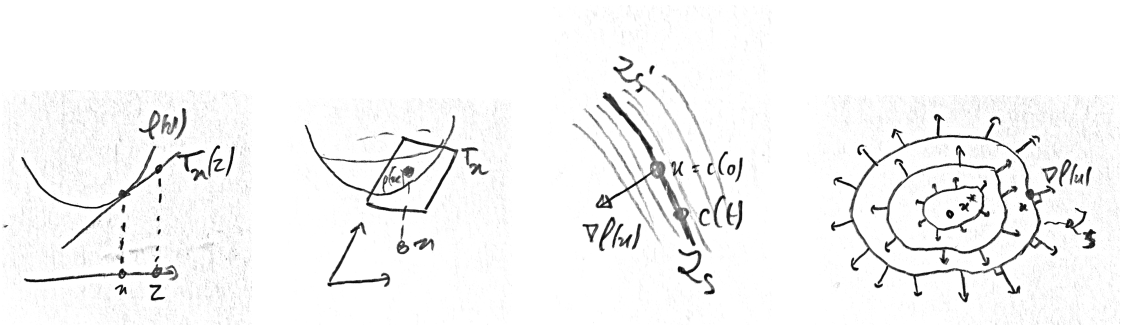


Figure 13.8: Left: First order Taylor expansion in 1-D and 2-D. Right: orthogonality of gradient and level sets and schematic of the proof.

For the special case $q = 1$, i.e. if $f : \mathbb{R}^p \to \mathbb{R}$, then the differential $\partial f(x) \in \mathbb{R}^{1\times p}$ and the gradient $\nabla f(x) \in \mathbb{R}^{p\times 1}$ are linked by equating the Taylor expansions (13.12) and (13.7)

$$\forall\, \varepsilon \in \mathbb{R}^p, \quad [\partial f(x)](\varepsilon) = \langle \nabla f(x), \varepsilon\rangle \quad \Leftrightarrow \quad [\partial f(x)](\varepsilon) = \nabla f(x)^\top.$$

The differential satisfies the following chain rule

$$\partial(G \circ H)(x) = [\partial G(H(x))] \times [\partial H(x)]$$

where "$\times$" is the matrix product. For instance, if $H : \mathbb{R}^p \to \mathbb{R}^q$ and $G = g : \mathbb{R}^q \mapsto \mathbb{R}$, then $f = g \circ H : \mathbb{R}^p \to \mathbb{R}$ and one can compute its gradient as follow

$$\nabla f(x) = (\partial f(x))^\top = ([\partial g(H(x))] \times [\partial H(x)])^\top = [\partial H(x)]^\top \times [\partial g(H(x))]^\top = [\partial H(x)]^\top \times \nabla g(H(x)).$$

When $H(x) = Bx$ is linear, one recovers formula (13.11).

13.4 Gradient Descent Algorithm

13.4.1 Steepest Descent Direction

The Taylor expansion (13.7) computes an affine approximation of the function f near x, since it can be written as

$$f(z) = T_x(z) + o(\|x - z\|) \quad \text{where} \quad T_x(z) \stackrel{\text{def.}}{=} f(x) + \langle \nabla f(x),\, z - x\rangle,$$

see Fig. 13.8. First order methods operate by locally replacing f by T_x.

The gradient $\nabla f(x)$ should be understood as a direction along which the function increases. This means that to improve the value of the function, one should move in the direction $-\nabla f(x)$. Given some fixed x, let us look as the function f along the 1-D half line

$$\tau \in \mathbb{R}^+ = [0, +\infty[\longmapsto f(x - \tau \nabla f(x)) \in \mathbb{R}.$$

If f is differentiable at x, one has

$$f(x - \tau\nabla f(x)) = f(x) - \tau\langle \nabla f(x),\, \nabla f(x)\rangle + o(\tau) = f(x) - \tau\|\nabla f(x)\|^2 + o(\tau).$$

So there are two possibility: either $\nabla f(x) = 0$, in which case we are already at a minimum (possibly a local minimizer if the function is non-convex) or if τ is chosen small enough,

$$f(x - \tau\nabla f(x)) < f(x)$$

which means that moving from x to $x - \tau\nabla f(x)$ has improved the objective function.

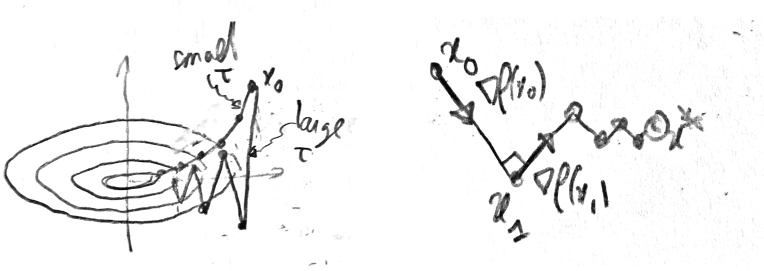


Figure 13.9: Influence of τ on the gradient descent (left) and optimal step size choice (right).

Remark 3 (Orthogonality to level sets). The level sets of f are the sets of point sharing the same value of f, i.e. for any $s \in \mathbb{R}$

$$\mathcal{L}_s \overset{\text{def.}}{=} \{x \; ; \; f(x) = s\}.$$

At some $x \in \mathbb{R}^p$, denoting $s = f(x)$, then $x \in \mathcal{L}_s$ (x belong to its level set). The gradient vector $\nabla f(x)$ is orthogonal to the level set (as shown on Fig. 13.8 right), and points toward level set of higher value (which is consistent with the previous computation showing that it is a valid ascent direction). Indeed, lets consider around x inside $\mathcal{L}_s$ a smooth curve of the form $t \in \mathbb{R} \mapsto c(t)$ where $c(0) = x$. Then the function $h(t) \overset{\text{def.}}{=} f(c(t))$ is constant $h(t) = s$ since $c(t)$ belong to the level set. So $h'(t) = 0$. But at the same time, we can compute its derivate at $t = 0$ as follow

$$h(t) = f(c(0) + tc'(0) + o(t)) = h(0) + \delta\langle c'(0), \nabla f(c(0))\rangle + o(t)$$

i.e. $h'(0) = \langle c'(0), \nabla f(x)\rangle = 0$, so that $\nabla f(x)$ is orthogonal to the tangent $c'(0)$ of the curve c, which lies in the tangent plane of $\mathcal{L}_s$ (as shown on Fig. 13.8, right). Since the curve c is arbitrary, the whole tangent plane is thus orthogonal to $\nabla f(x)$.

Remark 4 (Local optimal descent direction). One can prove something even stronger, that among all possible direction u with $\|u\| = r$, $r\frac{\nabla f(x)}{\|\nabla f(x)\|}$ becomes the optimal one as $r \to 0$ (so for very small step this is locally the best choice), more precisely,

$$\frac{1}{r}\operatorname*{argmin}_{\|u\|=r} f(x+u) \overset{r\to 0}{\longrightarrow} -\frac{\nabla f(x)}{\|\nabla f(x)\|}.$$

Indeed, introducing a Lagrange multiplier $\lambda \in \mathbb{R}$ for this constraint optimization problem, one obtains that the optimal u satisfies $\nabla f(x+u) = \lambda u$ and $\|u\| = r$. Thus $\frac{u}{r} = \pm\frac{\nabla f(x+u)}{\|\nabla f(x+u)\|}$, and assuming that ∇f is continuous, when $\|u\| = r \to 0$, this converges to $\frac{u}{\|u\|} = \pm\frac{\nabla f(x)}{\|\nabla f(x)\|}$. The sign $\pm$ should be $+1$ to obtain a maximizer and -1 for the minimizer.

13.4.2 Gradient Descent

The gradient descent algorithm reads, starting with some $x_0 \in \mathbb{R}^p$

$$x_{k+1} \overset{\text{def.}}{=} x_k - \tau_k \nabla f(x_k) \tag{13.13}$$

where $\tau_k > 0$ is the step size (also called learning rate). For a small enough τ_k, the previous discussion shows that the function f is decaying through the iteration. So intuitively, to ensure convergence, τ_k should be chosen small enough, but not too small so that the algorithm is as fast as possible. In general, one use a fix step size $\tau_k = \tau$, or try to adapt τ_k at each iteration (see Fig. 13.9).

Remark 5 (Greedy choice). Although this is in general too costly to perform exactly, one can use a "greedy" choice, where the step size is optimal at each iteration, i.e.

$$\tau_k \overset{\text{def.}}{=} \operatorname*{argmin}_{\tau} h(\tau) \overset{\text{def.}}{=} f(x_k - \tau\nabla f(x_k)).$$

Here $h(\tau)$ is a function of a single variable. One can compute the derivative of h as

$$h(\tau+\delta) = f(x_k - \tau\nabla f(x_k) - \delta\nabla f(x_k)) = f(x_k - \tau\nabla f(x_k)) - \langle \nabla f(x_k - \tau\nabla f(x_k)),\, \nabla f(x_k)\rangle + o(\delta).$$

One note that at $\tau = \tau_k$, $\nabla f(x_k - \tau\nabla f(x_k)) = \nabla f(x_{k+1})$ by definition of x_{k+1} in (17.2). Such an optimal $\tau = \tau_k$ is thus characterized by

$$h'(\tau_k) = -\langle \nabla f(x_k),\, \nabla f(x_{k+1})\rangle = 0.$$

This means that for this greedy algorithm, two successive descent direction $\nabla f(x_k)$ and $\nabla f(x_{k+1})$ are orthogonal (see Fig. 13.9).

13.5 Convergence Analysis

13.5.1 Quadratic Case

Convergence analysis for the quadratic case. We first analyze this algorithm in the case of the quadratic loss, which can be written as

$$f(x) = \frac{1}{2}\|Ax-y\|^2 = \frac{1}{2}\langle Cx,\, x\rangle - \langle x,\, b\rangle + \text{cst} \quad \text{where} \quad \begin{cases} C \overset{\text{def.}}{=} A^\top A \in \mathbb{R}^{p\times p}, \\ b \overset{\text{def.}}{=} A^\top y \in \mathbb{R}^p. \end{cases}$$

We already saw that in (13.9) if $\ker(A) = \{0\}$, which is equivalent to C being invertible, then there exists a single global minimizer $x^\star = (A^\top A)^{-1}A^\top y = C^{-1}u$.

Note that a function of the form $\frac{1}{2}\langle Cx,\, x\rangle - \langle x,\, b\rangle$ is convex if and only if the symmetric matrix C is positive semi-definite, i.e. that all its eigenvalues are non-negative (as already seen in (13.10)).

Proposition 41. *For $f(x) = \langle Cx,\, x\rangle - \langle b,\, x\rangle$ (C being symmetric semi-definite positive) with the eigenvalues of C upper-bounded by L and lower-bounded by $\mu > 0$, assuming there exists $(\tau_{\min}, \tau_{\max})$ such that*

$$0 < \tau_{\min} \leqslant \tau_\ell \leqslant \tilde{\tau}_{\max} < \frac{2}{L}$$

then there exists $0 \leqslant \tilde{\rho} < 1$ such that

$$\|x_k - x^\star\| \leqslant \tilde{\rho}^\ell \|x_0 - x^\star\|. \tag{13.14}$$

The best rate $\tilde{\rho}$ is obtained for

$$\tau_\ell = \frac{2}{L+\mu} \quad \Longrightarrow \quad \tilde{\rho} \overset{\text{def.}}{=} \frac{L-\mu}{L+\mu} = 1 - \frac{2\varepsilon}{1+\varepsilon} \quad \textit{where} \quad \varepsilon \overset{\text{def.}}{=} \mu/L. \tag{13.15}$$

Proof. One iterate of gradient descent reads

$$x_{k+1} = x_k - \tau_\ell(Cx_k - b).$$

Since the solution $x^\star$ (which by the way is unique by strict convexity) satisfy the first order condition $Cx^\star = b$, it gives

$$x_{k+1} - x^\star = x_k - x^\star - \tau_\ell C(x_k - x^\star) = (\mathrm{Id}_p - \tau_\ell C)(x_k - x^\star).$$

If $S \in \mathbb{R}^{p\times p}$ is a symmetric matrix, one has

$$\|Sz\| \leqslant \|S\|_{\text{op}}\|z\| \quad \text{where} \quad \|S\|_{\text{op}} \overset{\text{def.}}{=} \max_k |\lambda_k(S)|,$$

where $\lambda_k(S)$ are the eigenvalues of S and $\sigma_k(S) \overset{\text{def.}}{=} |\lambda_k(S)|$ are its singular values. Indeed, S can be diagonalized in an orthogonal basis U, so that $S = U\,\mathrm{diag}(\lambda_k(S))U^\top$, and $S^\top S = S^2 = U\,\mathrm{diag}(\lambda_k(S)^2)U^\top$ so that

$$\begin{aligned}\|Sz\|^2 &= \langle S^\top Sz,\, z\rangle = \langle U\,\mathrm{diag}(\lambda_k)U^\top z,\, z\rangle = \langle \mathrm{diag}(\lambda_k^2)U^\top z,\, U^\top z\rangle \\ &= \sum_i \lambda_k^2 (U^\top z)_k^2 \leqslant \max_k(\lambda_k^2)\|U^\top z\|^2 = \max_k(\lambda_k^2)\|z\|^2.\end{aligned}$$

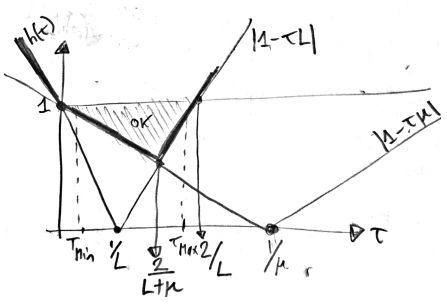


Figure 13.10: Contraction constant $h(\tau)$ for a quadratic function (right).

Applying this to $S = \mathrm{Id}_p - \tau_\ell C$, one has

$$h(\tau) \overset{\text{def.}}{=} \|\mathrm{Id}_p - \tau_\ell C\|_{\text{op}} = \max_k |\lambda_k(\mathrm{Id}_p - \tau_\ell C)| = \max_k |1 - \tau_\ell \lambda_k(C)| = \max(|1 - \tau_\ell \sigma_{\max}(C)|, |1 - \tau \sigma_{\min}(C)|)$$

For a quadratic function, one has $\sigma_{\min}(C) = \mu, \sigma_{\max}(C) = L$. Figure 13.10, right, shows a display of $h(\tau)$. One has that for $0 < \tau < 2/L$, $h(\tau) < 1$. The optimal value is reached at $\tau^\star = \frac{2}{L+\mu}$ and then

$$h(\tau^\star) = \left|1 - \frac{2L}{L+\mu}\right| = \frac{L-\mu}{L+\mu}.$$

□

Note that when the condition number $\xi \overset{\text{def.}}{=} \mu/L \ll 1$ is small (which is the typical setup for ill-posed problems), then the contraction constant appearing in (13.15) scales like

$$\tilde{\rho} \sim 1 - 2\xi. \tag{13.16}$$

The quantity ε in some sense reflects the inverse-conditioning of the problem. For quadratic function, it indeed corresponds exactly to the inverse of the condition number (which is the ratio of the largest to smallest singular value). The condition number is minimum and equal to 1 for orthogonal matrices.

The error decay rate (13.14), although it is geometrical $O(\rho^k)$ is called a "linear rate" in the optimization literature. It is a "global" rate because it hold for all k (and not only for large enough k).

If $\ker(A) \neq \{0\}$, then C is not definite positive (some of its eigenvalues vanish), and the set of solution is infinite. One can however still show a linear rate, by showing that actually the iterations x_k are orthogonal to $\ker(A)$ and redo the above proof replacing μ by the smaller non-zero eigenvalue of C. This analysis however leads to a very poor rate ρ (very close to 1) because μ can be arbitrary close to 0. Furthermore, such a proof does not extends to non-quadratic functions. It is thus necessary to do a different theoretical analysis, which only shows a sublinear rate on the objective function f itself rather than on the iterates x_k.

Proposition 42. *For $f(x) = \langle Cx, x\rangle - \langle b, x\rangle$, assuming the eigenvalue of C are bounded by L, then if $0 < \tau_k = \tau < 1/L$ is constant, then*

$$f(x_k) - f(x^\star) \leqslant \frac{\mathrm{dist}(x_0, \mathrm{argmin}\, f)^2}{\tau 8k}.$$

where

$$\mathrm{dist}(x_0, \mathrm{argmin}\, f) \overset{\text{def.}}{=} \min_{x^\star \in \mathrm{argmin}\, f} \|x_0 - x^\star\|.$$

Proof. We have $Cx^\star = b$ for any minimizer $x^\star$ and $x_{k+1} = x_k - \tau(Cx_k - b)$ so that as before

$$x_k - x^\star = (\mathrm{Id}_p - \tau C)^k (x_0 - x^\star).$$

Now one has

$$\frac{1}{2}\langle C(x_k - x^\star),\, x_k - x^\star\rangle = \frac{1}{2}\langle Cx_k,\, x_k\rangle - \langle Cx_k,\, x^\star\rangle + \frac{1}{2}\langle Cx^\star,\, x^\star\rangle$$

and we have $\langle Cx_k,\, x^\star\rangle = \langle x_k,\, Cx^\star\rangle = \langle x_k,\, b\rangle$ and also $\langle Cx^\star,\, x^\star\rangle = \langle x^\star,\, b\rangle$ so that

$$\frac{1}{2}\langle C(x_k - x^\star,\, x_k - x^\star\rangle = \frac{1}{2}\langle Cx_k,\, x_k\rangle - \langle x_k,\, b\rangle + \frac{1}{2}\langle x^\star,\, b\rangle = f(x_k) + \frac{1}{2}\langle x^\star,\, b\rangle.$$

Note also that

$$f(x^\star) = \frac{1}{2}\frac{Cx^\star}{x^\star} - \langle x^\star,\, b\rangle = \frac{1}{2}\langle x^\star,\, b\rangle - \langle x^\star,\, b\rangle = -\frac{1}{2}\langle x^\star,\, b\rangle.$$

This shows that

$$\frac{1}{2}\langle C(x_k - x^\star,\, x_k - x^\star\rangle = f(x_k) - f(x^\star).$$

This thus implies

$$f(x_k) - f(x^\star) = \frac{1}{2}\langle (\mathrm{Id}_p - \tau C)^k C(\mathrm{Id}_p - \tau C)^k (x_0 - x^\star),\, x_0 - x^\star\rangle \leqslant \frac{\sigma_{\max}(M_k)}{2}\|x_0 - x^\star\|^2$$

where we have denoted

$$M_k \stackrel{\text{def.}}{=} (\mathrm{Id}_p - \tau C)^k C(\mathrm{Id}_p - \tau C)^k.$$

Since $x^\star$ can be chosen arbitrary, one can replace $\|x_0 - x^\star\|$ by $\mathrm{dist}(x_0, \mathrm{argmin}\, f)$. One has, for any ℓ, the following bound

$$\sigma_\ell(M_k) = \sigma_\ell(C)(1 - \tau\sigma_\ell(C))^{2k} \leqslant \frac{1}{\tau 4k}$$

since one can show that (setting $t = \tau\sigma_\ell(C) \leqslant 1$ because of the hypotheses)

$$\forall\, t \in [0,1], \quad (1-t)^{2k} t \leqslant \frac{1}{4k}.$$

Indeed, one has

$$(1-t)^{2k} t \leqslant (e^{-t})^{2k} t = \frac{1}{2k}(2kt)e^{-2kt} \leqslant \frac{1}{2k}\sup_{u \geqslant 0} ue^{-u} = \frac{1}{2ek} \leqslant \frac{1}{4k}.$$

□

13.5.2 General Case

We detail the theoretical analysis of convergence for general smooth convex functions. The general idea is to replace the linear operator C involved in the quadratic case by the second order derivative (the hessian matrix).

Hessian. If the function is twice differentiable along the axes, the hessian matrix is

$$(\partial^2 f)(x) = \left(\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\right)_{1 \leqslant i,j \leqslant p} \in \mathbb{R}^{p \times p}.$$

Where recall that $\frac{\partial^2 f(x)}{\partial x_i \partial x_j}$ is the differential along direction x_j of the function $x \mapsto \frac{\partial f(x)}{\partial x_i}$. We also recall that $\frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \frac{\partial^2 f(x)}{\partial x_j \partial x_i}$ so that $\partial^2 f(x)$ is a symmetric matrix.

A differentiable function f is said to be twice differentiable at x if

$$f(x+\varepsilon) = f(x) + \langle \nabla f(x),\, \varepsilon\rangle + \frac{1}{2}\langle \partial^2 f(x)\varepsilon,\, \varepsilon\rangle + o(\|\varepsilon\|^2). \tag{13.17}$$

This means that one can approximate f near x by a quadratic function. The hessian matrix is uniquely determined by this relation, so that if one is able to write down an expansion with some matrix H

$$f(x+\varepsilon) = f(x) + \langle \nabla f(x), \varepsilon \rangle + \frac{1}{2}\langle H\varepsilon, \varepsilon \rangle + o(\|\varepsilon\|^2).$$

then equating this with the expansion (13.17) ensure that $\partial^2 f(x) = H$. This is thus a way to actually determine the hessian without computing all the p^2 partial derivative. This Hessian can equivalently be obtained by performing an expansion (i.e. computing the differential) of the gradient since

$$\nabla f(x+\varepsilon) = \nabla f(x) + [\partial^2 f(x)](\varepsilon) + o(\|\varepsilon\|)$$

where $[\partial^2 f(x)](\varepsilon) \in \mathbb{R}^p$ denotes the multiplication of the matrix $\partial^2 f(x)$ with the vector ε.

One can show that a twice differentiable function f on $\mathbb{R}^p$ is convex if and only if for all x the symmetric matrix $\partial^2 f(x)$ is positive semi-definite, i.e. all its eigenvalues are non-negative. Furthermore, if these eigenvalues are strictly positive then f is strictly convex (but the converse is not true, for instance x^4 is strictly convex on $\mathbb{R}$ but its second derivative vanishes at $x = 0$).

For instance, for a quadratic function $f(x) = \langle Cx, x \rangle - \langle x, u \rangle$, one has $\nabla f(x) = Cx - u$ and thus $\partial^2 f(x) = C$ (which is thus constant). For the classification function, one has

$$\nabla f(x) = -A^\top \operatorname{diag}(y) \nabla L(-\operatorname{diag}(y)Ax).$$

and thus

$$\begin{aligned}\nabla f(x+\varepsilon) &= -A^\top \operatorname{diag}(y) \nabla L(-\operatorname{diag}(y)Ax - -\operatorname{diag}(y)A\varepsilon) \\ &= \nabla f(x) - A^\top \operatorname{diag}(y) [\partial^2 L(-\operatorname{diag}(y)Ax)](-\operatorname{diag}(y)A\varepsilon)\end{aligned}$$

Since $\nabla L(u) = (\ell'(u_i))$ one has $\partial^2 L(u) = \operatorname{diag}(\ell''(u_i))$. This means that

$$\partial^2 f(x) = A^\top \operatorname{diag}(y) \times \operatorname{diag}(\ell''(-\operatorname{diag}(y)Ax)) \times \operatorname{diag}(y)A.$$

One verifies that this matrix is symmetric and positive if ℓ is convex and thus ℓ'' is positive.

Remark 6 (Second order optimality condition). The first use of Hessian is to decide wether a point $x^\star$ with $\nabla f(x^\star)$ is a local minimum or not. Indeed, if $\partial^2 f(x^\star)$ is a positive matrix (i.e. its eigenvalues are strictly positive), then $x^\star$ is a strict local minimum. Note that if $\partial^2 f(x^\star)$ is only non-negative (i.e. some its eigenvalues might vanish) then one cannot deduce anything (such as for instance x^3 on $\mathbb{R}$). Conversely, if $x^\star$ is a local minimum then $\partial^2 f(x)$

Remark 7 (Second order algorithms). A second use, is to be used in practice to define second order method (such as Newton's algorithm), which converge faster than gradient descent, but are more costly. The generalized gradient descent reads

$$x_{k+1} = x_k - H_k \nabla f(x_k)$$

where $H_k \in \mathbb{R}^{p \times p}$ is a positive symmetric matrix. One recovers the gradient descent when using $H_k = \tau_k \mathrm{Id}_p$, and Newton's algorithm corresponds to using the inverse of the Hessian $H_k = [\partial^2 f(x_k)]^{-1}$. Note that

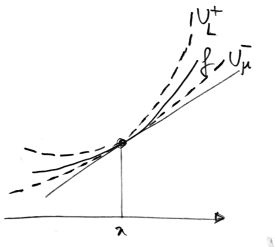
$$f(x_k) = f(x_k) - \langle H_k \nabla f(x_k), \nabla f(x_k) \rangle + o(\|H_k \nabla f(x_k)\|).$$

Since H_k is positive, if x_k is not a minimizer, i.e. $\nabla f(x_k) \neq 0$, then $\langle H_k \nabla f(x_k), \nabla f(x_k) \rangle > 0$. So if H_k is small enough one has a valid descent method in the sense that $f(x_{k+1}) < f(x_k)$. It is not the purpose of this chapter to explain in more detail these type of algorithm.

