Mathematics of X-ray Computed Tomography: Discretization, Regularization Theory and a Bit of Wavelets

Tatiana A. Bubba
Department of Mathematics and Computer Science, University of Ferrara
tatiana.bubba@unife.it

Physical Sciences Summer School
Puerto Varas, 5-8 January 2026

UNIVERSITÀ DEGLI STUDI DI FERRARA
- EX LABORE FRUCTUS -

MiVA

European Women in Mathematics

Brief Recap: The Tomographic Inverse Problem

Recall the tomographic inverse problem:

given noisy measurements $y^\delta = \mathcal{R}f$, determine f

where $\mathcal{R} : X \to Y$ is the Radon transform (linear forward operator), $f \in X$ is the unknown (quantity of interest), $y^\delta \in Y$ is the noisy measurement with $\epsilon \in Y$, such that $\|\epsilon\|_Y \leq \delta$, measurement noise.

Brief Recap: The Tomographic Inverse Problem

Recall the tomographic inverse problem:

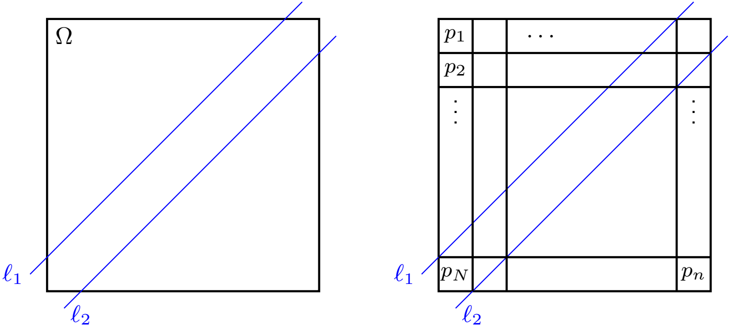
$$\text{given noisy measurements } y^\delta = \mathcal{R}f, \text{ determine } \; f$$

where $\mathcal{R} : X \to Y$ is the Radon transform (linear forward operator), $f \in X$ is the unknown (quantity of interest), $y^\delta \in Y$ is the noisy measurement with $\epsilon \in Y$, such that $\|\epsilon\|_Y \leq \delta$, measurement noise.

In practice, to represent the scanned object and projection data in a computer we need to discretize:

- **Sinogram** (measured data): already discrete (finite set of angles, finite set of detectors)
- **Object**: this discretization can be freely chosen, for instance pixels (in 2D) or voxels (in higher dimensions)
- **Forward model** (Radon transform): there are various approaches to discretize (compute or approximate) the line integrals

In Practice: Discretization



We discretise the domain Ω into $n = N^2$ pixels. Then, the approximation for the line integral over the line ℓ_i is given by:

$$y_i = \sum_{j=1}^{n} r_{ij} f_j \qquad \Longleftrightarrow \qquad \boldsymbol{y} = \boldsymbol{\mathcal{R}} \boldsymbol{f}$$

where $\boldsymbol{f} \in \mathbb{R}^n$, $\boldsymbol{y} \in \mathbb{R}^m$ and $\boldsymbol{\mathcal{R}} : \mathbb{R}^n \to \mathbb{R}^m$ with $\boldsymbol{\mathcal{R}} = (r_{ij})_{i,j=1,\dots,n}$, being r_{ij} the distance that the line ℓ_i travels in the jth pixel.

Briefly About Noise in CT

The intensity I_1 in the Beer-Lambert law ($-\log(I_1/I_0) = \int_\ell \mu(x)\,dx$) can be understood as the number of *transmitted* photons in a interval of time. This can be modelled as a Poisson random variable, namely the probability that k photons are transmitted is:

$$\mathbb{P}(\mathsf{x} = k) = \frac{\lambda^k}{k!}\,\mathrm{e}^{-\lambda}$$

where λ is the expected value of the number of transmitted photons.

Briefly About Noise in CT

The intensity I_1 in the Beer-Lambert law ($-\log(I_1/I_0) = \int_\ell \mu(x)\,dx$) can be understood as the number of *transmitted* photons in a interval of time. This can be modelled as a Poisson random variable, namely the probability that k photons are transmitted is:

$$\mathbb{P}(\mathsf{x} = k) = \frac{\lambda^k}{k!}\,\mathrm{e}^{-\lambda}$$

where λ is the expected value of the number of transmitted photons.

In practice, for large number of photons a Poisson distribution can be closely approximated by a **Gaussian distribution**, with constant mean and varying standard deviation (the standard deviation decays with increasing intensity).

This means that in practice in the discrete linear model of CT we consider Gaussian noise:

$$\boldsymbol{\mathcal{R}}\,\boldsymbol{f} \;=\; \boldsymbol{y} + \boldsymbol{\epsilon} \;=\; \boldsymbol{y}^\delta$$

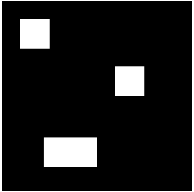
where $\boldsymbol{\epsilon} \in \mathbb{R}^m$, with $\|\boldsymbol{\epsilon}\| \leq \delta$ is the noise on the data.

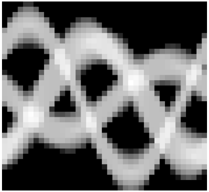
Direct and Inverse Problem: Discrete Version

Object f

Direct problem: $\mathcal{R}$