The last use of Hessian, that we explore next, is to study theoretically the convergence of the gradient descent. One simply needs to replace the boundedness of the eigenvalue of C of a quadratic function by a boundedness of the eigenvalues of $\partial^2 f(x)$ for all x. Roughly speaking, the theoretical analysis of the gradient descent for a generic function is obtained by applying this approximation and using the proofs of the previous section.

Smoothness and strong convexity. One also needs to quantify the smoothness of f. This is enforced by requiring that the gradient is L-Lipschitz, i.e.

$$\forall\,(x,x')\in(\mathbb{R}^p)^2,\quad \|\nabla f(x)-\nabla f(x')\|\leqslant L\|x-x'\|. \tag{$\mathcal{R}_L$}$$



In order to obtain fast convergence of the iterates themselve, it is needed that the function has enough "curvature" (i.e. is not too flat), which corresponds to imposing that f is μ-strongly convex

$$\forall\,(x,x'),\in(\mathbb{R}^p)^2,\quad \langle\nabla f(x)-\nabla f(x'),x-x'\rangle\geqslant\mu\|x-x'\|^2. \tag{$\mathcal{S}_\mu$}$$

The following proposition express these conditions as constraints on the hessian for $\mathcal{C}^2$ functions.

Proposition 43. *Conditions ($\mathcal{R}_L$) and ($\mathcal{S}_\mu$) imply*

$$\forall\,(x,x'),\quad f(x')+\langle\nabla f(x),x'-x\rangle+\frac{\mu}{2}\|x-x'\|^2\leqslant f(x)\leqslant f(x')+\langle\nabla f(x'),x'-x\rangle+\frac{L}{2}\|x-x'\|^2. \tag{13.18}$$

If f is of class $\mathcal{C}^2$, conditions ($\mathcal{R}_L$) and ($\mathcal{S}_\mu$) are equivalent to

$$\forall\,x,\quad \mu\mathrm{Id}_p\preceq\partial^2 f(x)\preceq L\mathrm{Id}_p \tag{13.19}$$

where $\partial^2 f(x)\in\mathbb{R}^{p\times p}$ is the Hessian of f, and where $\preceq$ is the natural order on symmetric matrices, i.e.

$$A\preceq B\quad\Longleftrightarrow\quad\forall\,x\in\mathbb{R}^p,\quad\langle Au,u\rangle\leqslant\langle Bu,u\rangle.$$

Proof. We prove (13.18), using Taylor expansion with integral remain

$$f(x')-f(x)=\int_0^1\langle\nabla f(x_t),x'-x\rangle\mathrm{d}t=\langle\nabla f(x),x'-x\rangle+\int_0^1\langle\nabla f(x_t)-\nabla f(x),x'-x\rangle\mathrm{d}t$$

where $x_t\overset{\text{def.}}{=}x+t(x'-x)$. Using Cauchy-Schwartz, and then the smoothness hypothesis ($\mathcal{R}_L$)

$$f(x')-f(x)\leqslant\langle\nabla f(x),x'-x\rangle+\int_0^1 L\|x_t-x\|\|x'-x\|\mathrm{d}t\leqslant\langle\nabla f(x),x'-x\rangle+L\|x'-x\|^2\int_0^1 t\mathrm{d}t$$

which is the desired upper-bound. Using directly ($\mathcal{S}_\mu$) gives

$$f(x')-f(x)=\langle\nabla f(x),x'-x\rangle+\int_0^1\langle\nabla f(x_t)-\nabla f(x),\frac{x_t-x}{t}\rangle\mathrm{d}t\geqslant\langle\nabla f(x),x'-x\rangle+\mu\int_0^1\frac{1}{t}\|x_t-x\|^2\mathrm{d}t$$

which gives the desired result since $\|x_t-x\|^2/t=t\|x'-x\|^2$. □

The relation (13.18) shows that a smooth (resp. strongly convex) functional is bounded by below (resp. above) by a quadratic tangential majorant (resp. minorant).

Condition (13.19) thus reads that the singular values of $\partial^2 f(x)$ should be contained in the interval $[\mu,L]$. The upper bound is also equivalent to $\|\partial^2 f(x)\|_{\text{op}}\leqslant L$ where $\|\cdot\|_{\text{op}}$ is the operator norm, i.e. the largest singular value. In the special case of a quadratic function of the form $\langle Cx,x\rangle-\langle b,x\rangle$ (recall that necessarily C is semi-definite symmetric positive for this function to be convex), $\partial^2 f(x)=C$ is constant, so that $[\mu,L]$ can be chosen to be the range of the eigenvalues of C.

Convergence analysis. We now give convergence theorem for a general convex function. On contrast to quadratic function, if one does not assumes strong convexity, one can only show a sub-linear rate on the function values (and no rate at all on the iterates themselves). It is only when one assume strong convexity that linear rate is obtained. Note that in this case, the solution of the minimization problem is not necessarily unique.

Theorem 23. *If f satisfy conditions $(\mathcal{R}_L)$, assuming there exists $(\tau_{\min}, \tau_{\max})$ such that*

$$0 < \tau_{\min} \leqslant \tau_\ell \leqslant \tau_{\max} < \frac{2}{L},$$

then x_k converges to a solution $x^\star$ of (17.1) and there exists $C > 0$ such that

$$f(x_k) - f(x^\star) \leqslant \frac{C}{\ell + 1}. \tag{13.20}$$

If furthermore f is μ-strongly convex, then there exists $0 \leqslant \rho < 1$ such that $\|x_k - x^\star\| \leqslant \rho^\ell \|x_0 - x^\star\|$.

Proof. In the case where f is not strongly convex, we only prove (13.20) since the proof that x_k converges is more technical. Note indeed that if the minimizer $x^\star$ is non-unique, then it might be the case that the iterate x_k "cycle" while approaching the set of minimizer, but actually convexity of f prevents this kind of pathological behavior. For simplicity, we do the proof in the case $\tau_\ell = 1/L$, but it extends to the general case. The L-smoothness property imply (13.18), which reads

$$f(x_{k+1}) \leqslant f(x_k) + \langle \nabla f(x_k),\, x_{k+1} - x_k \rangle + \frac{L}{2}\|x_{k+1} - x_k\|^2.$$

Using the fact that $x_{k+1} - x_k = -\frac{1}{L}\nabla f(x_k)$, one obtains

$$f(x_{k+1}) \leqslant f(x_k) - \frac{1}{L}\|\nabla f(x_k)\|^2 + \frac{1}{2L}\|\nabla f(x_k)\|^2 \leqslant f(x_k) - \frac{1}{2L}\|\nabla f(x_k)\|^2 \tag{13.21}$$

This shows that $(f(x_k))_\ell$ is a decaying sequence. By convexity

$$f(x_k) + \langle \nabla f(x_k),\, x^\star - x_k \rangle \leqslant f(x^\star)$$

and plugging this in (13.21) shows

$$\begin{aligned} f(x_{k+1}) &\leqslant f(x^\star) - \langle \nabla f(x_k),\, x^\star - x_k \rangle - \frac{1}{2L}\|\nabla f(x_k)\|^2 && (13.22)\\ &= f(x^\star) + \frac{L}{2}\left(\|x_k - x^\star\|^2 - \|x_k - x^\star - \frac{1}{L}\nabla f(x_k)\|^2 \right) && (13.23)\\ &= f(x^\star) + \frac{L}{2}\left(\|x_k - x^\star\|^2 - \|x^\star - x_{k+1}\|^2 \right). && (13.24) \end{aligned}$$

Summing these inequalities for $\ell = 0, \ldots, k$, one obtains

$$\sum_{\ell=0}^{k} f(x_{k+1}) - (k+1) f(x^\star) \leqslant \frac{L}{2}\left(\|x_0 - x^\star\|^2 - \|x^{(k+1)} - x^\star\|^2 \right)$$

and since $f(x_{k+1})$ is decaying $\sum_{\ell=0}^{k} f(x_{k+1}) \geqslant (k+1) f(x^{(k+1)})$, thus

$$f(x^{(k+1)}) - f(x^\star) \leqslant \frac{L\|x_0 - x^\star\|^2}{2(k+1)}$$

which gives (13.20) for $C \overset{\text{def.}}{=} L\|x_0 - x^\star\|^2/2$.

If we now assume f is μ-strongly convex, then, using $\nabla f(x^\star) = 0$, one has $\frac{\mu}{2}\|x^\star - x\|^2 \leqslant f(x) - f(x^\star)$ for all x. Re-manipulating (13.24) gives

$$\frac{\mu}{2}\|x_{k+1} - x^\star\|^2 \leqslant f(x_{k+1}) - f(x^\star) \leqslant \frac{L}{2}\left(\|x_k - x^\star\|^2 - \|x^\star - x_{k+1}\|^2 \right),$$

and hence

$$\|x_{k+1} - x^\star\| \leqslant \sqrt{\frac{L}{L+\mu}}\|x_{k+1} - x^\star\|, \tag{13.25}$$

which is the desired result. □

Note that in the low conditioning setting $\varepsilon \ll 1$, one retrieve a dependency of the rate (13.25) similar to the one of quadratic functions (13.16), indeed

$$\sqrt{\frac{L}{L+\mu}} = (1+\varepsilon)^{-\frac{1}{2}} \sim 1 - \frac{1}{2}\varepsilon.$$

Chapter 14

Optimization & Machine Learning: Advanced Topics

14.1 Regularization

When the number n of sample is not large enough with respect to the dimension p of the model, it makes sense to regularize the empirical risk minimization problem.

14.1.1 Penalized Least Squares

For the sake of simplicity, we focus here on regression and consider

$$\min_{x\in\mathbb{R}^p} f_\lambda(x) \stackrel{\text{def.}}{=} \frac{1}{2}\|Ax-y\|^2 + \lambda R(x) \tag{14.1}$$

where $R(x)$ is the regularizer and $\lambda \geqslant 0$ the regularization parameter. The regularizer enforces some prior knowledge on the weight vector x (such as small amplitude or sparsity, as we detail next) and λ needs to be tuned using cross-validation.

We assume for simplicity that R is positive and coercive, i.e. $R(x) \to +\infty$ as $\|x\| \to +\infty$. The following proposition that in the small λ limit, the regularization select a sub-set of the possible minimizer. This is especially useful when $\ker(A) \neq 0$, i.e. the equation $Ax = y$ has an infinite number of solutions.

Proposition 44. *If $(x_{\lambda_k})_k$ is a sequence of minimizers of f_λ, then this sequence is bounded, and any accumulation $x^\star$ is a solution of the constrained optimization problem*

$$\min_{Ax=y} R(x). \tag{14.2}$$

Proof. Let x_0 be so that $Ax_0 = y$, then by optimality of x_{λ_k}

$$\frac{1}{2}\|Ax_{\lambda_k}-y\|^2 + \lambda_k R(x_{\lambda_k}) \leqslant \lambda_k R(x_0). \tag{14.3}$$

Since all the term are positive, one has $R(x_{\lambda_k}) \leqslant R(x_0)$ so that $(x_{\lambda_k})_k$ is bounded by coercivity of R. Then also $\|Ax_{\lambda_k}-y\| \leqslant \lambda_k R(x_0)$, and passing to the limit, one obtains $Ax^\star = y$. And passing to the limit in $R(x_{\lambda_k}) \leqslant R(x_0)$ one has $R(x^\star) \leqslant R(x_0)$ which shows that $x^\star$ is a solution of (14.2). □

14.1.2 Ridge Regression

Ridge regression is by far the most popular regularizer, and corresponds to using $R(x) = \|x\|^2_{\mathbb{R}^p}$. Since it is strictly convex, the solution of (14.1) is unique

$$x_\lambda \stackrel{\text{def.}}{=} \operatorname*{argmin}_{x\in\mathbb{R}^p} f_\lambda(x) = \frac{1}{2}\|Ax-y\|^2_{\mathbb{R}^n} + \lambda\|x\|^2_{\mathbb{R}^p}.$$

One has

$$\nabla f_\lambda(x) = A^\top(Ax_\lambda - y) + \lambda x_\lambda = 0$$

so that x_λ depends linearly on y and can be obtained by solving a linear system. The following proposition shows that there are actually two alternate formula.

Proposition 45. *One has*

$$x_\lambda = (A^\top A + \lambda \mathrm{Id}_p)^{-1} A^\top y, \tag{14.4}$$
$$= A^\top (AA^\top + \lambda \mathrm{Id}_n)^{-1} y. \tag{14.5}$$

Proof. Denoting $B \overset{\text{def.}}{=} (A^\top A + \lambda \mathrm{Id}_p)^{-1} A^\top$ and $C \overset{\text{def.}}{=} A^\top (AA^\top + \lambda \mathrm{Id}_n)^{-1}$, one has $(A^\top A + \lambda \mathrm{Id}_p)B = A^\top$ while

$$(A^\top A + \lambda \mathrm{Id}_p)C = (A^\top A + \lambda \mathrm{Id}_p)A^\top (AA^\top + \lambda \mathrm{Id}_n)^{-1} = A^\top (AA^\top + \lambda \mathrm{Id}_n)(AA^\top + \lambda \mathrm{Id}_n)^{-1} = A^\top.$$

Since $A^\top A + \lambda \mathrm{Id}_p$ is invertible, this gives the desired result. □

The solution of these linear systems can be computed using either a direct method such as Cholesky factorization or an iterative method such as a conjugate gradient (which is vastly superior to the vanilla gradient descent scheme).

If $n > p$, then one should use (14.4) while if $n < p$ one should rather use (14.5).

Pseudo-inverse. As $\lambda \to 0$, then $x_\lambda \to x_0$ which is, using (14.2)

$$\operatorname*{argmin}_{Ax=y} \|x\|.$$

If $\ker(A) = \{0\}$ (overdetermined setting), $A^\top A \in \mathbb{R}^{p \times p}$ is an invertible matrix, and $(A^\top A + \lambda \mathrm{Id}_p)^{-1} \to (A^\top A)^{-1}$, so that

$$x_0 = A^+ y \quad \text{where} \quad A^+ \overset{\text{def.}}{=} (A^\top A)^{-1} A^\top.$$

Conversely, if $\ker(A^\top) = \{0\}$, or equivalently $\mathrm{Im}(A) = \mathbb{R}^n$ (undertermined setting) then one has

$$x_0 = A^+ y \quad \text{where} \quad A^+ \overset{\text{def.}}{=} A^\top (AA^\top)^{-1}.$$

In the special case $n = p$ and A is invertible, then both definitions of A^+ coincide, and $A^+ = A^{-1}$. In the general case (where A is neither injective nor surjective), A^+ can be computed using the Singular Values Decomposition (SVD). The matrix A^+ is often called the Moore-Penrose pseudo-inverse.

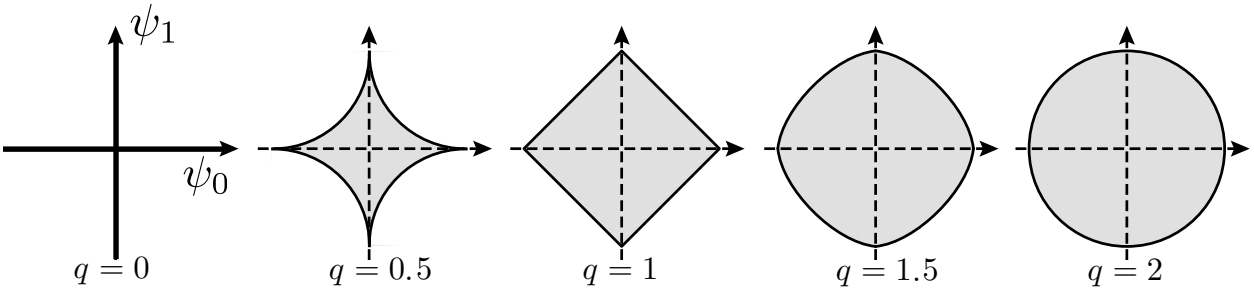


Figure 14.1: ℓ^q balls $\{x \,;\, \sum_k |x_k|^q \leqslant 1\}$ for varying q.

14.1.3 Lasso

The Lasso corresponds to using a ℓ^1 penalty

$$R(x) = \|x\|_1 \overset{\text{def.}}{=} \sum_{k=1}^{p} |x_k|.$$

The underlying idea is that solutions x_λ of a Lasso problem

$$x_\lambda \in \operatorname*{argmin}_{x \in \mathbb{R}^p} f_\lambda(x) = \frac{1}{2}\|Ax - y\|_{\mathbb{R}^n}^2 + \lambda \|x\|_1$$

are sparse, i.e. solutions x_λ (which might be non-unique) have many zero entries. To get some insight about this, Fig. 14.1 display the ℓ^q "balls" which shrink toward the axes as $q \to 0$ (thus enforcing more sparsity) but are non-convex for $q < 1$.

This can serve two purposes: (i) one knows before hand that the solution is expected to be sparse, which is the case for instance in some problems in imaging, (ii) one want to perform model selection by pruning some of the entries in the feature (to have simpler predictor, which can be computed more efficiently at test time, or that can be more interpretable). For typical ML problems though, the performance of the Lasso predictor is usually not better than the one obtained by Ridge.

Minimizing $f(x)$ is still a convex problem, but R is non-smooth, so that one cannot use a gradient descent. Section 14.1.4 shows how to modify the gradient descent to cope with this issue. In general, solutions x_λ cannot be computed in closed form, excepted when the design matrix A is orthogonal.

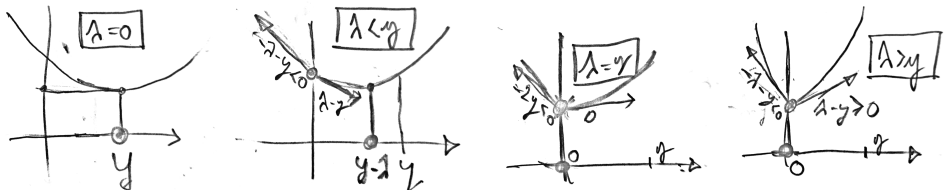


Figure 14.2: Evolution with λ of the function $F(x) \overset{\text{def.}}{=} \frac{1}{2}\|\cdot - y\|^2 + \lambda|\cdot|$.

Proposition 46. *When $n = p$ and $A = \mathrm{Id}_n$, one has*

$$\operatorname*{argmin}_{x \in \mathbb{R}^p} \frac{1}{2}\|x - y\|^2 + \lambda\|x_1\| = S_\lambda(x) \quad \textit{where} \quad S_\lambda(x) = (\operatorname{sign}(x_k)\max(|x_k| - \lambda, 0))_k$$

Proof. One has $f_\lambda(x) = \sum_k \frac{1}{2}(x_k - y_k)^2 + \lambda|x_k|$, so that one needs to find the minimum of the 1-D function $x \in \mathbb{R} \mapsto \frac{1}{2}(x-y)^2 + \lambda|x|$. We can do this minimization "graphically" as shown on Fig. 14.2. For $x > 0$, one has $F'(x) = x - y + \lambda$ wich is 0 at $x = y - \lambda$. The minimum is at $x = y - \lambda$ for $\lambda \leqslant y$, and stays at 0 for all $\lambda > y$. The problem is symmetric with respect to the switch $x \mapsto -x$. □

Here, S_λ is the celebrated soft-thresholding non-linear function.

14.1.4 Iterative Soft Thresholding

We now derive an algorithm using a classical technic of surrogate function minimization. We aim at minimizing

$$f(x) \overset{\text{def.}}{=} \frac{1}{2}\|y - Ax\|^2 + \lambda\|x\|_1$$

and we introduce for any fixed x' the function

$$f_\tau(x,x') \stackrel{\text{def.}}{=} f(x) - \frac{1}{2}\|Ax-Ax'\|^2 + \frac{1}{2\tau}\|x-x'\|^2.$$

We notice that $f_\tau(x,x)=0$ and one the quadratic part of this function reads

$$K(x,x') \stackrel{\text{def.}}{=} -\frac{1}{2}\|Ax-Ax'\|^2 + \frac{1}{2\tau}\|x-x'\| = \frac{1}{2}\langle\left(\frac{1}{\tau}\mathrm{Id}_N - A^\top A\right)(x-x'),\, x-x'\rangle.$$

This quantity $K(x,x')$ is positive if $\lambda_{\max}(A^\top A) \leqslant 1/\tau$ (maximum eigenvalue), i.e. $\tau \leqslant 1/\|A\|_{\text{op}}^2$, where we recall that $\|A\|_{\text{op}} = \sigma_{\max}(A)$ is the operator (algebra) norm. This shows that $f_\tau(x,x')$ is a valid surrogate functional, in the sense that

$$f(x) \leqslant f_\tau(x,x'), \quad f_\tau(x,x')=0, \quad \text{and} \quad f(\cdot) - f_\tau(\cdot,x') \text{ is smooth.}$$

We also note that this majorant $f_\tau(\cdot,x')$ is convex. This leads to define

$$x_{k+1} \stackrel{\text{def.}}{=} \operatorname*{argmin}_x f_\tau(x,x_k) \tag{14.6}$$

which by construction satisfies

$$f(x_{k+1}) \leqslant f(x_k).$$

Proposition 47. *The iterates x_k defined by (14.6) satisfy*

$$x_{k+1} = S_{\lambda\tau}\left(x_k - \tau A^\top(Ax_k - y)\right) \tag{14.7}$$

where $S_\lambda(x) = (s_\lambda(x_m))_m$ where $s_\lambda(r) = \mathrm{sign}(r)\max(|r|-\lambda,0)$ is the soft thresholding operator.

Proof. One has

$$\begin{aligned}
f_\tau(x,x') &= \frac{1}{2}\|Ax-y\|^2 - \frac{1}{2}\|Ax-Ax'\|^2 + \frac{1}{2\tau}\|x-x'\|^2 + \lambda\|x\|_1 \\
&= C + \frac{1}{2}\|Ax\|^2 - \frac{1}{2}\|Ax\|^2 + \frac{1}{2\tau}\|x\|^2 - \langle Ax,\, y\rangle + \langle Ax,\, Ax'\rangle - \frac{1}{\tau}\langle x,\, x'\rangle + \lambda\|x\|_1 \\
&= C + \frac{1}{2\tau}\|x\|^2 + \langle x,\, -A^\top y + AA^\top x' - \frac{1}{\tau}x'\rangle + \lambda\|x\|_1 \\
&= C' + \frac{1}{\tau}\left(\frac{1}{2}\|x-(x'-\tau A^\top(Ax'-y))\|^2 + \tau\lambda\|x\|_1.\right)
\end{aligned}$$

Proposition (46) shows that the minimizer of $f_\tau(x,x')$ is thus indeed $S_{\lambda\tau}(x'-\tau A^\top(Ax'-y))$ as claimed. □

Equation (14.7) defines the iterative soft-thresholding algorithm. It follows from a valid convex surrogate function if $\tau \leqslant 1/\|A\|^2$, but one can actually shows that it converges to a solution of the Lasso as soon as $\tau < 2/\|A\|^2$, which is exactly as for the classical gradient descent.

14.2 Stochastic Optimization

We detail some important stochastic Gradient Descent methods, which enable to perform optimization in the setting where the number of samples n is large and even infinite.

14.2.1 Minimizing Sums and Expectation

A large class of functionals in machine learning can be expressed as minimizing large sums of the form

$$\min_{x\in\mathbb{R}^p} f(x) \stackrel{\text{def.}}{=} \frac{1}{n}\sum_{i=1}^{n} f_i(x) \tag{14.8}$$

or even expectations of the form

$$\min_{x\in\mathbb{R}^p} f(x) \stackrel{\text{def.}}{=} \mathbb{E}_{\mathbf{z}\sim\pi}(f(x,\mathbf{z})) = \int_{\mathcal{Z}} f(x,z)\mathrm{d}\pi(z). \tag{14.9}$$

Problem (14.8) can be seen as a special case of (14.9), when using a discrete empirical uniform measure $\pi = \sum_{i=1}^n \delta_i$ and setting $f(x,i) = f_i(x)$. One can also viewed (14.8) as a discretized "empirical" version of (14.9) when drawing $(z_i)_i$ i.i.d. according to $\mathbf{z}$ and defining $f_i(x) = f(x,z_i)$. In this setup, (14.8) converges to (14.9) as $n \to +\infty$.

A typical example of such a class of problems is empirical risk minimization for linear model, where in these cases

$$f_i(x) = \ell(\langle a_i,\, x\rangle, y_i) \quad \text{and} \quad f(x,z) = \ell(\langle a,\, x\rangle, y) \tag{14.10}$$

for $z = (a,y) \in \mathcal{Z} = (\mathcal{A} = \mathbb{R}^p) \times \mathcal{Y}$ (typically $\mathcal{Y} = \mathbb{R}$ or $\mathcal{Y} = \{-1,+1\}$ for regression and classification), where ℓ is some loss function. We illustrate below the methods on binary logistic classification, where

$$L(s,y) \stackrel{\text{def.}}{=} \log(1+\exp(-sy)). \tag{14.11}$$

But this extends to arbitrary parametric models, and in particular deep neural networks.

While some algorithms (in particular batch gradient descent) are specific to finite sums (14.8), the stochastic methods we detail next work verbatim (with the same convergence guarantees) in the expectation case (14.9). For the sake of simplicity, we however do the exposition for the finite sums case, which is sufficient in the vast majority of cases. But one should keep in mind that n can be arbitrarily large, so it is not acceptable in this setting to use algorithms whose complexity per iteration depend on n.

If the functions $f_i(x)$ are very similar (the extreme case being that they are all equal), then of course there is a gain in using stochastic optimization (since in this case, $\nabla f_i \approx \nabla f$ but ∇f_i is n times cheaper). But in general stochastic optimization methods are notÂ necessarily faster than batch gradient descent. If n is not too large so that one afford the price of doing a few non-stochastic iterations, then deterministic methods can be faster. But if n is so large that one cannot do even a single deterministic iteration, then stochastic methods allow one to have a fine grained scheme by breaking the cost of determinstic iterations in smaller chunks. Another advantage is that they are quite easy to parallelize.

14.2.2 Batch Gradient Descent (BGD)

The usual deterministic (batch) gradient descent (BGD) is studied in details in Section 13.4. Its iterations read

$$x_{k+1} = x_k - \tau_k \nabla f(x_k)$$

and the step size should be chosen as $0 < \tau_{\min} < \tau_k < \tau_{\max} \stackrel{\text{def.}}{=} 2/L$ where L is the Lipschitz constant of the gradient ∇f. In particular, in this deterministic setting, this step size should not go to zero and this ensures quite fast convergence (even linear rates if f is strongly convex).

The computation of the gradient in our setting reads

$$\nabla f(x) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x) \tag{14.12}$$

so it typically has complexity $O(np)$ if computing ∇f_i has linear complexity in p.

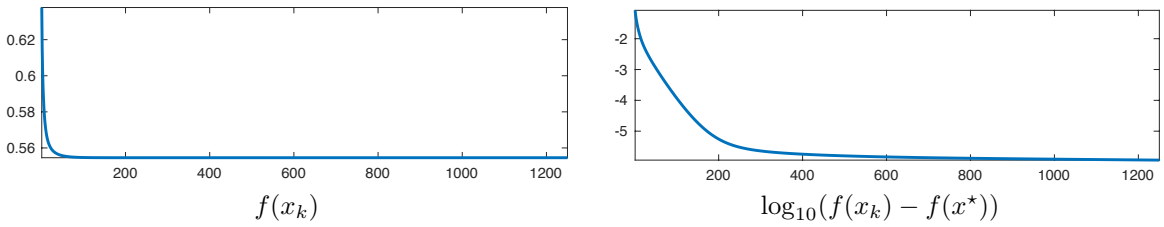


Figure 14.3: Evolution of the error of the BGD for logistic classification.

For ERM-type functions of the form (14.10), one can do the Taylor expansion of f_i

$$\begin{aligned} f_i(x+\varepsilon) &= \ell(\langle a_i,\, x\rangle + \langle a_i,\, \varepsilon\rangle, y_i) = \ell(\langle a_i,\, x\rangle, y_i) + \ell'(\langle a_i,\, x\rangle, y_i)\langle a_i,\, \varepsilon\rangle + o(\|\varepsilon\|) \\ &= f_i(x) + \langle \ell'(\langle a_i,\, x\rangle, y_i)a_i,\, x\rangle + o(\|\varepsilon\|), \end{aligned}$$

where $\ell(y, y') \in \mathbb{R}$ is the derivative with respect to the first variable, i.e. the gradient of the map $y \in \mathbb{R} \mapsto L(y, y') \in \mathbb{R}$. This computation shows that

$$\nabla f_i(x) = \ell'(\langle a_i,\, x\rangle, y_i)a_i. \tag{14.13}$$

For the logistic loss, one has

$$L'(s, y) = -s\frac{e^{-sy}}{1 + e^{-sy}}.$$

14.2.3 Stochastic Gradient Descent (SGD)

For very large n, computing the full gradient ∇f as in (14.12) is prohibitive. The idea of SGD is to trade this exact full gradient by an inexact proxy using a single functional f_i where i is drawn uniformly at random. The main idea that makes this work is that this sampling scheme provides an unbiased estimate of the gradient, in the sense that

$$\mathbb{E}_{\mathbf{i}} \nabla f_{\mathbf{i}}(x) = \nabla f(x) \tag{14.14}$$

where $\mathbf{i}$ is a random variable distributed uniformly in $\{1, \ldots, n\}$.

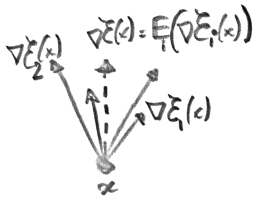


Figure 14.4: Unbiased gradient estimate

Starting from some x_0,the iterations of stochastic gradient descent (SGD) read

$$x_{k+1} = x_k - \tau_k \nabla f_{i(k)}(x_k)$$

where, for each iteration index k, $i(k)$ is drawn uniformly at random in $\{1, \ldots, n\}$. It is important that the iterates x_{k+1} are thus random vectors, and the theoretical analysis of the method thus studies wether this sequence of random vectors converges (in expectation or in probability for instance) toward a deterministic vector (minimizing f), and at which speed.

Note that each step of a batch gradient descent has complexity $O(np)$, while a step of SGD only has complexity $O(p)$. SGD is thus advantageous when n is very large, and one cannot afford to do several passes through the data. In some situation, SGD can provide accurate results even with $k \ll n$, exploiting redundancy between the samples.

A crucial question is the choice of step size schedule τ_k. It must tends to 0 in order to cancel the noise induced on the gradient by the stochastic sampling. But it should not go too fast to zero in order for the method to keep converging.

A typical schedule that ensures both properties is to have asymptotically $\tau_k \sim k^{-1}$ for $k \to +\infty$. We thus propose to use

$$\tau_k \overset{\text{def.}}{=} \frac{\tau_0}{1 + k/k_0} \tag{14.15}$$

Figure 14.5: Schematic view of SGD iterates

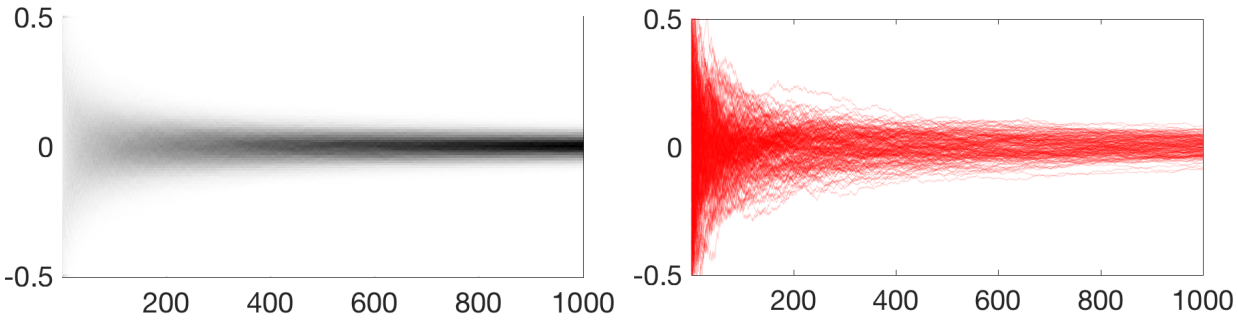


Figure 14.6: Display of a large number of trajectories $k \mapsto x_k \in \mathbb{R}$ generated by several runs of SGD. On the top row, each curve is a trajectory, and the bottom row displays the corresponding density.

where k_0 indicates roughly the number of iterations serving as a "warmup" phase.

Figure 14.6 shows a simple 1-D example to minimize $f_1(x) + f_2(x)$ for $x \in \mathbb{R}$ and $f_1(x) = (x-1)^2$ and $f_2(x) = (x+1)^2$. One can see how the density of the distribution of x_k progressively clusters around the minimizer $x^\star = 0$. Here the distribution of x_0 is uniform on $[-1/2, 1/2]$.

The following theorem shows the convergence in expectation with a $1/\sqrt{k}$ rate on the objective.

Theorem 24. *We assume f is μ-strongly convex as defined in ($\mathcal{S}_\mu$) (i.e. $\mathrm{Id}_p \preceq \partial^2 f(x)$ if f is $\mathcal{C}^2$), and is such that $\|\nabla f_i(x)\|^2 \leqslant C^2$. For the step size choice $\tau_k = \frac{1}{\mu(k+1)}$, one has*

$$\mathbb{E}(\|x_k - x^\star\|^2) \leqslant \frac{R}{k+1} \quad \textit{where} \quad R = \max(\|x_0 - x^\star\|, C^2/\mu^2), \tag{14.16}$$

where $\mathbb{E}$ indicates an expectation with respect to the i.i.d. sampling performed at each iteration.

Proof. By strong convexity, one has

$$f(x^\star) - f(x_k) \geqslant \langle \nabla f(x_k),\, x^\star - x_k \rangle + \frac{\mu}{2}\|x_k - x^\star\|^2$$
$$f(x_k) - f(x^\star) \geqslant \langle \nabla f(x^\star),\, x_k - x^\star \rangle + \frac{\mu}{2}\|x_k - x^\star\|^2.$$

Summing these two inequalities and using $\nabla f(x^\star) = 0$ leads to

$$\langle \nabla f(x_k) - \nabla f(x^\star),\, x_k - x^\star \rangle = \langle \nabla f(x_k),\, x_k - x^\star \rangle \geqslant \mu \|x_k - x^\star\|^2. \tag{14.17}$$

Considering only the expectation with respect to the ransom sample of $i(k) \sim \mathbf{i}_k$, one has

$$\begin{aligned}
\mathbb{E}_{\mathbf{i}_k}(\|x_{k+1} - x^\star\|^2) &= \mathbb{E}_{\mathbf{i}_k}(\|x_k - \tau_k \nabla f_{\mathbf{i}_k}(x_k) - x^\star\|^2) \\
&= \|x_k - x^\star\|^2 + 2\tau_k \langle \mathbb{E}_{\mathbf{i}_k}(\nabla f_{\mathbf{i}_k}(x_k)),\, x^\star - x_k \rangle + \tau_k^2 \mathbb{E}_{\mathbf{i}_k}(\|\nabla f_{\mathbf{i}_k}(x_k)\|^2) \\
&\leqslant \|x_k - x^\star\|^2 + 2\tau_k \langle \nabla f(x_k)),\, x^\star - x_k \rangle + \tau_k^2 C^2
\end{aligned}$$

where we used the fact (14.14) that the gradient is unbiased. Taking now the full expectation with respect to all the other previous iterates, and using (14.17) one obtains

$$\mathbb{E}(\|x_{k+1} - x^\star\|^2) \leqslant \mathbb{E}(\|x_k - x^\star\|^2) - 2\mu\tau_k \mathbb{E}(\|x_k - x^\star\|^2) + \tau_k^2 C^2 = (1 - 2\mu\tau_k)\mathbb{E}(\|x_k - x^\star\|^2) + \tau_k^2 C^2. \tag{14.18}$$

We show by recursion that the bound (14.16) holds. We denote $\varepsilon_k \overset{\text{def.}}{=} \mathbb{E}(\|x_k - x^\star\|^2)$. Indeed, for $k = 0$, this it is true that

$$\varepsilon_0 \leqslant \frac{\max(\|x_0 - x^\star\|, C^2/\mu^2)}{1} = \frac{R}{1}.$$

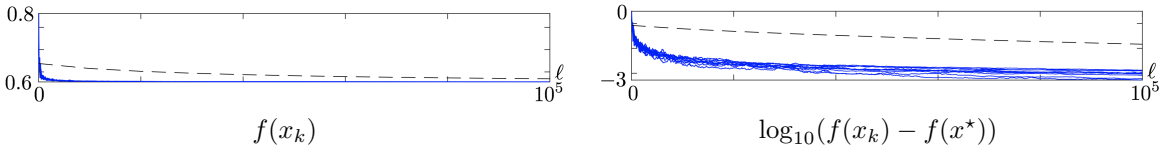


Figure 14.7: Evolution of the error of the SGD for logistic classification (dashed line shows BGD).

We now assume that $\varepsilon_k \leqslant \frac{R}{k+1}$. Using (14.18) in the case of $\tau_k = \frac{1}{\mu(k+1)}$, one has, denoting $m = k+1$

$$\begin{aligned}\varepsilon_{k+1} &\leqslant (1 - 2\mu\tau_k)\varepsilon_k + \tau_k^2 C^2 = \left(1 - \frac{2}{m}\right)\varepsilon_k + \frac{C^2}{(\mu m)^2} \\ &\leqslant \left(1 - \frac{2}{m}\right)\frac{R}{m} + \frac{R}{m^2} = \left(\frac{1}{m} - \frac{1}{m^2}\right)R = \frac{m-1}{m^2}R = \frac{m^2-1}{m^2}\frac{1}{m+1}R \leqslant \frac{R}{m+1}\end{aligned}$$

□

A weakness of SGD (as well as the SGA scheme studied next) is that it only weakly benefit from strong convexity of f. This is in sharp contrast with BGD, which enjoy a fast linear rate for strongly convex functionals, see Theorem 23.

Figure 14.7 displays the evolution of the energy $f(x_k)$. It overlays on top (black dashed curve) the convergence of the batch gradient descent, with a careful scaling of the number of iteration to account for the fact that the complexity of a batch iteration is n times larger.

14.2.4 Stochastic Gradient Descent with Averaging (SGA)

Stochastic gradient descent is slow because of the fast decay of τ_k toward zero. To improve somehow the convergence speed, it is possible to average the past iterate, i.e. run a "classical" SGD on auxiliary variables $(\tilde{x}_k)_k$

$$\tilde{x}^{(\ell+1)} = \tilde{x}_k - \tau_k \nabla f_{i(k)}(\tilde{x}_k)$$

and output as estimated weight vector the Cesaro average

$$x_k \overset{\text{def.}}{=} \frac{1}{k}\sum_{\ell=1}^{k} \tilde{x}_\ell.$$

This defines the Stochastic Gradient Descent with Averaging (SGA) algorithm.

Note that it is possible to avoid explicitly storing all the iterates by simply updating a running average as follow

$$x_{k+1} = \frac{1}{k}\tilde{x}_k + \frac{k-1}{k}x_k.$$

In this case, a typical choice of decay is rather of the form

$$\tau_k \overset{\text{def.}}{=} \frac{\tau_0}{1 + \sqrt{k/k_0}}.$$

Notice that the step size now goes much slower to 0, at rate $k^{-1/2}$.

Typically, because the averaging stabilizes the iterates, the choice of (k_0, τ_0) is less important than for SGD.

Bach proves that for logistic classification, it leads to a faster convergence (the constant involved are smaller) than SGD, since on contrast to SGD, SGA is adaptive to the local strong convexity of E.

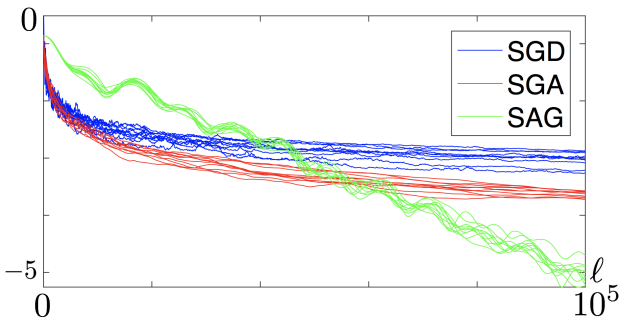


Figure 14.8: Evolution of $\log_{10}(f(x_k) - f(x^\star))$ for SGD, SGA and SAG.

14.2.5 Stochastic Averaged Gradient Descent (SAG)

For problem size n where the dataset (of size $n \times p$) can fully fit into memory, it is possible to further improve the SGA method by bookkeeping the previous gradients. This gives rise to the Stochastic Averaged Gradient Descent (SAG) algorithm.

We store all the previously computed gradients in $(G^i)_{i=1}^n$, which necessitates $O(n \times p)$ memory. The iterates are defined by using a proxy g for the batch gradient, which is progressively enhanced during the iterates.

The algorithm reads

$$x_{k+1} = x_k - \tau g \quad \text{where} \quad \left\{ \begin{array}{l} h \leftarrow \nabla f_{i(k)}(\tilde{x}_k), \\ g \leftarrow g - G^{i(k)} + h, \\ G^{i(k)} \leftarrow h. \end{array} \right.$$

Note that in contrast to SGD and SGA, this method uses a fixed step size τ. Similarly to the BGD, in order to ensure convergence, the step size τ should be of the order of $1/L$ where L is the Lipschitz constant of f.

This algorithm improves over SGA and SGD since it has a convergence rate of $O(1/k)$ as does BGD. Furthermore, in the presence of strong convexity (for instance when X is injective for logistic classification), it has a linear convergence rate, i.e.

$$\mathbb{E}(f(x_k)) - f(x^\star) = O\left(\rho^k\right),$$

for some $0 < \rho < 1$.

Note that this improvement over SGD and SGA is made possible only because SAG explicitly uses the fact that n is finite (while SGD and SGA can be extended to infinite n and more general minimization of expectations (14.9)).

Figure 14.8 shows a comparison of SGD, SGA and SAG.

14.3 Automatic Differentiation

The main computational bottleneck of gradient descent methods (batch or stochastic) is the computation of gradients $\nabla f(x)$. For simple functionals, such as those encountered in ERM for linear models, and also for MLP with a single hidden layer, it is possible to compute these gradients in closed form, and that the main computational burden is the evaluation of matrix-vector products. For more complicated functionals (such as those involving deep networks), computing the formula for the gradient quickly becomes cumbersome. Even worse: computing these gradients using the usual chain rule formula is sub-optimal. We presents methods to compute recursively in an optimal manner these gradients. The purpose of this approach is to automatize this computational step.

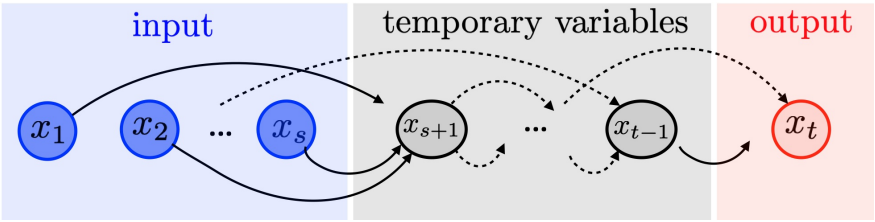


Figure 14.9: A computational graph.

14.3.1 Finite Differences and Symbolic Calculus

We consider $f : \mathbb{R}^p \to \mathbb{R}$ and want to derive a method to evaluate $\nabla f : \mathbb{R}^p \mapsto \mathbb{R}^p$. Approximating this vector field using finite differences, i.e. introducing $\varepsilon > 0$ small enough and computing

$$\frac{1}{\varepsilon}(f(x+\varepsilon\delta_1) - f(x), \ldots, f(x+\varepsilon\delta_p) - f(x))^\top \approx \nabla f(x)$$

requires $p+1$ evaluations of f, where we denoted $\delta_k = (0, \ldots, 0, 1, 0, \ldots, 0)$ where the 1 is at index k. For a large p, this is prohibitive. The method we describe in this section (the so-called reverse mode automatic differentiation) has in most cases a cost proportional to a single evaluation of f. This type of method is similar to symbolic calculus in the sense that it provides (up to machine precision) exact gradient computation. But symbolic calculus does not takes into account the underlying algorithm which compute the function, while automatic differentiation factorizes the computation of the derivative according to an efficient algorithm.

14.3.2 Computational Graphs

We consider a generic function $f(x)$ where $x = (x_1, \ldots, x_s)$ are the input variables. We assume that f is implemented in an algorithm, with intermediate variable $(x_{s+1}, \ldots, x_t)$ where t is the total number of variables. The output is x_t, and we thus denote $x_t = f(x)$ this function. We denote $x_k \in \mathbb{R}^{n_k}$ the dimensionality of the variables. The goal is to compute the derivatives $\frac{\partial f(x)}{\partial x_k} \in \mathbb{R}^{n_t \times n_k}$ for $k = 1, \ldots, s$. For the sake of simplicity, one can assume in what follows that $n_k = 1$ so that all the involved quantities are scalar (but if this is not the case, beware that the order of multiplication of the matrices of course matters).

A numerical algorithm can be represented as a succession of functions of the form

$$\forall\, k = s+1, \ldots, t, \quad x_k = f_k(x_1, \ldots, x_{k-1})$$

where f_k is a function which only depends on the previous variables, see Fig. 14.9. One can represent this algorithm using a directed acyclic graph (DAG), linking the variables involved in f_k to x_k. The node of this graph are thus conveniently ordered by their indexing, and the directed edges only link a variable to another one with a strictly larger index. The evaluation of $f(x)$ thus corresponds to a forward traversal of this graph. Note that the goal of automatic differentiation is not to define an efficient computational graph, it is up to the user to provide this graph. Computing an efficient graph associated to a mathematical formula is a complicated combinatorial problem, which still has to be solved by the user. Automatic differentiation thus leverage the availability of an efficient graph to provide an efficient algorithm to evaluate derivatives.

14.3.3 Forward Mode of Automatic Differentiation

The forward mode correspond to the usual way of computing differentials. It compute the derivative $\frac{\partial x_k}{\partial x_1}$ of all variables x_k with respect to x_1. One then needs to repeat this method p times to compute all the derivative with respect to $x_1, x_2, \ldots, x_p$ (we only write thing for the first variable, the method being of course the same with respect to the other ones).

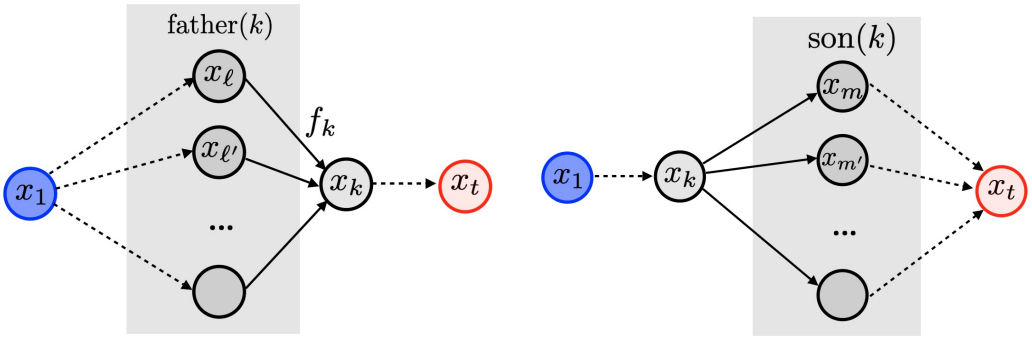


Figure 14.10: Relation between the variable for the forward (left) and backward (right) modes.

The method initialize the derivative of the input nodes

$$\frac{\partial x_1}{\partial x_1} = \mathrm{Id}_{n_1 \times n_1}, \quad \frac{\partial x_2}{\partial x_1} = 0_{n_2 \times n_1}, \ldots, \quad \frac{\partial x_s}{\partial x_1} = 0_{n_s \times n_1},$$

(and thus 1 and 0's for scalar variables), and then iteratively make use of the following recursion formula

$$\forall\, k = s+1, \ldots, t, \quad \frac{\partial x_k}{\partial x_1} = \sum_{\ell \in \mathrm{parent}(k)} \left[\frac{\partial x_k}{\partial x_\ell}\right] \times \frac{\partial x_\ell}{\partial x_1} = \sum_{\ell \in \mathrm{parent}(k)} \frac{\partial f_k}{\partial x_\ell}(x_1, \ldots, x_{k-1}) \times \frac{\partial x_\ell}{\partial x_1}.$$

The notation "parent(k)" denotes the nodes $\ell < k$ of the graph that are connected to k, see Figure 14.10, left. Here the quantities being computed (i.e. stored in computer variables) are the derivatives $\frac{\partial x_\ell}{\partial x_1}$, and $\times$ denotes in full generality matrix-matrix multiplications. We have put in $[\ldots]$ an informal notation, since here $\frac{\partial x_k}{\partial x_\ell}$ should be interpreted not as a numerical variable but needs to be interpreted as derivative of the function f_k, which can be evaluated on the fly (we assume that the derivative of the function involved are accessible in closed form).

Assuming all the involved functions $\frac{\partial f_k}{\partial x_k}$ have the same complexity (which is likely to be the case if all the n_k are for instance scalar or have the same dimension), and that the number of parent node is bounded, one sees that the complexity of this scheme is p times the complexity of the evaluation of f (since this needs to be repeated p times for $\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_p}$). For a large p, this is prohibitive.

Simple example. We consider the fonction

$$f(x, y) = y \log(x) + \sqrt{y \log(x)} \tag{14.19}$$

whose computational graph is displayed on Figure 14.11. The iterations of the forward mode to compute the derivative with respect to x read

$$\begin{aligned}
&\frac{\partial x}{\partial x} = 1, \quad \frac{\partial y}{\partial x} = 0 \\
&\frac{\partial a}{\partial x} = \left[\frac{\partial a}{\partial x}\right] \frac{\partial x}{\partial x} = \frac{1}{x} \frac{\partial x}{\partial x} && \{x \mapsto a = \log(x)\} \\
&\frac{\partial b}{\partial x} = \left[\frac{\partial b}{\partial a}\right] \frac{\partial a}{\partial x} + \left[\frac{\partial b}{\partial y}\right] \frac{\partial y}{\partial x} = y \frac{\partial a}{\partial x} + 0 && \{(y, a) \mapsto b = ya\} \\
&\frac{\partial c}{\partial x} = \left[\frac{\partial c}{\partial b}\right] \frac{\partial b}{\partial x} = \frac{1}{2\sqrt{b}} \frac{\partial b}{\partial x} && \{b \mapsto c = \sqrt{b}\} \\
&\frac{\partial f}{\partial x} = \left[\frac{\partial f}{\partial b}\right] \frac{\partial b}{\partial x} + \left[\frac{\partial f}{\partial c}\right] \frac{\partial c}{\partial x} = 1 \frac{\partial b}{\partial x} + 1 \frac{\partial c}{\partial x} && \{(b, c) \mapsto f = b + c\}
\end{aligned}$$

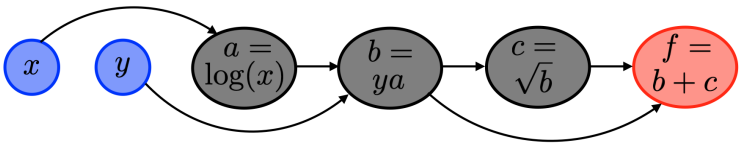


Figure 14.11: Example of a simple computational graph.

One needs to run another forward pass to compute the derivative with respect to y

$$\begin{aligned}
&\frac{\partial x}{\partial y} = 0, \quad \frac{\partial y}{\partial y} = 1 \\
&\frac{\partial a}{\partial y} = \left[\frac{\partial a}{\partial x}\right]\frac{\partial x}{\partial y} = 0 && \{x \mapsto a = \log(x)\} \\
&\frac{\partial b}{\partial y} = \left[\frac{\partial b}{\partial a}\right]\frac{\partial a}{\partial y} + \left[\frac{\partial b}{\partial y}\right]\frac{\partial y}{\partial y} = 0 + a\frac{\partial y}{\partial y} && \{(y,a) \mapsto b = ya\} \\
&\frac{\partial c}{\partial y} = \left[\frac{\partial c}{\partial b}\right]\frac{\partial b}{\partial y} = \frac{1}{2\sqrt{b}}\frac{\partial b}{\partial y} && \{b \mapsto c = \sqrt{b}\} \\
&\frac{\partial f}{\partial y} = \left[\frac{\partial f}{\partial b}\right]\frac{\partial b}{\partial y} + \left[\frac{\partial f}{\partial c}\right]\frac{\partial c}{\partial y} = 1\frac{\partial b}{\partial y} + 1\frac{\partial c}{\partial y} && \{(b,c) \mapsto f = b + c\}
\end{aligned}$$

Dual numbers. A convenient way to implement this forward pass is to make use of so called "dual number", which is an algebra over the real where the number have the form $x + \varepsilon x'$ where ε is a symbol obeying the rule that $\varepsilon^2 = 0$. Here $(x, x') \in \mathbb{R}^2$ and x' is intended to store a derivative with respect to some input variable. These number thus obeys the following arithmetic operations

$$(x + \varepsilon x')(y + \varepsilon y') = xy + \varepsilon(xy' + yx') \quad \text{and} \quad \frac{1}{x + \varepsilon x'} = \frac{1}{x} - \varepsilon\frac{x'}{x^2}.$$

If f is a polynomial or a rational function, from these rules one has that

$$f(x + \varepsilon) = f(x) + \varepsilon f'(x).$$

For a more general basic function f, one needs to overload it so that

$$f(x + \varepsilon x') \stackrel{\text{def.}}{=} f(x) + \varepsilon f'(x)x'.$$

Using this definition, one has that

$$(f \circ g)(x + \varepsilon) = f(g(x)) + \varepsilon f'(g(x))g'(x)$$

which corresponds to the usual chain rule. More generally, if $f(x_1, \ldots, x_s)$ is a function implemented using these overloaded basic functions, one has

$$f(x_1 + \varepsilon, x_2, \ldots, x_s) = f(x_1, \ldots, x_s) + \varepsilon\frac{\partial f}{\partial x_1}(x_1, \ldots, x_s)$$

and this evaluation is equivalent to applying the forward mode of automatic differentiation to compute $\frac{\partial f}{\partial x_1}(x_1, \ldots, x_s)$ (and similarly for the other variables).

14.3.4 Reverse Mode of Automatic Differentiation

Instead of evaluating the differentials $\frac{\partial x_k}{\partial x_1}$ which is problematic for a large p, the reverse mode evaluates the differentials $\frac{\partial x_t}{\partial x_k}$, i.e. it computes the derivative of the output node with respect to the all the inner nodes.

The method initialize the derivative of the final node

$$\frac{\partial x_t}{\partial x_t} = \mathrm{Id}_{n_t \times n_t},$$

and then iteratively makes use, from the last node to the first, of the following recursion formula

$$\forall\, k = t-1, t-2, \ldots, 1, \quad \frac{\partial x_t}{\partial x_k} = \sum_{m \in \mathrm{son}(k)} \frac{\partial x_t}{\partial x_m} \times \left[\frac{\partial x_m}{\partial x_k}\right] = \sum_{m \in \mathrm{son}(k)} \frac{\partial x_t}{\partial x_m} \times \frac{\partial f_m(x_1, \ldots, x_m)}{\partial x_k}.$$

The notation "parent(k)" denotes the nodes $\ell < k$ of the graph that are connected to k, see Figure 14.10, right.

Back-propagation. In the special case where $x_t \in \mathbb{R}$, then $\frac{\partial x_t}{\partial x_k} = [\nabla_{x_k} f(x)]^\top \in \mathbb{R}^{1 \times n_k}$ and one can write the recursion on the gradient vector as follow

$$\forall\, k = t-1, t-2, \ldots, 1, \quad \nabla_{x_k} f(x) = \sum_{m \in \mathrm{son}(k)} \left(\frac{\partial f_m(x_1, \ldots, x_m)}{\partial x_k}\right)^\top (\nabla_{x_m} f(x)).$$

where $\left(\frac{\partial f_m(x_1,\ldots,x_m)}{\partial x_k}\right)^\top \in \mathbb{R}^{n_k \times n_m}$ is the adjoint of the Jacobian of f_m. This form of recursion using adjoint is often referred to as "back-propagation", and is the most frequent setting in applications to ML.

In general, when $n_t = 1$, the backward is the optimal way to compute the gradient of a function. Its drawback is that it necessitate the pre-computation of all the intermediate variables $(x_k)_{k=p}^t$, which can be prohibitive in term of memory usage when t is large. There exists check-pointing method to alleviate this issue, but it is out of the scope of this course.

Simple example. We consider once again the fonction $f(x)$ of (14.19), the iterations of the reverse mode read

$$\begin{aligned}
&\frac{\partial f}{\partial f} = 1 \\
&\frac{\partial f}{\partial c} = \frac{\partial f}{\partial f}\left[\frac{\partial f}{\partial c}\right] = \frac{\partial f}{\partial f} 1 && \{c \mapsto f = b + c\} \\
&\frac{\partial f}{\partial b} = \frac{\partial f}{\partial c}\left[\frac{\partial c}{\partial b}\right] + \frac{\partial f}{\partial f}\left[\frac{\partial f}{\partial b}\right] = \frac{\partial f}{\partial c}\frac{1}{2\sqrt{b}} + \frac{\partial f}{\partial f} 1 && \{b \mapsto c = \sqrt{b}, b \mapsto f = b + c\} \\
&\frac{\partial f}{\partial a} = \frac{\partial f}{\partial b}\left[\frac{\partial b}{\partial a}\right] = \frac{\partial f}{\partial b} y && \{a \mapsto b = ya\} \\
&\frac{\partial f}{\partial y} = \frac{\partial f}{\partial b}\left[\frac{\partial b}{\partial y}\right] = \frac{\partial f}{\partial b} a && \{y \mapsto b = ya\} \\
&\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a}\left[\frac{\partial a}{\partial x}\right] = \frac{\partial f}{\partial a}\frac{1}{x} && \{x \mapsto a = \log(x)\}
\end{aligned}$$

The advantage of the reverse mode is that a single traversal of the computational graph allows to compute both derivatives with respect to x, y, while the forward more necessitates two passes.

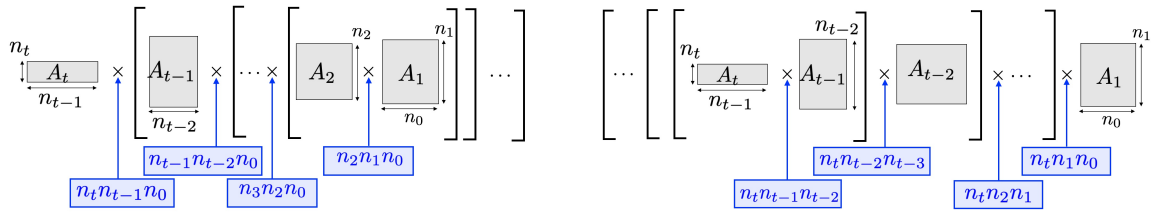


Figure 14.12: Complexity of forward (left) and backward (right) modes for composition of functions.

14.3.5 Feed-forward Compositions

The simplest computational graphs are purely feedforward, and corresponds to the computation of

$$f = f_t \circ f_{t-1} \circ \ldots \circ f_2 \circ f_1 \tag{14.20}$$

for functions $f_k : \mathbb{R}^{n_{k-1}} \to \mathbb{R}^{n_k}$.

The forward function evaluation algorithm initializes $x_0 = x \in \mathbb{R}^{n_0}$ and then computes

$$\forall\, k = 1, \ldots, t, \quad x_k = f_k(x_{k-1})$$

where at the output, one retrieves $f(x) = x_t$.

Denoting $A_k \overset{\text{def.}}{=} \partial f_k(x_{k-1}) \in \mathbb{R}^{n_k \times n_{k-1}}$ the Jacobian, one has

$$\partial f(x) = A_t \times A_{t-1} \times \ldots A_2 \times A_1.$$

The forward (resp. backward) mode corresponds to the computation of the product of the Jacobian from right to left (resp. left to right)

$$\partial f(x) = A_t \times (A_{t-1} \times (\ldots \times (A_3 \times (A_2 \times A_1)))),$$
$$\partial f(x) = ((((A_t \times A_{t-1}) \times A_{t-2}) \times \ldots) \times A_2) \times A_1.$$

We note that the computation of the product $A \times B$ of $A \in \mathbb{R}^{n \times p}$ with $B \in \mathbb{R}^{p \times q}$ necessitates npq operations. As shown on Figure 14.12, the complexity of the forward and backward modes are

$$n_0 \sum_{k=1}^{t-1} n_k n_{k+1} \quad \text{and} \quad n_t \sum_{k=0}^{t-2} n_k n_{k+1}$$

So if $n_t \ll n_0$ (which is the typical case in ML scenario where $n_t = 1$) then the backward mode is cheaper.

14.3.6 Feed-forward Architecture

We can generalize the previous example to account for feed-forward architectures, such as neural networks, which are of the form

$$\forall\, k = 1, \ldots, t, \quad x_k = f_k(x_{k-1}, \theta_{k-1}) \tag{14.21}$$

where θ_{k-1} is a vector of parameters and $x_0 \in \mathbb{R}^{n_0}$ is given. The function to minimize has the form

$$f(\theta) \overset{\text{def.}}{=} L(x_t) \tag{14.22}$$

where $L : \mathbb{R}^{n_t} \to \mathbb{R}$ is some loss function (for instance a least square or logistic prediction risk) and $\theta = (\theta_k)_{k=0}^{t-1}$. Figure 14.13, top, displays the associated computational graph.

One can use the reverse mode automatic differentiation to compute the gradient of f by computing successively the gradient with respect to all (x_k, θ_k). One initializes

$$\nabla_{x_t} f = \nabla L(x_t)$$

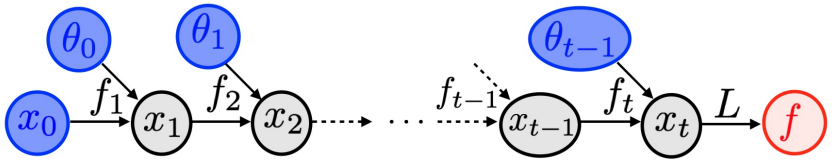


Figure 14.13: Computational graph for a feedforward architecture.

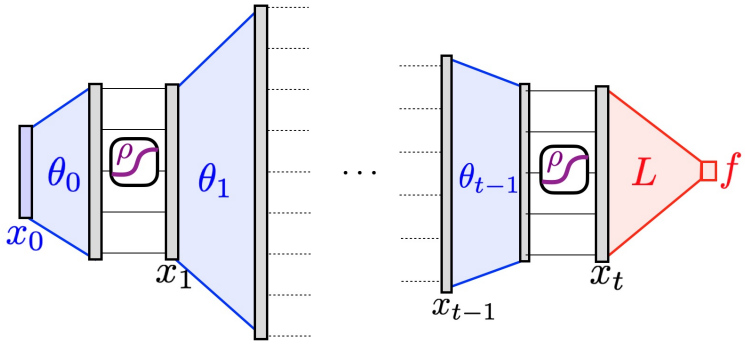


Figure 14.14: Multi-layer perceptron parameterization.

and then recurse from $k = t-1$ to 0

$$z_{k-1} = [\partial_x f_k(x_{k-1}, \theta_{k-1})]^\top z_k \quad \text{and} \quad \nabla_{\theta_{k-1}} f = [\partial_\theta f_k(x_{k-1}, \theta_{k-1})]^\top (\nabla_{x_k} f) \tag{14.23}$$

where we denoted $z_k \overset{\text{def.}}{=} \nabla_{x_k} f(\theta)$ the gradient with respect to x_k.

Multilayers perceptron. For instance, feedforward deep network (fully connected for simplicity) corresponds to using

$$\forall\, x_{k-1} \in \mathbb{R}^{n_{k-1}}, \quad f_k(x_{k-1}, \theta_{k-1}) = \rho(\theta_{k-1} x_{k-1}) \tag{14.24}$$

where $\theta_{k-1} \in \mathbb{R}^{n_k \times n_{k-1}}$ are the neuron's weights and ρ a fixed pointwise linearity, see Figure 14.14. One has, for a vector $z_k \in \mathbb{R}^{n_k}$ (typically equal to $\nabla_{x_k} f$)

$$\begin{cases} [\partial_x f_k(x_{k-1}, \theta_{k-1})]^\top (z_k) = \theta_{k-1}^\top w_k z_k, \\ [\partial_\theta f_k(x_{k-1}, \theta_{k-1})]^\top (z_k) = w_k x_{k-1}^\top \end{cases} \quad \text{where} \quad w_k \overset{\text{def.}}{=} \operatorname{diag}(\rho'(\theta_{k-1} x_{k-1})).$$

Link with adjoint state method. One can interpret (14.21) as a time discretization of a continuous ODE. One imposes that the dimension $n_k = n$ is fixed, and denotes $x(t) \in \mathbb{R}^n$ a continuous time evolution, so that $x_k \to x(k\tau)$ when $k \to +\infty$ and $k\tau \to t$. Imposing then the structure

$$f_k(x_{k-1}, \theta_{k-1}) = x_{k-1} + \tau u(x_{k-1}, \theta_{k-1}, k\tau) \tag{14.25}$$

where $u(x, \theta, t) \in \mathbb{R}^n$ is a parameterized vector field, as $\tau \to 0$, one obtains the non-linear ODE

$$\dot{x}(t) = u(x(t), \theta(t), t) \tag{14.26}$$

with $x(t=0) = x_0$.

Denoting $z(t) = \nabla_{x(t)} f(\theta)$ the "adjoint" vector field, the discrete equations (14.28) becomes the so-called adjoint equations, which is a linear ODE

$$\dot{z}(t) = -[\partial_x u(x(t), \theta(t), t)]^\top z(t) \quad \text{and} \quad \nabla_{\theta(t)} f(\theta) = [\partial_\theta u(x(t), \theta(t), t)]^\top z(t).$$

Note that the correct normalization is $\frac{1}{\tau} \nabla_{\theta_{k-1}} f \to \nabla_{\theta(t)} f(\theta)$

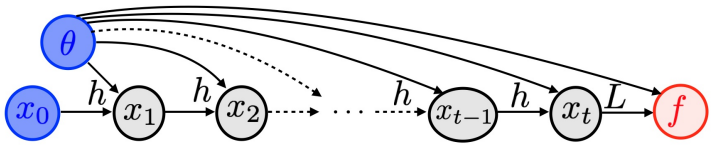


Figure 14.15: Computational graph for a recurrent architecture.

14.3.7 Recurrent Architectures

Parametric recurrent functions are obtained by using the same parameter $\theta = \theta_k$ and $f_k = h$ recursively in (14.24), so that

$$\forall\, k = 1, \ldots, t, \quad x_k = h(x_{k-1}, \theta). \tag{14.27}$$

We consider a real valued function of the form

$$f(\theta) = L(x_t, \theta)$$

so that here the final loss depends on θ (which is thus more general than (14.22)). Figure 14.15, bottom, displays the associated computational graph.

The back-propagation then operates as

$$\nabla_{x_{k-1}} f = [\partial_x h(x_{k-1}, \theta)]^\top \nabla_{x_k} f \quad \text{and} \quad \nabla_\theta f = \nabla_\theta L(x_t, \theta) + \sum_k [\partial_\theta h(x_{k-1}, \theta)]^\top \nabla_{x_k} f. \tag{14.28}$$

Similarly, writing $h(x, \theta) = x + \tau u(x, \theta)$, letting $(k, k\tau) \to (+\infty, t)$, one obtains the forward non-linear ODE with a time-stationary vector field

$$\dot{x}(t) = u(x(t), \theta)$$

and the following linear backward adjoint equation, for $f(\theta) = L(x(T), \theta)$

$$\dot{z}(t) = -[\partial_x u(x(t), \theta)]^\top z(t) \quad \text{and} \quad \nabla_\theta f(\theta) = \nabla_\theta L(x(T), \theta) + \int_0^T [\partial_\theta f(x(t), \theta)]^\top z(t) \mathrm{d}t. \tag{14.29}$$

with $z(0) = \nabla_x L(x_t, \theta)$.

Residual recurrent networks. A recurrent network is defined using

$$h(x, \theta) = x + W_2^\top \rho(W_1 x)$$

as displayed on Figure 14.16, where $\theta = (W_1, W_2) \in (\mathbb{R}^{n \times q})^2$ are the weights and ρ is a pointwise non-linearity. The number q of hidden neurons can be increased to approximate more complex functions. In the special case where $W_2 = -\tau W_1$, and $\rho = \psi'$, then this is a special case of an argmin layer (14.31) to minimize the function $\mathcal{E}(x, \theta) = \psi(W_1 x)$ using gradient descent, where $\psi(u) = \sum_i \psi(u_i)$ is a separable function. The Jacobians $\partial_\theta h$ and $\partial_x h$ are computed similarly to (14.29).

Mitigating memory requirement. The main issue of applying this backpropagation method to compute $\nabla f(\theta)$ is that it requires a large memory to store all the iterates $(x_k)_{k=0}^t$. A workaround is to use checkpointing, which stores some of these intermediate results and re-run partially the forward algorithm to reconstruct missing values during the backward pass. Clever hierarchical method perform this recursively in order to only require $\log(t)$ stored values and a $\log(t)$ increase on the numerical complexity.

In some situation, it is possible to avoid the storage of the forward result, if one assume that the algorithm can be run backward. This means that there exists some functions g_k so that

$$x_k = g_k(x_{k+1}, \ldots, x_t).$$

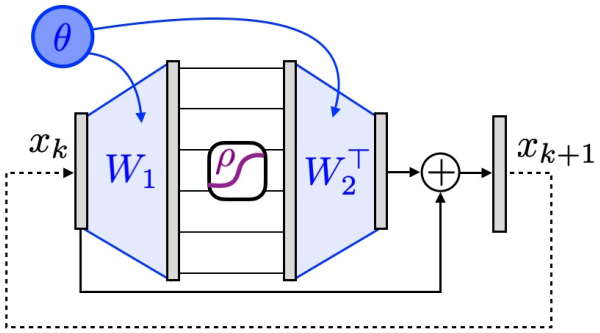


Figure 14.16: Recurrent residual perceptron parameterization.

In practice, this function typically also depends on a few extra variables, in particular on the input values $(x_0, \ldots, x_s)$.

An example of this situation is when one can split the (continuous time) variable as $x(t) = (r(t), s(t))$ and the vector field u in the continuous ODE (14.26) has a symplectic structure of the form $u((r,s),\theta,t) = (F(s,\theta,t), G(r,\theta,t))$. One can then use a leapfrog integration scheme, which defines

$$r_{k+1} = r_k + \tau F(s_k, \theta_k, \tau k) \quad \text{and} \quad s_{k+1} = s_k + \tau G(r_{k+1}, \theta_{k+1/2}, \tau(k+1/2)).$$

One can reverse these equation exactly as

$$s_k = s_{k+1} - \tau G(r_{k+1}, \theta_{k+1/2}, \tau(k+1/2)). \quad \text{and} \quad r_k = r_{k+1} - \tau F(s_k, \theta_k, \tau k).$$

Fixed point maps In some applications (some of which are detailed below), the iterations x_k converges to some $x^\star(\theta)$ which is thus a fixed point

$$x^\star(\theta) = h(x^\star(\theta), \theta).$$

Instead of applying the back-propagation to compute the gradient of $f(\theta) = L(x_t, \theta)$, one can thus apply the implicit function theorem to compute the gradient of $f^\star(\theta) = L(x^\star(\theta), \theta)$. Indeed, one has

$$\nabla f^\star(\theta) = [\partial x^\star(\theta)]^\top (\nabla_x L(x^\star(\theta), \theta)) + \nabla_\theta L(x^\star(\theta), \theta). \tag{14.30}$$

Using the implicit function theorem one can compute the Jacobian as

$$\partial x^\star(\theta) = -\left(\frac{\partial h}{\partial x}(x^\star(\theta), \theta)\right)^{-1} \frac{\partial h}{\partial \theta}(x^\star(\theta), \theta).$$

In practice, one replace in these formulas $x^\star(\theta)$ by x_t, which produces an approximation of $\nabla f(\theta)$. The disadvantage of this method is that it requires the resolution of a linear system, but its advantage is that it bypass the memory storage issue of the backpropagation algorithm.

Argmin layers One can define a mapping from some parameter θ to a point $x(\theta)$ by solving a parametric optimization problem

$$x(\theta) = \operatorname*{argmin}_x \mathcal{E}(x, \theta).$$

The simplest approach to solve this problem is to use a gradient descent scheme, $x_0 = 0$ and

$$x_{k+1} = x_k - \tau \nabla \mathcal{E}(x_k, \theta). \tag{14.31}$$

This has the form (14.25) when using the vector field $u(x, \theta) = \nabla \mathcal{E}(x_k, \theta)$.

Using formula (14.30) in this case where $h = \nabla \mathcal{E}$, one obtains

$$\nabla f^\star(\theta) = -\left(\frac{\partial^2 \mathcal{E}}{\partial x \partial \theta}(x^\star(\theta), \theta)\right)^\top \left(\frac{\partial^2 \mathcal{E}}{\partial x^2}(x^\star(\theta), \theta)\right)^{-1} (\nabla_x L(x^\star(\theta), \theta)) + \nabla_\theta L(x^\star(\theta), \theta)$$

In the special case where the function $f(\theta)$ is the minimized function itself, i.e. $f(\theta) = \mathcal{E}(x^\star(\theta), \theta)$, i.e. $L = \mathcal{E}$, then one can apply the implicit function theorem formula (14.30), which is much simpler since in this case $\nabla_x L(x^\star(\theta), \theta) = 0$ so that

$$\nabla f^\star(\theta) = \nabla_\theta L(x^\star(\theta), \theta). \tag{14.32}$$

This result is often called Danskin theorem or the envelope theorem.

Sinkhorn's algorithm Sinkhorn algorithm approximates the optimal distance between two histograms $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$ using the following recursion on multipliers, initialized as $x_0 \overset{\text{def.}}{=} (u_0, v_0) = (1_n, 1_m)$

$$u_{k+1} = \frac{a}{K v_k}, \quad \text{and} \quad v_{k+1} = \frac{b}{K^\top u_k}.$$

where $\div$ is the pointwise division and $K \in \mathbb{R}_+^{n \times m}$ is a kernel. Denoting $\theta = (a, b) \in \mathbb{R}^{n+m}$ and $x_k = (u_k, v_k) \in \mathbb{R}^{n+m}$, the OT distance is then approximately equal to

$$f(\theta) = \mathcal{E}(x_t, \theta) \overset{\text{def.}}{=} \langle a, \log(u_t) \rangle + \langle b, \log(v_t) \rangle - \varepsilon \langle K u_t, v_t \rangle.$$

Sinkhorn iteration are alternate minimization to find a minimizer of $\mathcal{E}$.

Denoting $\mathcal{K} \overset{\text{def.}}{=} \begin{pmatrix} 0 & K \\ K^\top & 0 \end{pmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}$, one can re-write these iterations in the form (14.27) using

$$h(x, \theta) = \frac{\theta}{\mathcal{K} x} \quad \text{and} \quad L(x_t, \theta) = \mathcal{E}(x_t, \theta) = \langle \theta, \log(x_t) \rangle - \varepsilon \langle K u_t, v_t \rangle.$$

One has the following differential operator

$$[\partial_x h(x, \theta)]^\top = -\mathcal{K}^\top \operatorname{diag}\left(\frac{\theta}{(\mathcal{K} x)^2}\right), \quad [\partial_\theta h(x, \theta)]^\top = \operatorname{diag}\left(\frac{1}{\mathcal{K} x}\right).$$

Similarly as for the argmin layer, at convergence $x_k \to x^\star(\theta)$, one finds a minimizer of $\mathcal{E}$, so that $\nabla_x L(x^\star(\theta), \theta) = 0$ and thus the gradient of $f^\star(\theta) = \mathcal{E}(x^\star(\theta), \theta)$ can be computed using (14.32) i.e.

$$\nabla f^\star(\theta) = \log(x^\star(\theta)).$$

Chapter 15

Deep Learning

Before detailing deep architectures and their use, we start this chapter by presenting two essential computational tools that are used to train these models: stochastic optimization methods and automatic differentiation. In practice, they work hand-in-hand to be able to learn painlessly complicated non-linear models on large-scale datasets.

15.1 Multi-Layers Perceptron

In this section, we study the simplest example of non-linear parametric models, namely Multi-Layers Perceptron (MLP) with a single hidden layer (so they have in total 2 layers). Perceptron (with no hidden layer) corresponds to the linear models studied in the previous sections. MLP with more layers are obtained by stacking together several such simple MLP, and are studied in Section **??**, since the computation of their derivatives is very suited to automatic-differentiation methods.

15.1.1 MLP and its derivative

The basic MLP $a \mapsto h_{W,u}(a)$ takes as input a feature vector $a \in \mathbb{R}^p$, computes an intermediate hidden representation $b = Wa \in \mathbb{R}^q$ using q "neurons" stored as the rows $w_k \in \mathbb{R}^p$ of the weight matrix $W \in \mathbb{R}^{q \times p}$, passes these through a non-linearity $\rho : \mathbb{R} \to \mathbb{R}$, i.e. $\rho(b) = (\rho(b_k))_{k=1}^q$ and then outputs a scalar value as a linear combination with output weights $u \in \mathbb{R}^q$, i.e.

$$h_{W,u}(a) = \langle \rho(Wa),\, u \rangle = \sum_{k=1}^{q} u_k \rho((Wa)_k) = \sum_{k=1}^{q} u_k \rho(\langle a,\, w_k \rangle).$$

This function $h_{W,u}(\cdot)$ is thus a weighted sum of q "ridge functions" $\rho(\langle \cdot,\, w_k \rangle)$. These functions are constant in the direction orthogonal to the neuron w_k and have a profile defined by ρ.

The most popular non-linearities are sigmoid functions such as

$$\rho(r) = \frac{e^r}{1+e^r} \quad \text{and} \quad \rho(r) = \frac{1}{\pi}\mathrm{atan}(r) + \frac{1}{2}$$

and the rectified linear unit (ReLu) function $\rho(r) = \max(r, 0)$.

One often add a bias term in these models, and consider functions of the form $\rho(\langle \cdot,\, w_k \rangle + z_k)$ but this bias term can be integrated in the weight as usual by considering $(\langle a,\, w_k \rangle + z_k = \langle (a, 1),\, (w_k, z_k) \rangle$, so we ignore it in the following section. This simply amount to replacing $a \in \mathbb{R}^p$ by $(a, 1) \in \mathbb{R}^{p+1}$ and adding a dimension $p \mapsto p + 1$, as a pre-processing of the features.

Expressiveness. In order to define function of arbitrary complexity when q increases, it is important that ρ is non-linear. Indeed, if $\rho(s) = s$, then $h_{W,u}(a) = \langle Wa,\, u\rangle = \langle a,\, W^\top u\rangle$. It is thus a linear function with weights $W^\top u$, whatever the number q of neurons. Similarly, if ρ is a polynomial on $\mathbb{R}$ of degree d, then $h_{W,u}(\cdot)$ is itself a polynomial of degree d in $\mathbb{R}^p$, which is a linear space V of finite dimension $\dim(V) = O(p^d)$. So even if q increases, the dimension $\dim(V)$ stays fixed and $h_{W,u}(\cdot)$ cannot approximate an arbitrary function outside V. In sharp contrast, one can show that if ρ is not polynomial, then $h_{W,u}(\cdot)$ can approximate any continuous function, as studied in Section 15.1.3.

15.1.2 MLP and Gradient Computation

Given pairs of features and data values $(a_i, y_i)_{i=1}^n$, and as usual storing the features in the rows of $A \in \mathbb{R}^{n\times p}$, we consider the following least square regression function (similar computation can be done for classification losses)

$$\min_{x=(W,u)} f(W,u) \overset{\text{def.}}{=} \frac{1}{2}\sum_{i=1}^n (h_{W,u}(a_i) - y_i)^2 = \frac{1}{2}\|\rho(AW^\top)u - y\|^2.$$

Note that here, the parameters being optimized are $(W,u) \in \mathbb{R}^{q\times p} \times \mathbb{R}^q$.

Optimizing with respect to u. This function f is convex with respect to u, since it is a quadratic function. Its gradient with respect to u can be computed as in (13.8) and thus

$$\nabla_u f(W,u) = \rho(AW^\top)^\top(\rho(AW^\top)u - y)$$

and one can compute in closed form the solution (assuming $\ker(\rho(AW^\top)) = \{0\}$) as

$$u^\star = [\rho(AW^\top)^\top \rho(AW^\top)]^{-1}\rho(AW^\top)^\top y = [\rho(WA^\top)\rho(AW^\top)]^{-1}\rho(WA^\top)y$$

When $W = \mathrm{Id}_p$ and $\rho(s) = s$ one recovers the least square formula (13.9).

Optimizing with respect to W. The function f is non-convex with respect to W because the function ρ is itself non-linear. Training a MLP is thus a delicate process, and one can only hope to obtain a local minimum of f. It is also important to initialize correctly the neurons $(w_k)_k$ (for instance as unit norm random vector, but bias terms might need some adjustment), while u can be usually initialized at 0.

To compute its gradient with respect to W, we first note that for a perturbation $\varepsilon \in \mathbb{R}^{q\times p}$, one has

$$\rho(A(W+\varepsilon)^\top) = \rho(AW^\top + A\varepsilon^\top) = \rho(AW^\top) + \rho'(AW^\top)\odot(A\varepsilon^\top)$$

where we have denoted "$\odot$" the entry-wise multiplication of matrices, i.e. $U \odot V = (U_{i,j}V_{i,j})_{i,j}$. One thus has,

$$\begin{aligned} f(W+\varepsilon,u) &= \frac{1}{2}\|e + [\rho'(AW^\top)\odot(A\varepsilon^\top)]y\|^2 \quad \text{where} \quad e \overset{\text{def.}}{=} \rho(AW^\top)u - y \in \mathbb{R}^n \\ &= f(W,u) + \langle e,\, [\rho'(AW^\top)\odot(A\varepsilon^\top)]y\rangle + o(\|\varepsilon\|) \\ &= f(W,u) + \langle A\varepsilon^\top,\, \rho'(AW^\top)\odot(eu^\top)\rangle \\ &= f(W,u) + \langle \varepsilon^\top,\, A^\top \times [\rho'(AW^\top)\odot(eu^\top)]\rangle. \end{aligned}$$

The gradient thus reads

$$\nabla_W f(W,u) = [\rho'(WA^\top)\odot(ue^\top)] \times A \in \mathbb{R}^{q\times p}.$$

15.1.3 Universality

In this section, to ease the exposition, we explicitly introduce the bias and use the variable "$x \in \mathbb{R}^p$" in place of "$a \in \mathbb{R}^p$". We thus write the function computed by the MLP (including explicitly the bias z_k) as

$$h_{W,z,u}(x) \overset{\text{def.}}{=} \sum_{k=1}^{q} u_k \varphi_{w_k,z_k}(x) \quad \text{where} \quad \varphi_{w,z}(x) \overset{\text{def.}}{=} \rho(\langle x,\, w\rangle + z).$$

The function $\varphi_{w,z}(x)$ is a ridge function in the direction orthogonal to $\bar{w} \overset{\text{def.}}{=} w/\|w\|$ and passing around the point $-\frac{z}{\|w\|}\bar{w}$.

In the following we assume that $\rho : \mathbb{R} \to \mathbb{R}$ is a bounded function such that

$$\rho(r) \overset{r\to-\infty}{\longrightarrow} 0 \quad \text{and} \quad \rho(r) \overset{r\to+\infty}{\longrightarrow} 1. \tag{15.1}$$

Note in particular that such a function cannot be a polynomial and that the ReLu function does not satisfy these hypothesis (universality for the ReLu is more involved to show). The goal is to show the following theorem.

Theorem 25 (Cybenko, 1989)**.** *For any compact set $\Omega \subset \mathbb{R}^p$, the space spanned by the functions $\{\varphi_{w,z}\}_{w,z}$ is dense in $\mathcal{C}(\Omega)$ for the uniform convergence. This means that for any continuous function f and any $\varepsilon > 0$, there exists $q \in \mathbb{N}$ and weights $(w_k, z_k, u_k)_{k=1}^q$ such that*

$$\forall\, x \in \Omega, \quad |f(x) - \sum_{k=1}^{q} u_k \varphi_{w_k,z_k}(x)| \leqslant \varepsilon.$$

In a typical ML scenario, this implies that one can "overfit" the data, since using a q large enough ensures that the training error can be made arbitrary small. Of course, there is a bias-variance tradeoff, and q needs to be cross-validated to account for the finite number n of data, and ensure a good generalization properties.

Proof in dimension $p = 1$. In 1D, the approximation $h_{W,z,u}$ can be thought as an approximation using smoothed step functions. Indeed, introducing a parameter $\varepsilon > 0$, one has (assuming the function is Lipschitz to ensure uniform convergence),

$$\varphi_{\frac{w}{\varepsilon},\frac{z_k}{\varepsilon}} \overset{\varepsilon\to0}{\longrightarrow} 1_{[-z/w,+\infty[}$$

This means that

$$h_{\frac{W}{\varepsilon},\frac{z}{\varepsilon},u} \overset{\varepsilon\to0}{\longrightarrow} \sum_k u_k 1_{[-z_k/w_k,+\infty[},$$

which is a piecewise constant function. Inversely, any piecewise constant function can be written this way. Indeed, if h assumes the value d_k on each interval $[t_k, t_{k+1}[$, then it can be written as

$$h = \sum_k d_k (1_{[t_k,+\infty[} - 1_{[t_k,+\infty[}).$$

Since the space of piecewise constant functions is dense in continuous function over an interval, this proves the theorem.

Proof in arbitrary dimension p. We start by proving the following dual characterization of density, using bounded Borel measure $\mu \in \mathcal{M}(\Omega)$ i.e. such that $\mu(\Omega) < +\infty$.

Proposition 48. *If ρ is such that for any Borel measure $\mu \in \mathcal{M}(\Omega)$*

$$\left(\forall\,(w,z), \int \rho(\langle x,\, w\rangle + z)\mathrm{d}\mu(x) = 0\right) \quad \Longrightarrow \quad \mu = 0, \tag{15.2}$$

then Theorem 25 holds.

Proof. We consider the linear space

$$\mathcal{S} \stackrel{\text{def.}}{=} \left\{ \sum_{k=1}^{q} u_k \varphi_{w_k,z_k} \;;\; q \in \mathbb{N}, w_k \in \mathbb{R}^p, u_k \in \mathbb{R}, z_k \in \mathbb{R} \right\} \subset \mathcal{C}(\Omega).$$

Let $\bar{\mathcal{S}}$ be its closure in $\mathcal{C}(\Omega)$ for $\|\cdot\|_\infty$, which is a Banach space. If $\bar{\mathcal{S}} \neq \mathcal{C}(\Omega)$, let us pick $g \neq 0$, $g \in \mathcal{C}(\Omega)\backslash\bar{\mathcal{S}}$. We define the linear form L on $\bar{\mathcal{S}} \oplus \text{span}(g)$ as

$$\forall\, s \in \bar{\mathcal{S}},\, \forall\, \lambda \in \mathbb{R}, \qquad L(s + \lambda g) = \lambda$$

so that $L = 0$ on $\bar{\mathcal{S}}$. L is a bounded linear form, so that by Hahn-Banach theorem, it can be extended in a bounded linear form $\bar{L} : \mathcal{C}(\Omega) \to \mathbb{R}$. Since $L \in \mathcal{C}(\Omega)^*$ (the dual space of continuous linear form), and that this dual space is identified with Borel measures, there exists $\mu \in \mathcal{M}(\Omega)$, with $\mu \neq 0$, such that for any continuous function h, $\bar{L}(h) = \int_\Omega h(x)\mathrm{d}\mu(x)$. But since $\bar{L} = 0$ on $\bar{\mathcal{S}}$, $\int \rho(\langle \cdot, w\rangle + z)\mathrm{d}\mu = 0$ for all (w, z) and thus by hypothesis, $\mu = 0$, which is a contradiction. □

The theorem now follows from the following proposition.

Proposition 49. *If ρ is continuous and satisfies (15.1), then it satisfies (15.2).*

Proof. One has

$$\varphi_{\frac{w}{\varepsilon},\frac{u}{\varepsilon}+t}(x) = \rho\left(\frac{\langle x,\, w\rangle + u}{\varepsilon} + t\right) \stackrel{\varepsilon\to 0}{\longrightarrow} \gamma(x) \stackrel{\text{def.}}{=} \begin{cases} 1 & \text{if} \quad H_{w,u}, \\ \rho(t) & \text{if} \quad x \in P_{w,u}, \\ 0 & \text{if} \quad \langle w,\, x\rangle + u < 0, \end{cases}$$

where we defined $H_{w,u} \stackrel{\text{def.}}{=} \{x \;;\; \langle w,\, x\rangle + u > 0\}$ and $P_{w,u} \stackrel{\text{def.}}{=} \{x \;;\; \langle w,\, x\rangle + u = 0\}$. By Lebesgue dominated convergence (since the involved quantities are bounded uniformly on a compact set)

$$\int \varphi_{\frac{w}{\varepsilon},\frac{u}{\varepsilon}+t}\mathrm{d}\mu \stackrel{\varepsilon\to 0}{\longrightarrow} \int \gamma \mathrm{d}\mu = \varphi(t)\mu(P_{w,u}) + \mu(H_{w,u}).$$

Thus if μ is such that all these integrals vanish, then

$$\forall\,(w, u, t), \quad \varphi(t)\mu(P_{w,u}) + \mu(H_{w,u}) = 0.$$

By selecting (t, t') such that $\varphi(t) \neq \varphi(t')$, one has that

$$\forall\,(w, u), \quad \mu(P_{w,u}) = \mu(H_{w,u}) = 0.$$

We now need to show that $\mu = 0$. For a fixed $w \in \mathbb{R}^p$, we consider the function

$$h \in L^\infty(\mathbb{R}), \quad F(h) \stackrel{\text{def.}}{=} \int_\Omega h(\langle w,\, x\rangle)\mathrm{d}\mu(x).$$

$F : L^\infty(\mathbb{R}) \to \mathbb{R}$ is a bounded linear form since $|F(\mu)| \leqslant \|h\|_\infty \mu(\Omega)$ and $\mu(\Omega) < +\infty$. One has

$$F(1_{[-u,+\infty[} = \int_\Omega 1_{[-u,+\infty[}(\langle w,\, x\rangle)\mathrm{d}\mu(x) = \mu(P_{w,u}) + \mu(H_{w,u}) = 0.$$

By linearity, $F(h) = 0$ for all piecewise constant functions, and F is a continuous linear form, so that by density $F(h) = 0$ for all functions $h \in L^\infty(\mathbb{R})$. Applying this for $h(r) = e^{\mathrm{i}r}$ one obtains

$$\hat{\mu}(w) \stackrel{\text{def.}}{=} \int_\Omega e^{\mathrm{i}\langle x,\, w\rangle}\mathrm{d}\mu(x) = 0.$$

This means that the Fourier transform of μ is zero, so that $\mu = 0$. □

Quantitative rates. Note that Theorem 25 is not constructive in the sense that it does not explain how to compute the weights $(w_k, u_k, z_k)_k$ to reach a desired accuracy. Since for a fixed q the function is non-convex, this is not surprising. Some recent studies show that if q is large enough, a simple gradient descent is able to reach an arbitrary good accuracy, but it might require a very large q.

Theorem 25 is also not quantitative since it does not tell how much neurons q is needed to reach a desired accuracy. To obtain quantitative bounds, continuity is not enough, it requires to add smoothness constraints. For instance, Barron proved that if

$$\int \|\omega\| |\hat{f}(\omega)| \mathrm{d}\omega \leqslant C_f$$

where $\hat{f}(\omega) = \int f(x) e^{-\mathrm{i}\langle x, \omega \rangle} \mathrm{d}x$ is the Fourier transform of f, then for $q \in \mathbb{N}$ there exists $(w_k, u_k, z_k)_k$

$$\frac{1}{\mathrm{Vol}(B(0,r))} \int_{\|x\| \leqslant r} |f(x) - \sum_{k=1}^{q} u_k \varphi_{w_k, z_k}(x)|^2 \mathrm{d}x \leqslant \frac{(2rC_f)^2}{q}.$$

The surprising part of this Theorem is that the $1/q$ decay is independent of the dimension p. Note however that the constant involved C_f might depend on p.

15.2 Deep Discriminative Models

15.2.1 Deep Network Structure

Deep learning are estimator $f(x, \beta)$ which are built as composition of simple building blocks. In their simplest form (non-recursive), they corresponds to a simple linear computational graph as already defined in (14.20) (without the loss $\mathcal{L}$), and we write this as

$$f(\cdot, \beta) = f_{L-1}(\cdot, \beta_1) \circ f_{L-2}(\cdot, \beta_2) \circ \ldots \circ f_0(\cdot, \beta_0)$$

where $\beta = (\beta_0, \ldots, \beta_{L-1})$ is the set of parameters, and

$$f_\ell(\cdot, \beta_\ell) : \mathbb{R}^{n_\ell} \to \mathbb{R}^{n_{\ell+1}}$$

While it is possible to consider more complicated architecture (in particular recurrent ones), we restrict here out attention to these simple linear graph computation structures (so-called feedforward networks).

The supervised learning of these parameters β is usually done by empirical risk minimization (12.11) using SGD-type methods as explained in Section 14.2. Note that this results in highly non-convex optimization problems. In particular, strong convergence guarantees such as Theorem 24 do not hold anymore, and only weak convergence (toward stationary points) holds. SGD type technics are however found to work surprisingly well in practice, and it now believe that the success of these deep-architecture approaches (in particular the ability of these over-parameterized model to generalize well) are in large part due to the dynamics of the SGD itself, which induce an implicit regularization effect.

For these simple linear architectures, the gradient of the ERM loss (14.13) can be computed using the reverse mode computation detailed in Section **??**. In particular, in the context of deep learning, formula (15.4). One should however keep in mind that for more complicated (e.g. recursive) architectures, such a simple formula is not anymore available, and one should resort to reverse mode automatic differentiation (see Section **??**), which, while being conceptually simple, is actually implementing possibly highly non-trivial and computationally optimal recursive differentiation.

In most successful applications of deep-learning, each computational block $f_\ell(\cdot, \beta_\ell)$ is actually very simple, and is the composition of

- an affine map, $B_\ell \cdot + b_\ell$ with a matrix $B_\ell \in \mathbb{R}^{n_\ell \times \tilde{n}_\ell}$ and a vector $b_\ell \in \mathbb{R}^{\tilde{n}_\ell}$ parametrized (in most case linearly) by β_ℓ,
- a fixed (not depending on β_ℓ) non-linearity $\rho_\ell : \mathbb{R}^{\tilde{n}_\ell} \to \mathbb{R}^{n_{\ell+1}}$

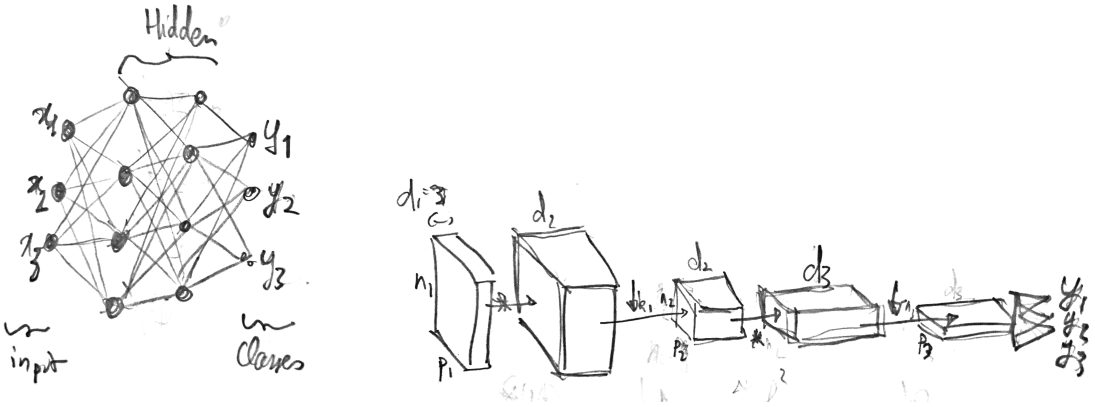


Figure 15.1: Left: example of fully connected network. Right: example of convolutional neural network.

which we write as

$$\forall\, x_\ell \in \mathbb{R}^{n_\ell}, \quad f_\ell(x_\ell, \beta_\ell) = \rho_\ell(B_\ell x_\ell + b_\ell) \in \mathbb{R}^{n_{\ell+1}}. \tag{15.3}$$

In the simplest case, the so-called "fully connected", one has $(B_\ell, b_\ell) = \beta_\ell$, i.e. B_ℓ is a full matrix and its entries (together with the bias b_ℓ) are equal to the set of parameters β_ℓ. Also in the simplest cases ρ_ℓ is a pointwise non-linearity $\rho_\ell(z) = (\tilde{\rho}_\ell(z_k))_k$, where $\tilde{\rho}_\ell : \mathbb{R} \to \mathbb{R}$ is non-linear. The most usual choices are the rectified linear unit (ReLu) $\tilde{\rho}_\ell(s) = \max(s, 0)$ and the sigmoid $\tilde{\rho}_\ell(s) = \theta(s) = (1 + e^{-s})^{-1}$.

The important point here is that the interleaving of non-linear map progressively increases the complexity of the function $f(\cdot, \beta)$.

The parameter $\beta = (B_\ell, b_\ell)_\ell$ of such a deep network are then trained by minimizing the ERM functional (12.11) using SGD-type stochastic optimization method. The gradient can be computed efficiently (with complexity proportional to the application of the model, i.e. $O(\sum_\ell n_\ell^2)$) by automatic differentiation. Since such models are purely feedforward, one can directly use the back-propagation formula (14.20).

For regression tasks, one can can directly use the output of the last layer (using e.g. a ReLu non-linearity) in conjunction with a ℓ^2 squared loss L. For classification tasks, the output of the last layer needs to be transformed into class probabilities by a multi-class logistic map (**??**).

An issue with such a fully connected setting is that the number of parameters is too large to be applicable to large scale data such as images. Furthermore, it ignores any prior knowledge about the data, such as for instance some invariance. This is addressed in more structured architectures, such as for instance convolutional networks detailed in Section 15.2.3.

15.2.2 Perceptron and Shallow Models

Before going on with the description of deep architectures, let us re-interpret the logistic classification method detailed in Sections 12.4.2 and 12.4.3.

The two-class logistic classification model (12.21) is equal to a single layer ($L = 1$) network of the form (15.3) (ignoring the constant bias term) where

$$B_0 x = \langle x, \beta \rangle \quad \text{and} \quad \tilde{\lambda}_0(u) = \theta(u).$$

The resulting one-layer network $f(x, \beta) = \theta(\langle x, \beta \rangle)$ (possibly including a bias term by adding one dummy dimension to x) is trained using the loss, for binary classes $y \in \{0, 1\}$

$$L(t, y) = -\log(t^y (1-t)^{1-y}) = -y \log(t) - (1-y) \log(1-t).$$

In this case, the ERM optimization is of course a convex program.

Multi-class models with K classes are obtained by computing $B_0 x = (\langle x, \beta_k \rangle)_{k=1}^K$, and a normalized logistic map

$$f(x, \beta) = \mathcal{N}((\exp(\langle x, \beta_k \rangle))_k) \quad \text{where} \quad \mathcal{N}(u) = \frac{u}{\sum_k u_k}$$

and assuming the classes are represented using vectors y on the probability simplex, one should use as loss

$$L(t, y) = -\sum_{k=1}^K y_k \log(t_k).$$

15.2.3 Convolutional Neural Networks

In order to be able to tackle data of large size, and also to improve the performances, it is important to leverage some prior knowledge about the structure of the typical data to process. For instance, for signal, images or videos, it is important to make use of the spacial location of the pixels and the translation invariance (up to boundary handling issues) of the domain.

Convolutional neural networks are obtained by considering that the manipulated vectors $x_\ell \in \mathbb{R}^{n_\ell}$ at depth ℓ in the network are of the form $x_\ell \in \mathbb{R}^{\bar{n}_\ell \times d_\ell}$, where $\bar{n}_\ell$ is the number of "spatial" positions (typically along a 1-D, 2-D, or 3-D grid) and d_ℓ is the number of "channels". For instance, for color images, one starts with $\tilde{n}_\ell$ being the number of pixels, and $d_\ell = 3$.

The linear operator $B_\ell : \mathbb{R}^{\bar{n}_\ell \times d_\ell} \to \mathbb{R}^{\bar{n}_\ell \times d_{\ell+1}}$ is then (up to boundary artefact) translation invariant and hence a convolution along each channel (note that the number of channels can change between layers). It is thus parameterized by a set of filters $(\psi_{\ell,r,s})_{s=1,\dots,d_\ell}^{r=1,\dots,d_{\ell+1}}$. Denoting $x_\ell = (x_{\ell,s,\cdot})_{s=1}^{d_\ell}$ the different layers composing x_ℓ, the linear map reads

$$\forall\, r \in \{1, \dots, d_{\ell+1}\}, \quad (B_\ell x_\ell)_{r,\cdot} = \sum_{s=1}^{d_\ell} \psi_{\ell,r,s} \star x_{\ell,s,\cdot}$$

and the bias term $b_\ell \in \mathbb{R}$ is contant (to maintain translation invariance).

The non-linear maps across layers serve two purposes: as before a pointwise non-linearity is applied, and then a sub-sampling helps to reduce the computational complexity of the network. This is very similar to the construction of the fast wavelet transform. Denoting by m_k the amount of down-sampling, where usually $m_k = 1$ (no reduction) or $m_k = 2$ (reduction by a factor two in each direction). One has

$$\lambda_\ell(u) = \left(\tilde{\lambda}_\ell(u_{s,m_k \cdot})\right)_{s=1\dots,d_{\ell+1}}.$$

In the literature, it has been proposed to replace linear sub-sampling by non-linear sub-sampling, for instance the so-called max-pooling (that operate by taking the maximum among groups of m_ℓ successive values), but it seems that linear sub-sampling is sufficient in practice when used in conjunction with very deep (large L) architectures.

The intuition behind such model is that as one moves deeper through the layers, the neurons are receptive to larger areas in the image domain (although, since the transform is non-linear, precisely giving sense to this statement and defining a proper "receptive field" is non-trivial). Using an increasing number of channels helps to define different classes of "detectors" (for the first layer, they detect simple patterns such as edges and corner, and progressively capture more elaborated shapes).

In practice, the last few layers (2 or 3) of such a CNN architectures are chosen to be fully connected. This is possible because, thanks to the sub-sampling, the dimension of these layers are small.

The parameters of such a model are the filters $\beta = (\psi_{\ell,r,s})_{\ell,s,r}$, and they are trained by minimizing the ERM functional (12.11). The gradient is typically computed by backpropagation. Indeed, when computing the gradient with respect to some filter $\psi_{\ell,r,s}$, the feedforward computational graph has the form (14.20). For simplicity, we re-formulate this computation in the case of a single channel per layer (multiple layer can

be understood as replacing convolution by matrix-domain convolution). The forward pass computes all the inner coefficients, by traversing the network from $\ell = 0$ to $\ell = L-1$,

$$x_{\ell+1} = \lambda_\ell(\psi_\ell \star x_\ell)$$

where $\lambda_\ell(u) = (\tilde{\lambda}_\ell(u_i))_i$ is applied component wise. Then, denoting $\mathcal{E}(\beta) = \mathcal{L}(\beta, y)$ the loss to be minimized with respect to the set of filters $\beta = (\psi_\ell)_\ell$, and denoting $\nabla_\ell \mathcal{E}(\beta) = \frac{\partial \mathcal{E}(\beta)}{\partial \psi_\ell}$ the gradient with respect to ψ_ℓ, one computes all the gradients by traversing the network in reverse order, from $\ell = L-1$ to $\ell = 0$

$$\nabla_\ell \mathcal{E}(\beta) = [\lambda'_\ell(\psi_\ell \star x_\ell)] \odot [\bar{\psi}_\ell \star \nabla_{\ell+1} \mathcal{E}(\beta)], \tag{15.4}$$

where $\lambda'_\ell(u) = (\tilde{\lambda}'_\ell(u_i))_i$ applies the derivative of $\tilde{\lambda}_\ell$ component wise, and where $\bar{\psi}_\ell = \psi_\ell(-\cdot)$ is the reversed filter. Here, $\odot$ is the pointwise multiplication of vectors. The recursion is initialized as $\nabla \mathcal{E}_L(\beta) = \nabla \mathcal{L}(x_L, y)$, the gradient of the loss itself.

This recursion (15.4) is the celebrated backpropagation algorithm put forward by Yann Lecun. Note that to understand and code these iterations, one does not need to rely on the advanced machinery of reverse mode automatic differentiation exposed in Section **??**. The general automatic differentiation method is however crucial to master because advanced deep-learning architectures are not purely feedforward, and might include recursive connexions. Furthermore, automatic differentiation is useful outside deep learning, and considerably eases prototyping for modern data-sciences with complicated non-linear models.

15.2.4 Scattering Transform

The scattering transform, introduced by Mallat and his collaborators, is a specific instance of deep convolutional network, where the filters $(\psi_{\ell,r,s})_{\ell,s,r}$ are not trained, and are fixed to be wavelet filters. This network can be understood as a non-linear extension of the wavelet transform. In practice, the fact that it is fixed prevent it to be applied to arbitrary data (and is used mostly on signals and images) and it does not lead to state of the art results for natural images. Nevertheless, it allows to derives some regularity properties about the feature extraction map $f(\cdot, \beta)$ computed by the network in term of stability to diffeomorphisms. It can also be used as a set of fixed initial features which can be further enhanced by a trained deep network, as shown by Edouard Oyallon.

Chapter 16

Convex Analysis

The main references for this chapter are [10, 3]. This chapters uses different notations than the previous one, and we denote $f(x)$ a typical function to minimized with respect to the variable x. We discus here some important concepts from convex analysis for non-smooth optimization.

16.1 Basics of Convex Analysis

We consider minimization problems of the form

$$\min_{x \in \mathcal{H}} f(x) \tag{16.1}$$

over the finite dimension (Hilbertian) space $\mathcal{H} \overset{\text{def.}}{=} \mathbb{R}^N$, with the canonical inner product $\langle \cdot, \cdot \rangle$. Most of the results of this chapter extends to possibly infinite dimensional Hilbert space.

Here $f : \mathcal{H} \longrightarrow \bar{\mathbb{R}} \overset{\text{def.}}{=} \mathbb{R} \cup \{+\infty\}$ is a convex function. Note that we allow here f to take the value $+\infty$ to integrate constraints in the objective, and the constraint set is thus the "domain" of the function

$$\mathrm{dom}(f) \overset{\text{def.}}{=} \{x \,;\, f(x) < +\infty\}.$$

A useful notation is the indicator function of a set $\mathcal{C} \subset \mathcal{H}$

$$\iota_{\mathcal{C}}(x) \overset{\text{def.}}{=} \begin{cases} 0 & \text{if} \quad x \in \mathcal{C}, \\ +\infty & \text{otherwise.} \end{cases}$$

16.1.1 Convex Sets and Functions

A convex set $\Omega \subset \mathcal{H}$ is such that

$$\forall\, (x, y, t) \in \mathcal{H}^2 \times [0, 1], \quad (1-t)x + ty \in \Omega.$$

A convex function is such that

$$\forall\, (x, y, t) \in \mathcal{H}^2 \times [0, 1], \quad f((1-t)x + ty) \leqslant (1-t)f(x) + tf(y) \tag{16.2}$$

and this is equivalent to its epigraph $\{(x, r) \in \mathcal{H} \times \mathbb{R} \,;\, r \geqslant f(x)\}$ being a convex set. Note that here we use $\leqslant$ as a comparison over $\bar{\mathbb{R}}$. The function f is strictly convex if equality in (16.2) only holds for $t \in \{0, 1\}$. A set Ω being convex is equivalent to $\iota_{\mathcal{C}}$ being a convex function.

In the remaining part of this chapter, we consider convex function f which are proper, i.e. such that $\mathrm{dom}(f) \neq \emptyset$, and that should be lower-semi-continuous (lsc), i.e. such that for all $x \in \mathcal{H}$,

$$\liminf_{y \to x} f(y) \geqslant f(x).$$

It is equivalent to $\mathrm{epi}(f)$ being a closed convex set. We denote $\Gamma_0(\mathcal{H})$ the set of proper convex lsc functions.

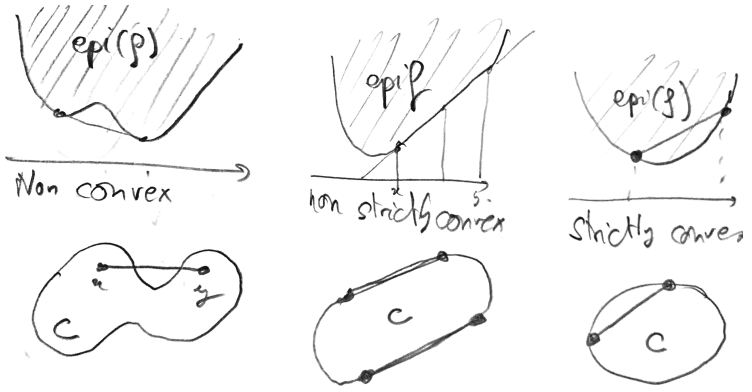


Figure 16.1: Convexity and strict convexity for function and sets.

16.1.2 First Order Conditions

Existence of minimizers. Before looking at first optimality conditions, one has to check that there exists minimizers, which is implied by the l.s.c. property and coercivity.

Proposition 50. *If f is l.s.c. and coercive (i.e. $f(x) \to +\infty$ as $x \to +\infty$), then there exists a minimizer $x^\star$ of f.*

Proof. Since f is coercive, it is bounded from below, one can consider a minimizing sequence $(x_n)_n$ such that $f(x_n) \to \min f$. Since f is l.s.c., this implies that the sub-level set of f are closed, and coercivity imply they are bounded, hence compact. One can thus extract from $(x_n)_n$ a converging sub-sequence $(x_{n(p)})_p$, $x_{n(p)} \to x^\star$. Lower semi-continuity implies that $\min f = \lim_p f(x_{n(p)}) \geqslant f(x^\star)$, and hence $x^\star$ is a minimizer. □

This existence proof is often called the "direct method of calculus of variation". Note that if the function f is in $\Gamma_0(\mathcal{H})$, then the set of minimizer $\operatorname{argmin} f$ is a closed convex set, and all local minimizers (i.e. minimizer of the function restricted to an open ball) are global one. If it is furthermore strictly convex, then there is a single minimizer.

Sub-differential. The sub-differential at x of such a f is defined as

$$\partial f(x) \stackrel{\text{def.}}{=} \{u \in \mathcal{H}^* \,;\, \forall y, f(y) \geqslant f(x) + \langle u,\, y - x\rangle\}\,.$$

We denote here $\mathcal{H}^* = \mathbb{R}^N$ the set of "dual" vector. Although in finite dimensional Euclidean space, this distinction is not needed, it helps to distinguish primal from dual vectors, and recall that the duality pairing implicitly used depends on the choice of an inner product. The sub-differential $\partial f(x)$ is thus the set of "slopes" u of tangent affine planes $f(x) + \langle u,\, z - x\rangle$ that fits below the graph of f.

Note that f being differentiable at x is equivalent to the sub-differential being reduced to a singleton (equal to the gradient vector)

$$\partial f(x) = \{\nabla f(x)\}.$$

Informally, the "size" of $\partial f(x)$ controls how smooth f is at x.

Note that one can have $\partial f(x) = \emptyset$, for instance if $x \notin \operatorname{dom}(f)$. Note also that one can still have $x \in \operatorname{dom}(f)$ and $\partial f(x) = \emptyset$, for instance take $f(x) = -\sqrt{1-x^2} + \iota_{[-1,1]}(x)$ at $x = \pm 1$.

Since $\partial f(x) \subset \mathcal{H}^*$ is an intersection of half space, it is a closed convex set. The operator $\partial f : \mathcal{H} \mapsto 2^{\mathcal{H}^*}$ is thus "set-valued", and we often denote this as $\partial f : \mathcal{H} \hookrightarrow \mathcal{H}^*$.

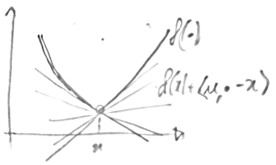


Figure 16.2: The subdifferential

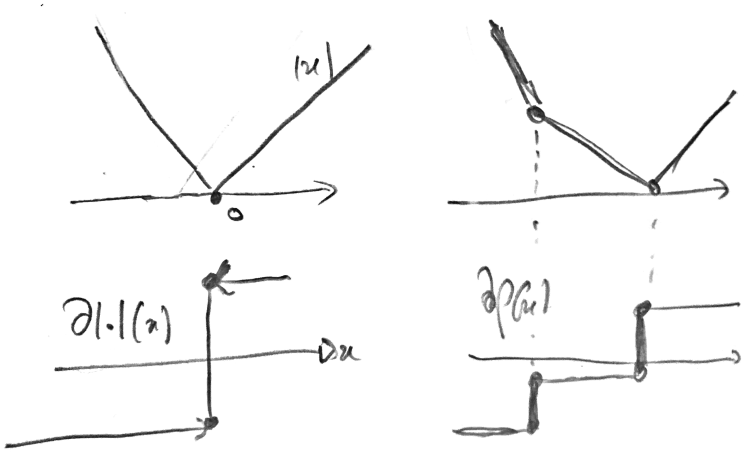


Figure 16.3: Subdifferential of the absolute value and a piecewise affine convex function.

Remark 8 (Maximally monotone operator). The operator ∂f is particular instance of so-called monotone operator, since one can check that $U = \partial f$ satisfies

$$\forall\, (u,v) \in U(x) \times U(y), \quad \langle y-x,\, v-u \rangle \geqslant 0.$$

In the 1-D setting, being monotone is the same as being an increasing map. Sub-differential can also be shown to be maximally monotone, in the sense that such an operator is not striclty included in the graph of another monotone operator. Note that there exists monotone maps which are not subdifferential, for instance $(x,y) \mapsto (-y,x)$. Much of the theory of convex analysis and convex optimization can be extended to deal with arbitrary maximally monotone-maps in place of subdifferential, but we will not pursue this here.

A prototypical example is the absolute value $f(x) = |\cdot|$, and writing conveniently $\partial f(x) = \partial|\cdot|(x)$, one verifies that

$$\partial|\cdot|(x) = \begin{cases} -1 & \text{if} \quad x<0, \\ +1 & \text{if} \quad x>0, \\ [-1,1] & \text{if} \quad x=0. \end{cases}$$

First Order Conditions. The subdifferential is crucial for this simple but extremely important proposition.

Proposition 51. *$x^\star$ is a minimizer of f is and only if $0 \in \partial f(x^\star)$.*

Proof. One has

$$x^\star \in \operatorname{argmin} f \quad \Leftrightarrow \quad (\forall y, f(x^\star) \leqslant f(y) + \langle 0,\, x^\star - y \rangle) \quad \Leftrightarrow \quad 0 \in \partial f(x^\star).$$

□

Sub-differential calculus. There is a large set of calculus rules that allows to simplify the computation of sub-differentials. For decomposable function $f(x_1,\ldots,x_K) = \sum_{k=1}^K f_k(x_k)$, the sub-differential is the product of the sub-differentials

$$\partial f(x_1,\ldots,x_K) = \partial f_1(x_1) \times \ldots \times \partial f_K(x_K).$$

This can be used to compute the sub-differential of the ℓ^1 norm $\|x\|_1 = \sum_{k=1}^N |x_k|$

$$\partial\|\cdot\|_1(x) = \prod_{k=1}^N \partial|\cdot|(x_k)$$

which is thus an hyper rectangle. This means that, denoting $I = \mathrm{supp}(x)$, one has $u \in \partial\|\cdot\|_1(x)$ is equivalent to

$$u_I = \mathrm{sign}(x_I) \quad \text{and} \quad \|u_{I^c}\|_\infty \leqslant 1.$$

A tricky problem is to compute the sub-differential of the sum of two functions. If one of the two function is continuous at x (i.e. it has a finite value), then

$$\partial(f+g)(x) = \partial f(x) \oplus \partial g(x) = \{u+v \;;\; (u,v) \in \partial f(x) \times \partial g(x)\}$$

where $\oplus$ thus denotes the Minkowski sum. For instance, if f is differentiable at x, then

$$\partial(f+g)(x) = \nabla f(x) + \partial g(x) = \{\nabla f(x) + v \;;\; v \in \partial g(x)\}\,.$$

Positive linear scaling is simple to handle

$$\forall\, \lambda \in \mathbb{R}_+, \quad \partial(\lambda f)(x) = \lambda(\partial f(x)).$$

The chain rule for sub-differential is difficult since in general composition does not work so-well with convexity. The only simple case is composition with linear functions, which preserves convexity. Denoting $A \in \mathbb{R}^{P\times N}$ and $f \in \Gamma_0(\mathbb{R}^P)$, one has that $f \circ A \in \Gamma_0(\mathbb{R}^N)$ and

$$\partial(f\circ A)(x) = A^*(\partial f)(Ax) \overset{\text{def.}}{=} \{A^*u \;;\; u \in \partial f(Ax)\}\,.$$

Normal cone. The sud-differential of an indicator function is a convex cone, the so-called normal cone to the constraint

$$\forall\, x \in \mathcal{C}, \quad \partial\iota_{\mathcal{C}}(x) = \mathcal{N}_{\mathcal{C}}(x) \overset{\text{def.}}{=} \{v \;;\; \forall\, z \in \mathcal{C}, \langle z-x,\, v\rangle \leqslant 0\}\,.$$

Note that for $x \notin \mathcal{C}$, $\partial\iota_{\mathcal{C}}(x) = \emptyset$. For an affine space $\mathcal{C} = a + \mathcal{V}$ where $\mathcal{V} \subset \mathcal{H}$ is a linear space, then $\mathcal{N}_{\mathcal{C}}(x) = \mathcal{V}^\perp$ is the usual orthogonal for linear spaces. If $x \in \mathrm{int}(\mathcal{C})$ is in the interior of $\mathcal{C}$, then $\mathcal{N}_{\mathcal{C}}(x) = \{0\}$. In some sense, the more non-regular the boundary of $\mathcal{C}$ is at x, the larger is the normal cone.

The normal cone is a way to express first order condition for constrained problem

$$\min_{x\in\mathcal{C}} f(x)$$

which reads, if f is continuous

$$0 \in \partial f(x) + \partial\iota_{\mathcal{C}}(x) \quad \Leftrightarrow \quad \exists \xi \in \partial f(x), -\xi \in \mathcal{N}_{\mathcal{C}}(x) \quad \Leftrightarrow \quad \partial f(x) \cap (-\mathcal{N}_{\mathcal{C}}(x)) \neq \emptyset.$$

If f is differentiable, it reads $-\nabla f(x) \in \mathcal{N}_{\mathcal{C}}(x)$.

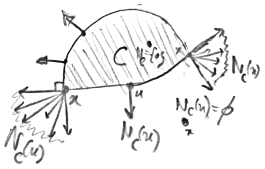


Figure 16.4: Normal cones

16.2 Convex Duality

Duality is associated to a particular formulation of the optimization problem, so that for instance making change of variables results in a different duality.

16.2.1 Lagrange Duality

We consider a minimization of the form

$$p^\star = \min_{x\in\mathbb{R}^N} \{f(x) \;;\; Ax = y \quad \text{and} \quad g(x) \leqslant 0\} \tag{16.3}$$

for a continuous convex functions $f : \mathcal{H} \to 0$, a matrix $A \in \mathbb{R}^{P\times N}$ and a function $g : \mathcal{H} \to \mathbb{R}^Q$ such that each of its coordinates $g_i : \mathcal{H} \to \mathbb{R}$ are continuous and convex. Note that it is always the case that equality in convex program corresponds to affine ones. One can always write a convex minimization problem with

positivity constraints in the form (16.3), although there exists infinite way of doing so (each one giving a different duality formula).

Here we have assumed for simplicity that f is continuous, i.e. $\mathrm{dom}(f) = \mathbb{R}^N$. The following exposition can be generalized to $\mathrm{dom}(f)$ being arbitrary, but this is more technical. For the sake of simplicity, we thus assume all the constraint defining the domain are encoded in $Ax = y$ and $g(x) \leqslant 0$

Note that it is possible to generalized the previous Lagrange duality results by replacing "$x \geqslant 0$" by "$X \succeq 0$" where X is a matrix (and in fact even more generally using convex cones).

We use the following fact

$$\sup_{u \in \mathbb{R}^P} \langle r,\, u \rangle = \begin{cases} 0 & \text{if} \quad r = 0, \\ +\infty & \text{if} \quad r \neq 0, \end{cases} \qquad \text{and} \qquad \sup_{v \in \mathbb{R}^Q_+} \langle s,\, v \rangle = \begin{cases} 0 & \text{if} \quad s \leqslant 0, \\ +\infty & \text{otherwise,} \end{cases}$$

to encode the constraints $r = Ax - y = 0$ and $s = g(x) \leqslant 0$.

One can represent the constraints appearing in (16.3) conveniently using a maximization over so-called Lagrange multipliers

$$p^\star = \inf_x \max_{u \in \mathbb{R}^P, v \in \mathbb{R}^Q_+} \mathcal{L}(x,u,v) \overset{\text{def.}}{=} f(x) + \langle Ax - y,\, u \rangle + \langle g(x),\, v \rangle.$$

It is tempting to inverse the inf and the sup, and study

$$d^\star = \sup_{(u,v) \in \mathbb{R}^P \times \mathbb{R}^Q_+} F(u,v) \overset{\text{def.}}{=} \inf_x f(x) + \langle Ax - y,\, u \rangle + \langle g(x),\, v \rangle. \tag{16.4}$$

One remarks that F is a concave function (as being the minimum of linear forms), and this "dual" problem is thus a maximization of a concave function.

The following proposition is the so-called weak duality, which assert that values of the dual problems always lower bounds values of the primal one

Proposition 52. *One always has, for all $(u,v) \in \mathbb{R}^P \times \mathbb{R}^Q_+$,*

$$F(u,v) \leqslant p^\star \quad \Longrightarrow \quad d^\star \leqslant p^\star.$$

Proof. Since $g(x) \leqslant 0$ and $v \geqslant 0$, one has $\langle g(x),\, v \rangle \leqslant 0$, and since $Ax = y$, one has $\langle Ax - y,\, u \rangle = 0$, so that

$$\mathcal{L}(x,u,v) \leqslant f(x) \quad \Longrightarrow \quad F(u,v) = \inf_x \mathcal{L}(x,u,v) \leqslant \inf_x f(x) = p^\star.$$

□

The following fundamental theorem, more difficult to prove, gives a sufficient condition (so-called qualification of the constraints) such that one actually has equality.

Theorem 26. *If*

$$\exists x_0 \in \mathbb{R}^N, \quad Ax_0 = y \quad \textit{and} \quad g(x_0) < 0, \tag{16.5}$$

then $p^\star = d^\star$. Furthermore, $x^\star$ and $(u^\star, v^\star)$ are solutions of respectively (16.3) and (16.4) if and only if

$$Ax^\star = y, \quad g(x^\star) \leqslant 0, \quad u^\star \geqslant 0 \tag{16.6}$$

$$0 \in \partial f(x^\star) + A^* u^\star + \sum_i v_i^\star \partial g_i(x^\star) \tag{16.7}$$

$$\forall\, i, \quad u_i^\star g_i(x^\star) = 0 \tag{16.8}$$

The existence of such an x_0 is called "constraint qualification", and as written here, this corresponds to the so-called "Slater" qualification condition (many other weaker sufficient conditions exist).

Condition (16.6) is simply the primal and dual constraints. Condition (16.7) is the first order condition for the minimization of $\mathcal{L}(x,u,v)$ over x. Condition (16.8) is the first order condition for the maximization of $\mathcal{L}(x,u,v)$ over (u,v). These three conditions are often referred to as "Karush-Kuhn-Tucker" (KKT) conditions, and under a constraint qualification condition, they are necessary and sufficient condition for optimality.

The last condition $u_i^\star g_i(x^\star)=0$ (so called "complementary slackness") states that if $g_i(x^\star)<0$ (the constraints is not saturated) then $u_i=0$, and also that if $u_i>0$ then $g_i(x^\star)=0$.

Note that it is possible to weaken the hypotheses of this theorem, for the linear constraints of the form $g_i(x)=\langle x,\,h_i\rangle-c_i\leqslant 0$, by replacing the $g_i(x_0)<0$ by the weaker condition $\langle x_0,\,h_i\rangle\leqslant c_i$.

One can generalize this theorem to the setting where $\mathrm{dom}(f)$ is not equal to $\mathbb{R}^N$ (i.e. it is not continuous, and thus integrates extra constraint beside the $\leqslant$). In this case, one has to add the extra constraint $x_0\in\mathrm{relint}(\mathrm{dom}(f))$.

Theorem 26 generalizes the necessary conditions provided by Lagrange multipliers for equality constrained optimization. The setting is both more complex because one can deal with inequalities that might be saturated (so this introduce positivity constraints on the multipliers v) but also simpler because of convexity (which thus gives also necessary conditions).

As a simple example, we now derive the dual for a simple linear projection problem. A more complicated computation is carried over in Section 10.1.5 for the Lasso. We consider

$$p^\star=\min_{Ax=y}\frac{1}{2}\|x-z\|^2=\min_x\max_u\frac{1}{2}\|x-z\|^2+\langle Ax-y,\,u\rangle=\max_u F(u)=\min_x\frac{1}{2}\|x-z\|^2+\langle Ax-y,\,u\rangle,$$

where we used the fact that strong duality holds because only linear constraints are involved. For each u, the optimal x satisfies $x-z+A^*u$, i.e. $x=z-A^*u$, so that

$$F(u)=\frac{1}{2}\|A^*u\|^2+\langle A(z-A^*u)-y,\,u\rangle=-\frac{1}{2}\|A^*u\|^2+\langle u,\,Az-y\rangle.$$

Weak duality states $p^\star\geqslant F(u)$ for any u, and $p^\star=F(u^\star)$ where the optimal $u^\star$ satisfies $AA^*u=Az-y$. If $y\in\mathrm{Im}(A)$, then such a $u^\star$ exists and can be chosen as $u^\star=u=(AA^*)^{-1}(Az-y)$, and the (unique) primal solution reads

$$x^\star=\mathrm{Proj}_{A\cdot=y}(z)(\mathrm{Id}-A^+A)z-A^+y. \tag{16.9}$$

16.2.2 Legendre-Fenchel Transform

In order to simplify and accelerate computation involving Lagrange duality, it is very convenient to introduce a particular transformation of convex functions the Legendre-Fenchel transform. In some sense, it is the canonical "isomorphisms" (pairing) between convex functions. In spirit, is plays a similar role for convex function as the Fourier transform for signal or images.

For $f\in\Gamma_0(\mathcal{H})$, we define its Legendre-Fenchel transform as

$$f^*(u)\overset{\text{def.}}{=}\sup_x\langle x,\,u\rangle-f(x). \tag{16.10}$$

Being the maximum of affine functional, one obtains that f^* is itself a convex function, and that in fact $f^\star\in\Gamma_0(\mathcal{H}^*)$. One can prove the following fundamental bi-duality result.

Theorem 27. *One has*

$$\forall\, f\in\Gamma_0(\mathcal{H}),\quad (f^*)^*=f.$$

In fact, f^* is convex even in the case where f is not, and f^{**} is the convex envelop of f (i.e. the largest convex function smaller than f). **[ToDo: drawing]**

One has the following basic property relating the sub-differentials of f and f^*.

Proposition 53. *One has $\partial f^*=(\partial f)^{-1}$, where the inverse of a set valued map is defined in (17.10), and*

$$\forall\,(x,y),\quad \langle x,\,y\rangle\leqslant f(x)+f^*(y)\quad\textit{and}\quad\langle x,\,y\rangle=f(x)+f^*(y)\quad\Leftrightarrow\quad x\in\partial f^*(y)\quad\Leftrightarrow\quad y\in\partial f(x).$$

Proposition 54. *For $1/p + 1/q = 1$,*

$$(\iota_{\|\cdot\|_p \leqslant 1})^* = \|\cdot\|_q \quad \text{and} \quad (\|\cdot\|_q)^* = \iota_{\|\cdot\|_p \leqslant 1}$$

Let us now give some example of Legendre transform.

Proposition 55. *For $f(x) = \frac{1}{2}\langle Ax, x\rangle - \langle b, x\rangle$ with A inversible, then $f^*(u) = \frac{1}{2}\langle A^{-1}u, u\rangle - \frac{1}{2}\langle A^{-1}b, b\rangle$. In particular, for $f = \|\cdot\|^2/2$, then $f^* = f$. One has* ***[ToDo: check]***

$$f(\cdot - z)^* = f + \langle z, \cdot\rangle, \quad (f + \langle z, \cdot\rangle)^* = f(\cdot - z), \quad (\lambda f)^* = \lambda f^*(\cdot/\lambda).$$

Proof. One has $f^*(u) = \langle Ax^\star, x^\star\rangle - \langle b, x^\star\rangle$ where $x^\star$ solves

$$u = Ax^\star - b \quad \Longrightarrow \quad x^\star = A^{-1}u + A^{-1}b.$$

Hence

$$f^*(u) = \frac{1}{2}\langle AA^{-1}(u+b),\, A^{-1}(u+b)\rangle - \langle b,\, A^{-1}(u+b)\rangle = \frac{1}{2}\langle A^{-1}u, u\rangle - \frac{1}{2}\langle A^{-1}b, b\rangle$$

□

Legendre transform and smoothness. While the Fourier transform is a pairing between smoothness and decay (see Section **??**), the Legendre-Fenchel is really a pairing between smoothness and strong convexity. This can be intuitively seen by the fact that the Legendre-Fenchel inverts the sub-differentials (**??**) and hence when the functions involved are $\mathcal{C}^2$, it inverse the Hessians

$$\partial^2 f(x) = (\partial^2 f^*(y))^{-1} \quad \text{at} \quad y = \nabla f(x).$$

This relation between Hessian can be seen as implying the exchange of strong convexity and uniform bound on the Hessian, as detailed in Proposition 43.

Proposition 56. *One has*

$$\nabla f \text{ is } L\text{-Lipschitz} \quad \Longleftrightarrow \quad \nabla f^* \text{ is } \mu\text{-strongly convex.}$$

This results suggests a way to smooth any function f. Instead of doing a convolution, one can use the infimal convolution

$$(f \otimes g)(x) \stackrel{\text{def.}}{=} \sup_{y+y'=x} f(y) + g(y').$$

One can check that if (f, g) are convex, so is $f \otimes g$, and that the Legendre transform actually exchanges sum and inf-convolution

$$(f+g)^* = f \otimes g \quad \text{and} \quad (f \otimes g)^* = f + g.$$

The Moreau-Yosida regularization of f is corresponds to a μ-strict-convexification of f^*, i.e.

$$f_\mu \stackrel{\text{def.}}{=} f \otimes (\frac{1}{2\mu}\|\cdot\|^2) = (f^* + \frac{\mu}{2}\|\cdot\|^2)^*. \tag{16.11}$$

Since $f^* + \frac{\mu}{2}\|\cdot\|^2$ is at least μ-strongly convex, then f_μ as a $1/\mu$-Lipchitz gradient.

As an example, the Moreau-Yosida regularization of the absolute value reads

$$(|\cdot|_\mu)(x) = \begin{cases} \frac{1}{2\mu}x^2 & \text{if} \quad |x| \leqslant \mu, \\ |x| - \frac{\mu}{2} & \text{if} \quad |x| > \mu. \end{cases}$$

This should be compared with the regularization $\sqrt{x^2 + \mu^2}$ (which is above the curve) that we used previously. **[ToDo: add drawing]**

16.2.3 Fenchel-Rockafellar Duality

Very often the Lagrange dual can be expressed using the conjugate of the function f. We give here a particularly important example, which is often called Fenchel-Rockafellar Duality.

We consider the following structured minimization problem

$$p^\star = \inf_x \; f(x) + g(Ax). \tag{16.12}$$

Re-writing it as

$$\inf_{y=Ax} \; f(x) + g(y),$$

we can form the primal-dual problem

$$\inf_{(x,y)} \sup_u \; f(x) + g(y) + \langle Ax - y,\, u\rangle.$$

If sufficient condition on the domain of (f, g) holds (such as those stated in Theorem **??**), one one can exchange the min and the max and obtains the dual problem

$$d^\star = \sup_u \min_{(x,y)} \; f(x) + g(y) + \langle Ax - y,\, u\rangle \tag{16.13}$$

$$= \sup_u \; \Big(\min_x \; \langle x,\, A^*u\rangle + f(x)\Big) + \Big(\min_y \; -\langle y,\, u\rangle + g(y)\Big) \tag{16.14}$$

which leads to the celebrated Fenchel-Rockafellar, which we summarize together with qualification sufficient condition ensuring strong duality.

Theorem 28 (Fenchel-Rockafellar)**.** *If*

$$0 \in \operatorname{relint}(\operatorname{dom}(g)) - A \operatorname{relint}(\operatorname{dom}(f)) \tag{16.15}$$

the one has the following strong duality

$$\inf_x \; f(x) + g(Ax) = \inf_x \sup_u \; \mathcal{L}(x,u) = \sup_u \inf_x \; \mathcal{L}(x,u) = \sup_u \; -f^*(-A^*u) - g^*(u) \tag{16.16}$$

$$\text{where} \quad \mathcal{L}(x,u) \overset{\text{def.}}{=} f(x) + \langle Ax,\, u\rangle - g^*(u).$$

Furthermore one has that $(x^\star, u^\star)$ is a pair of optimal primal-dual solutions if and only if

$$-A^*u^\star \in \partial f(x^\star) \quad \text{and} \quad Ax^\star \in \partial g^*(u^\star). \tag{16.17}$$

Condition (16.15) is the constraint qualification ensuring that one can inverse the inf and the sup in (16.16). It can be recovered from Slater's qualification condition (16.5) when deriving the dual problem as in (16.13). The primal-dual relations (16.17) are the first order condition along the x and the u variables in minimization and maximization of $\mathcal{L}$. They are sometimes summarised in "matrix" form

$$0 \in \begin{pmatrix} \partial f & A^* \\ -A & \partial g^* \end{pmatrix} \begin{pmatrix} x^\star \\ u^\star \end{pmatrix}.$$

Chapter 17

Non-smooth Convex Optimization

The main references for this chapter are [7, 8, 3], see also [19, 2, 1].
We consider a general convex optimization problem

$$\min_{x \in \mathcal{H}} f(x) \tag{17.1}$$

where $\mathcal{H} = \mathbb{R}^p$ is a finite dimensional Hilbertian (i.e. Euclidean) space, and try to devise "cheap" algorithms with a low computational cost per iterations. The class of algorithms considered are first order, i.e. they make use of gradient information.

17.1 Descent Methods

We have already encountered the gradient descent method informally in Section **??**Â for the regularization of inverse problem. We now give a detailed analysis of the method.

17.1.1 Gradient Descent

The optimization program (8.26) is an example of unconstrained convex optimization of the form (17.1) where $f : \mathcal{H} \to \mathbb{R}$ is a $\mathcal{C}^1$ function with Lipschitz gradient (so-called "smooth" function). Recall that the gradient $\nabla f : \mathcal{H} \mapsto \mathcal{H}$ of this functional (not to be confound with the discretized gradient $\nabla x \in \mathcal{H}$ of f) is defined by the following first order relation

$$f(x + r) = f(x) + \langle f, r \rangle_{\mathcal{H}} + O(\|r\|_{\mathcal{H}}^2)$$

where we used $O(\|r\|_{\mathcal{H}}^2)$ in place of $o(\|r\|_{\mathcal{H}})$ (for differentiable function) because we assume here f is of class $\mathcal{C}^1$ (i.e. the gradient is continuous). Section 8.4.3 shows typical examples of gradient computation.

For such a function, the gradient descent algorithm is defined as

$$x^{(\ell+1)} \overset{\text{def.}}{=} x^{(\ell)} - \tau_\ell \nabla f(x^{(\ell)}), \tag{17.2}$$

where the step size $\tau_\ell > 0$ should be small enough to guarantee convergence, but large enough for this algorithm to be fast.

17.1.2 Sub-gradient Descent

The gradient descent (17.2) cannot be applied on a non-smooth function f. One can use in place of a gradient a sub-gradient, which defines the sub-gradient descent

$$x^{(\ell+1)} \overset{\text{def.}}{=} x^{(\ell)} - \tau_\ell g^{(\ell)} \quad \text{where} \quad g^{(\ell)} \in \partial f(x^{(\ell)}). \tag{17.3}$$

The main issue with this scheme is that to ensure convergence, the iterate should go to zero. One can easily convince oneself why by looking at the iterates on a function $f(x) = |x|$.

Theorem 29. *If $\sum_\ell \tau_\ell = +\infty$ and $\sum_\ell \tau_\ell^2 < +\infty$, then $x^{(\ell)}$ converges to a minimizer of f.*

17.1.3 Projected Gradient Descent

We consider a generic constraint optimization problem as

$$\min_{x \in \mathcal{C}} f(x) \tag{17.4}$$

where $\mathcal{C} \subset \mathbb{R}^S$ is a closed convex set and $f : \mathbb{R}^S \to \mathbb{R}$ is a smooth convex function (at least of class $\mathcal{C}^1$).

The gradient descent algorithm (17.2)Â is generalized to solve a constrained problem using the projected gradient descent

$$x^{(\ell+1)} \overset{\text{def.}}{=} \mathrm{Proj}_{\mathcal{C}}\left(x^{(\ell)} - \tau_\ell \nabla f(x^{(\ell)})\right), \tag{17.5}$$

where $\mathrm{Proj}_{\mathcal{C}}$ is the orthogonal projector on $\mathcal{C}$

$$\mathrm{Proj}_{\mathcal{C}}(x) = \operatorname*{argmin}_{x' \in \mathcal{C}} \|x - x'\|$$

which is always uniquely defined because $\mathcal{C}$ is closed and convex. The following proposition shows that all the convergence properties of the classical gradient descent caries over to this projected algorithm.

Theorem 30. *Theorems ?? and 23 still holds when replacing iterations (17.2) by (17.5).*

Proof. The proof of Theorem ?? extends because the projector is contractant, $\|\mathrm{Proj}_{\mathcal{C}}(x) - \mathrm{Proj}_{\mathcal{C}}(x')\| \leqslant \|x - x'\|$ so that the strict contraction properties of the gradient descent is maintained by this projection. □

The main bottleneck that often prevents to use (17.5) is that the projector is often complicated to compute. We are however lucky since for ℓ^1 mininization, one can apply in a straightforward manner this method.

17.2 Proximal Algorithm

For non-smooth functions f, it is not possible to perform an "explicit" gradient descent step because the gradient is not even defined. One thus needs to replace this "explicit" step by an "implicit" one, which is possible even if f is non-smooth.

17.2.1 Proximal Map

The implicit stepping of amplitude $\tau > 0$ is defined as

$$\forall\, x, \quad \mathrm{Prox}_{\tau f}(x) \overset{\text{def.}}{=} \operatorname*{argmin}_{x'} \frac{1}{2}\|x - x'\|^2 + f(x'). \tag{17.6}$$

It amounts to minimize function f locally around x, in a ball of radius controlled by τ. This the involved function $\frac{1}{2}\|x - \cdot\|^2 + f$ is strongly convex, this operator $\mathrm{Prox}_{\tau f}$ is well defined and single-valued.

When $f = \iota_{\mathcal{C}}$ is an indicator, the proximal map boils down to a projection $\mathrm{Prox}_{\iota_{\mathcal{C}}} = \mathrm{Proj}_{\mathcal{C}}$, it is thus in some sense a generalization of the projection to arbitrary function. And can also be interpreted as a projector on a level set of f. An interesting feature of the proximal map is that it is a contraction, thus generalizing the well-known property of projectors.

Proposition 57. *One has $\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\| \leqslant \|x - y\|$.*

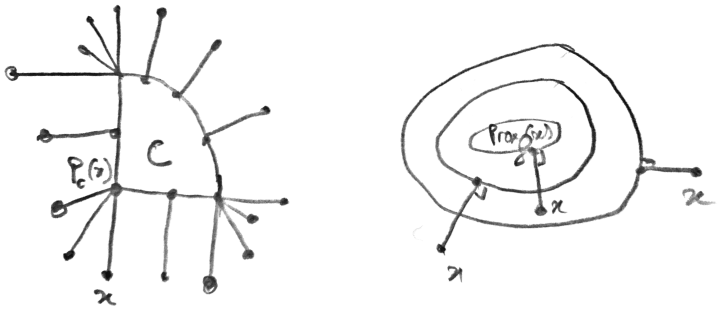


Figure 17.1: Proximal map and projection map.

Examples The following proposition states a few simples examples.

Proposition 58. *One has*

$$\mathrm{Prox}_{\frac{\tau}{2}\|\cdot\|^2}(x) = \frac{x}{1+\tau}, \quad \textit{and} \quad \mathrm{Prox}_{\tau\|\cdot\|_1} = \mathcal{S}_\tau^1(x), \tag{17.7}$$

where the soft-thresholding is defined as

$$\mathcal{S}_\tau^1(x) \overset{\text{def.}}{=} (S_\tau(x_i))_{i=1}^p \quad \textit{where} \quad S_\tau(r) \overset{\text{def.}}{=} \mathrm{sign}(r)(|r|-\lambda)_+,$$

(see also (9.5)). For $A \in \mathbb{R}^{P\times N}$, one has

$$\mathrm{Prox}_{\frac{\tau}{2}\|A\cdot-y\|^2}(x) = (\mathrm{Id}_N + \tau A^*A)^{-1}(x + \tau A^* y). \tag{17.8}$$

Proof. The proximal map of $\|\cdot\|_1$ was derived in Proposition 26. For the quadratic case

$$z = \mathrm{Prox}_{\frac{\tau}{2}\|A\cdot-y\|^2}(x) \quad \Leftrightarrow \quad z - x + \tau A^*(Az-y) = 0 \quad \Leftrightarrow \quad (\mathrm{Id}_N + \tau A^*A)z = x + \tau A^* y.$$

□

Note that in some case, the proximal map of a non-convex function is well defined, for instance $\mathrm{Prox}_{\tau\|\cdot\|_0}$ is the hard thresholding associated to the threshold $\sqrt{2\tau}$, see Proposition 26.

17.2.2 Basic Properties

We recap some useful proximal-calculus.

Proposition 59. *One has*

$$\mathrm{Prox}_{f+\langle y,\cdot\rangle} = y + \mathrm{Prox}_f, \quad \mathrm{Prox}_{f(\cdot-y)} = y + \mathrm{Prox}_f(\cdot - y).$$

If $f(x) = \sum_{k=1}^K f(x_k)$ for $x = (x_1, \ldots, x_K)$ is separable, then

$$\mathrm{Prox}_{\tau f}(x) = (\mathrm{Prox}_{\tau f_k}(x_k))_{k=1}^K. \tag{17.9}$$

Proof. One has

$$z = \mathrm{Prox}_{f+\langle y,\cdot\rangle}(x) \quad \Leftrightarrow \quad 0 \in x - z + (\partial f(x) + y) \quad \Leftrightarrow \quad 0 \in x - (z-y) + \partial f(x)$$

which is the optimality condition for $z - y = \mathrm{Prox}_f(x)$.

One has

$$z = \mathrm{Prox}_{f(\cdot-y)}(x) \quad \Leftrightarrow \quad 0 \in x - z + \lambda\partial f(x-y) \quad \Leftrightarrow \quad 0 \in x' - (z-y) + \partial f(x')$$

where we defined $x' \overset{\text{def.}}{=} x - y$, and this is the optimality condition for $z - y = \mathrm{Prox}_f(x')$ □

The following proposition is very useful.

Proposition 60. *If $A \in \mathbb{R}^{P\times N}$ is a tight frame, i.e. $AA^* = \mathrm{Id}_P$, then*

$$\mathrm{Prox}_{f\circ A} = A^* \circ \mathrm{Prox}_f \circ A + \mathrm{Id}_N - A^*A.$$

In particular, if A is orthogonal, then $\mathrm{Prox}_{f\circ A} = A^* \circ \mathrm{Prox}_f \circ A$.

17.2.3 Related Concepts

Link with sub-differential. For a set-valued map $U : \mathcal{H} \rightsquigarrow \mathcal{G}$, we define the inverse set-valued map $U^{-1} : \mathcal{G} \rightsquigarrow \mathcal{H}$ by

$$h \in U^{-1}(g) \quad \Longleftrightarrow \quad g \in U(h) \tag{17.10}$$

[ToDo: add picture] The following proposition shows that the proximal map is related to a regularized inverse of the sub-differential.

Proposition 61. *One has $\mathrm{Prox}_{\tau f} = (\mathrm{Id} + \tau\partial f)^{-1}$.*

Proof. One has the following equivalence

$$z = \mathrm{Prox}_{\tau f}(x) \Leftrightarrow 0 \in z - x + \tau\partial f(z) \Leftrightarrow x \in (\mathrm{Id} + \tau\partial f)(z) \Leftrightarrow z = (\mathrm{Id} + \tau\partial f)^{-1}(x)$$

where for the last equivalence, we have replace "$\in$" by "$=$" because the proximal map is single valued. □

The proximal operator is hence often referred to the "resolvent" $\mathrm{Prox}_{\tau f} = (\mathrm{Id} + \tau\partial f)^{-1}$ of the maximal monotone operator ∂f.

Link with duality. One has the following fundamental relation between the proximal operator of a function and of its Legendre-Fenchel transform

Theorem 31 (Moreau decomposition)**.** *One has*

$$\mathrm{Prox}_{\tau f} = \mathrm{Id} - \tau\,\mathrm{Prox}_{f^*/\tau}(\cdot/\tau).$$

This theorem shows that the proximal operator of f is simple to compute if and only the proximal operator of f^* is also simple. As a particular instantiation, since according to , one can re-write the soft thresholding as follow

$$\mathrm{Prox}_{\tau\|\cdot\|_1}(x) = x - \tau\,\mathrm{Proj}_{\|\cdot\|_\infty\leqslant 1}(x/\tau) = x - \mathrm{Proj}_{\|\cdot\|_\infty\leqslant\tau}(x) \quad \text{where} \quad \mathrm{Proj}_{\|\cdot\|_\infty\leqslant\tau}(x) = \min(\max(x, -\tau), \tau).$$

In the special case where $f = \iota_{\mathcal{C}}$ where $\mathcal{C}$ is a closed convex cone, then

$$(\iota_{\mathcal{C}})^* = \iota_{\mathcal{C}^\circ} \quad \text{where} \quad \mathcal{C}^\circ \overset{\text{def.}}{=} \{y\,;\, \forall\, x \in \mathcal{C}, \langle x, y\rangle \leqslant 0\} \tag{17.11}$$

and $\mathcal{C}^\circ$ is the so-called polar cone. Cones are fundament object in convex optimization because they are invariant by duality, in the sense of (17.11) (if $\mathcal{C}$ is not a cone, its Legendre transform would not be an indicator). Using (17.11), one obtains the celebrated Moreau polar decomposition

$$x = \mathrm{Proj}_{\mathcal{C}}(x) +^{\perp} \mathrm{Proj}_{\mathcal{C}^\circ}(x)$$

where "$+^{\perp}$" denotes an orthogonal sum (the terms in the sum are mutually orthogonal). **[ToDo: add drawing]** In the case where $\mathcal{C} = V$ is a linear space, this corresponds to the usual decomposition $\mathbb{R}^p = V \oplus^{\perp} V^{\perp}$.

Link with Moreau-Yosida regularization. The following proposition shows that the proximal operator can be interpreted as performing a gradient descent step on the Moreau-Yosida smoothed version f_μ of f, defined in (16.11).

Proposition 62. *One has*

$$\mathrm{Prox}_{\mu f} = \mathrm{Id} - \mu \nabla f_\mu.$$

17.3 Primal Algorithms

We now describe some important algorithm which assumes some structure (a so-called "splitting") of the minimized functional to be able to apply proximal maps on sub-functions. Note that there is obviously many ways to split or structure a given initial problem, so there are many non-equivalent ways to apply a given proximal-based method to solve the problem. Finding the "best" way to split a problem is a bit like black magic, and there is no definite answer. Also all there algorithm comes with step size and related parameters, and there is no obvious way to tune these parameters automatically (although some insight might be gained by studying convergence rate).

17.3.1 Proximal Point Algorithm

One has the following equivalence

$$x^\star \in \mathrm{argmin}\, f \quad \Leftrightarrow \quad 0 \in \partial f(x^\star) \quad \Leftrightarrow \quad x^\star \in (\mathrm{Id} + \tau \partial f)(x^\star) \tag{17.12}$$

$$\Leftrightarrow \quad x^\star = (\mathrm{Id} + \tau \partial f)^{-1}(x^\star) = \mathrm{Prox}_{\tau f}(x^\star). \tag{17.13}$$

This shows that being a minimizer of f is equivalent to being a fixed point of $\mathrm{Prox}_{\tau f}$. This suggest the following fixed point iterations, which are called the proximal point algorithm

$$x^{(\ell+1)} \overset{\text{def.}}{=} \mathrm{Prox}_{\tau_\ell f}(x^{(\ell)}). \tag{17.14}$$

On contrast to the gradient descent fixed point scheme, the proximal point method is converging for any sequence of steps.

Theorem 32. *If $0 < \tau_{\min} \leqslant \tau_\ell \leqslant \gamma_{\max} < +\infty$, then $x^{(\ell)} \to x^\star$ a minimizer of f.*

This implicit step (17.14) should be compared with a gradient descent step (17.2)

$$x^{(\ell+1)} \overset{\text{def.}}{=} (\mathrm{Id} + \tau_\ell \nabla f)(x^{(\ell)}).$$

One sees that the implicit resolvent $(\mathrm{Id} - \tau_\ell \partial f)^{-1}$ replaces the explicit step $\mathrm{Id} + \tau_\ell \nabla f$. For small τ_ℓ and smooth f, they are equivalent at first order. But the implicit step is well defined even for non-smooth function, and the scheme (the proximal point) is always convergent (whereas the explicit step size should be small enough for the gradient descent to converge). This is inline with the general idea the implicit stepping (e.g. implicit Euler for integrating ODE, which is very similar to the proximal point method) is more stable. Of course, the drawback is that explicit step are very easy to implement whereas in general proximal map are hard to solve (most of the time as hard as solving the initial problem).

17.3.2 Forward-Backward

It is in general impossible to compute $\mathrm{Prox}_{\gamma f}$ so that the proximal point algorithm is not implementable. In oder to derive more practical algorithms, it is important to restrict the class of considered function, by imposing some structure on the function to be minimized. We consider functions of the form

$$\min_x \mathcal{E}(x) \overset{\text{def.}}{=} f(x) + g(x) \tag{17.15}$$

where $g \in \Gamma_0(\mathcal{H})$ can be an arbitrary, but f needs to be smooth.

One can modify the fixe point derivation (17.12) to account for this special structure

$$\begin{aligned} x^\star \in \operatorname{argmin} f+g &\quad\Leftrightarrow\quad 0 \in \nabla f(x^\star) + \partial g(x^\star) \quad\Leftrightarrow\quad x^\star - \tau\nabla f(x^\star) \in (\mathrm{Id}+\tau\partial g)(x^\star) \\ &\quad\Leftrightarrow\quad x^\star = (\mathrm{Id}+\tau\partial g)^{-1} \circ (\mathrm{Id}-\tau\nabla f)(x^\star). \end{aligned}$$

This fixed point suggests the following algorithm, with the celebrated Forward-Backward

$$x^{(\ell+1)} \stackrel{\text{def.}}{=} \mathrm{Prox}_{\tau_\ell g}\left(x^{(\ell)} - \tau_\ell \nabla f(x^{(\ell)})\right). \tag{17.16}$$

Derivation using surrogate functionals. An intuitive way to derive this algorithm, and also a way to prove its convergence, it using the concept of surrogate functional.

To derive an iterative algorithm, we modify the energy $\mathcal{E}(x)$ to obtain a surrogate functional $\mathcal{E}(x, x^{(\ell)})$ whose minimization corresponds to a simpler optimization problem, and define the iterations as

$$x^{(\ell+1)} \stackrel{\text{def.}}{=} \operatorname*{argmin}_x \mathcal{E}(x, x^{(\ell)}). \tag{17.17}$$

In order to ensure convergence, this function should satisfy the following property

$$\mathcal{E}(x) \leqslant \mathcal{E}(x, x') \qutext{and} \quad \mathcal{E}(x,x) = \mathcal{E}(x) \tag{17.18}$$

and $\mathcal{E}(x) - \mathcal{E}(x,x')$ should be a smooth function. Property (17.18) guarantees that f is decaying by the iterations

$$\mathcal{E}(x^{(\ell+1)}) \leqslant \mathcal{E}(x^{(\ell)})$$

and it simple to check that actually all accumulation points of $(x^{(\ell)})_\ell$ are stationary points of f.

In order to derive a valid surrogate $\mathcal{E}(x,x')$ for our functional (17.15), since we assume f is L-smooth (i.e. satisfies $(\mathcal{R}_L)$), let us recall the quadratic majorant (13.18)

$$f(x) \leqslant f(x') + \langle \nabla f(x'), x'-x\rangle + \frac{L}{2}\|x-x'\|^2,$$

so that for $0 < \tau < \frac{1}{L}$, the function

$$\mathcal{E}(x,x') \stackrel{\text{def.}}{=} f(x') + \langle \nabla f(x'), x'-x\rangle + \frac{1}{2\tau}\|x-x'\|^2 + g(x) \tag{17.19}$$

satisfies the surrogate conditions (17.18). The following proposition shows that minimizing the surrogate functional corresponds to the computation of a so-called proximal operator.

Proposition 63. *The update* (17.17) *for the surrogate* (17.19) *is exactly* (17.16).

Proof. This follows from the fact that

$$\langle \nabla f(x'), x'-x\rangle + \frac{1}{2\tau}\|x-x'\|^2 = \frac{1}{2\tau}\|x-(x'-\tau\nabla f(x'))\|^2 + \text{cst}.$$

□

Convergence of FB. Although we impose $\tau < 1/L$ to ensure majorization property, one can actually show convergence under the same hypothesis as for the gradient descent, i.e. $0 < \tau < 2/L$, with the same convergence rates. This means that Theorem 30 for the projected gradient descent extend to FB.

Theorem 33. *Theorems* **??** *and 23 still holds when replacing iterations* (17.2) *by* (17.16).

Note furthermore that the projected gradient descent algorithm (17.5) is recovered as a special case of (17.16) when setting $J = \iota_{\mathcal{C}}$ the indicator of the constraint set, since $\mathrm{Prox}_{\rho J} = \mathrm{Proj}_{\mathcal{C}}$ in this case.

Of course the difficult point is to be able to compute in closed form $\mathrm{Prox}_{\tau g}$ in (17.16), and this is usually possible only for very simple function. We have already seen such an example in Section 14.1.4 for the resolution of ℓ^1-regularized inverse problems (the Lasso).

17.3.3 Douglas-Rachford

We consider here the structured minimization problem

$$\min_{x\in\mathbb{R}^p} f(x)+g(x), \tag{17.20}$$

but on contrary to the Forward-Backward setting studied in Section 17.3.2, no smoothness is imposed on f. We here suppose that we can compute easily the proximal map of f and g.

Example 3 (Constrained Lasso). An example of a problem of the form (17.20) where one can apply Douglas-Rachford is the noiseless constrained Lasso problem (9.11)

$$\min_{Ax=y} \|x\|_1$$

where one can use $f=\iota_{\mathcal{C}_y}$ where $\mathcal{C}_y \overset{\text{def.}}{=} \{x\;;\; Ax=y\}$ and $g=\|\cdot\|_1$. As noted in Section 9.3.1, this problem is equivalent to a linear program. The proximal operator of g is the soft thresholding as stated in (17.7), while the proximal operator of g is the orthogonal projector on the affine space $\mathcal{C}_y$, which can be computed by solving a linear system as stated in (16.9) (this is especially convenient for inpainting problems or deconvolution problem where this is achieved efficiently).

The Douglas-Rachford iterations read

$$\tilde{x}^{(\ell+1)} \overset{\text{def.}}{=} \Big(1-\frac{\mu}{2}\Big)\tilde{x}^{(\ell)} + \frac{\mu}{2}\,\mathrm{rProx}_{\tau g}(\mathrm{rProx}_{\tau f}(\tilde{x}^{(\ell)})) \quad\text{and}\quad x^{(\ell+1)} \overset{\text{def.}}{=} \mathrm{Prox}_{\tau f}(\tilde{x}^{(\ell+1)}), \tag{17.21}$$

where we have used the following shortcuts

$$\mathrm{rProx}_{\tau f}(x) = 2\,\mathrm{Prox}_{\tau f}(x)-x.$$

One can show that for any value of $\tau>0$, any $0<\mu<2$, and any $\tilde{x}_0$, $x^{(\ell)}\to x^\star$ which is a minimizer of $f+g$.

Note that it is of course possible to inter-change the roles of f and g, which defines another set of iterations.

More than two functions. Another sets of iterations can be obtained by "symetrizing" the algorithm. More generally, if we have K functions $(f_k)_k$, we re-write

$$\min_x \sum_k f_k(x) = \min_{X=(x_1,\ldots,x_k)} f(X)+g(X) \quad\text{where}\quad f(X)=\sum_k f_k(x_k) \quad\text{and}\quad g(X)=\iota_\Delta(X)$$

where $\Delta=\{X\;;\; x_1=\ldots=x_k\}$ is the diagonal. The proximal operator of f is

$$\mathrm{Prox}_{\tau f}(X)=\mathrm{Proj}_\Delta(X)=(\bar{x},\ldots,\bar{x}) \quad\text{where}\quad \bar{x}=\frac{1}{K}\sum_k x_k$$

while the proximal operator of f is easily computed from those of the $(f_k)_k$ using (17.9). One can thus apply DR iterations (17.21).

Handling a linear operator. One can handle a minimization of the form (17.23) by introducing extra variables

$$\inf_x f_1(x)+f_2(Ax) = \inf_{z=(x,y)} f(z)+g(z) \quad\text{where}\quad \begin{cases} f(z)=f_1(x)+f_2(y) \\ g(z)=\iota_{\mathcal{C}}(x,y), \end{cases}$$

where $\mathcal{C}=\{(x,y)\;;\; Ax=y\}$. This problem can be handled using DR iterations (17.21), since the proximal operator of f is obtained from those of (f_1,f_2) using (17.9), while the proximal operator of g is the projector on $\mathcal{C}$, which can be computed in two alternative ways as the following proposition shows.

Proposition 64. *One has*

$$\mathrm{Proj}_{\mathcal{C}}(x,y) = (x + A^*\tilde{y}, y - \tilde{y}) = (\tilde{x}, A\tilde{x}) \quad \textit{where} \quad \begin{cases} \tilde{y} \overset{\text{def.}}{=} (\mathrm{Id}_P + AA^*)^{-1}(Ax - y) \\ \tilde{x} \overset{\text{def.}}{=} (\mathrm{Id}_N + A^*A)^{-1}(A^*y + x). \end{cases} \tag{17.22}$$

Proof. **[ToDo: todo]** □

Remark 9 (Inversion of linear operator). At many places (typically to compute some sort of projector) one has to invert matrices of the form AA^*, A^*A, $\mathrm{Id}_P + AA^*$ or $\mathrm{Id}_N + A^*A$ (see for instance (17.22)). There are some case where this can be done efficiently. Typical examples where this is simple are inpainting inverse problem where AA^* is diagonal, and deconvolution or partial Fourier measurement (e.g. fMRI) for which A^*A is diagonalized using the FFT. If this inversion is too costly, one needs to use more advanced methods, based on duality, which allows to avoid trading the inverse A by the application of A^*. They are however typically converging more slowly.

17.4 Dual and Primal-Dual Algorithms

Convex duality, detailed in Section 16.2Â (either from the Lagrange or the Fenchel-Rockafellar point of view – which are essentially equivalent), is very fruitful to derive new optimization algorithm or to apply existing algorithm on a dual reformulation.

17.4.1 Forward-backward on the Dual

Let us illustrate first the idea of applying a known algorithm to the dual problem. We consider here the structured minimization problem associated to Fenchel-Rockafellar duality (16.12)

$$p^\star = \inf_x f(x) + g(Ax), \tag{17.23}$$

but furthermore assume that f is μ-strongly convex, and we assume for simplicity that both (f,g) are continuous. If f were also smooth (but it needs to be!), one could think about using the Forward-Backward algorithm (17.16). But the main issue is that in general $\mathrm{Prox}_{\tau g\circ A}$ cannot be computed easily even if one can compute $\mathrm{Prox}_{\tau g\circ A}$. An exception to this is when A is a tight frame, as exposed in Proposition 60, but in practice it is rarely the case.

Example 4 (TV denoising). A typical example, which was the one used by Antonin Chambolle [6] to develop this class of method, is the total varation denoising

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

where $\nabla x \in \mathbb{R}^{N\times d}$ is the gradient (a vector field) of a signal ($d=1$)or image ($d=2$) x, and $\|\cdot\|_{1,2}$ is the vectorial-ℓ^1 norm (also called $\ell^1-\ell^2$ norm), such that for a d-dimensional vector field $(v_i)_{i=1}^p$

$$\|v\|_{1,2} \overset{\text{def.}}{=} \sum_i \|v_i\|.$$

Here

$$f = \frac{1}{2}\|\cdot - y\|^2 \quad \text{and} \quad g = \lambda\|\cdot\|_{1,2}$$

so that f is $\mu = 1$ strongly convex, and one sets $A = \nabla$ the linear operator.

Applying Fenchel-Rockafellar Theorem 28 (since strong duality holds, all involved functions being continuous), one has that

$$p^\star = \sup_u \; -g^*(u) - f^*(-A^*u).$$

But more importantly, since f is μ-strongly convex, one has that f^* is smooth with a $1/\mu$-Lipschitz gradient. One can thus use the Forward-Backward algorithm (17.16) on (minus the energy of) this problem, which reads

$$u^{(\ell+1)} = \mathrm{Prox}_{\tau_k g^*}\left(u^{(\ell)} + \tau_k A \nabla f^*(A^* u^{(\ell)})\right).$$

To guarantee convergence, the step size τ_k should be smaller than $2/L$ where L is the Lipschitz constant of $A \circ \nabla f^* \circ A^*$, which is smaller than $\|A\|^2/\mu$.

Last but not least, one some (not necessarily unique) dual minimizer $u^\star$ is computed, the primal-dual relationships (16.17) ensures that one retrieves the unique primal minimizer $x^\star$ as

$$-A^* u^\star \in \partial f(x^\star) \quad \Leftrightarrow \quad x^\star \in (\partial f)^{-1}(-A^* u^\star) = \partial f^*(-A^* u^\star) \quad \Leftrightarrow \quad x^\star = \nabla f^*(-A^* u^\star)$$

where we used here the crucial fact that f^* is smooth.

Example 5 (TV denoising). In the particular case of the TV denoising problem, one has

$$g^* = \iota_{\|\cdot\|_{\infty,2} \leqslant \lambda} \quad \text{where} \quad \|v\|_{\infty,2} \overset{\text{def.}}{=} \max_i \|v_i\| \quad \Longrightarrow \quad \mathrm{Prox}_{\tau g^*}(u) = \left(\min(\|v_i\|, \lambda)\frac{v_i}{\|v_i\|}\right)$$

$$f^\star(h) = \frac{1}{2}\|h\|^2 + \langle h,\, y\rangle \quad \text{and} \quad \nabla f^\star(h) = h + y.$$

Furthermore, $\mu = 1$ and $A^*A = \Delta$ is the usual finite difference approximation of the Laplacian, so that $\|A\|^2 = \|\Delta\| = 4d$ where d is the dimension.

17.4.2 Primal-Dual Splitting

We now comeback to the more general structure problemÂ of the form (17.23), which we consider in primal-dual form as

$$\inf_x f(x) + g(Ax) = \sup_u \inf_x f(x) + \langle Ax,\, u\rangle - g^*(u), \tag{17.24}$$

but we do not suppose anymore that f is strongly convex.

A typical instance of such a problem is for the TV regularization of the inverse problem $\mathcal{K}x = y$, which corresponds to solving

$$\min_x \frac{1}{2}\|y - \mathcal{K}x\|^2 + \lambda\|\nabla x\|_{1,2}.$$

where $A = \nabla$, $f(x) = \frac{1}{2}\|y - \mathcal{K}\cdot\|^2$ and $g = \lambda\|\cdot\|_{1,2}$. Note however that with such a splitting, one will have to compute the proximal operator of f, which, following (17.8), requires inverting either $\mathrm{Id}_P + AA^*$ or $\mathrm{Id}_N + A^*A$, see Remark 9.

A standard primal-dual algorithm, which is detailed in [], reads

$$\begin{aligned} z^{(\ell+1)} &\overset{\text{def.}}{=} \mathrm{Prox}_{\sigma g^*}(z^{(\ell)} + \sigma A(\tilde{x}^{(\ell)}) \\ x^{(\ell+1)} &\overset{\text{def.}}{=} \mathrm{Prox}_{\tau f}(x^{(\ell)} - \tau A^*(z^{(\ell+1)})) \\ \tilde{x}^{(\ell)} &\overset{\text{def.}}{=} x^{(\ell+1)} + \theta(x^{(\ell+1)} - x^{(\ell)}) \end{aligned}$$

if $0 \leqslant \theta \leqslant 1$ and $\sigma\tau\|K\|^2 < 1$, then $x^{(\ell)}$ converges to a minimizer of (17.24) .

Bibliography

[1] Amir Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*. SIAM, 2014.

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] E. Candès and D. Donoho. New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities. *Commun. on Pure and Appl. Math.*, 57(2):219–266, 2004.

[5] E. J. Candès, L. Demanet, D. L. Donoho, and L. Ying. Fast discrete curvelet transforms. *SIAM Multiscale Modeling and Simulation*, 5:861–899, 2005.

[6] A. Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20:89–97, 2004.

[7] Antonin Chambolle, Vicent Caselles, Daniel Cremers, Matteo Novaga, and Thomas Pock. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 9(263-340):227, 2010.

[8] Antonin Chambolle and Thomas Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[9] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[10] Philippe G Ciarlet. Introduction à l'analyse numérique matricielle et à l'optimisation. 1982.

[11] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *SIAM Multiscale Modeling and Simulation*, 4(4), 2005.

[12] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. on Pure and Appl. Math.*, 57:1413–1541, 2004.

[13] D. Donoho and I. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, Dec 1994.

[14] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[15] M. Figueiredo and R. Nowak. An EM Algorithm for Wavelet-Based Image Restoration. *IEEE Trans. Image Proc.*, 12(8):906–916, 2003.

[16] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Birkhäuser Basel, 2013.

[17] Stephane Mallat. *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[18] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. on Pure and Appl. Math.*, 42:577–685, 1989.

[19] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[20] Gabriel Peyré. *L'algèbre discrète de la transformée de Fourier*. Ellipses, 2004.

[21] J. Portilla, V. Strela, M.J. Wainwright, and Simoncelli E.P. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Trans. Image Proc.*, 12(11):1338–1351, November 2003.

[22] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.

[23] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, Frank Lenzen, and L Sirovich. *Variational methods in imaging*. Springer, 2009.

[24] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[25] Jean-Luc Starck, Fionn Murtagh, and Jalal Fadili. *Sparse image and signal processing: Wavelets and related geometric multiscale analysis*. Cambridge university press, 2015.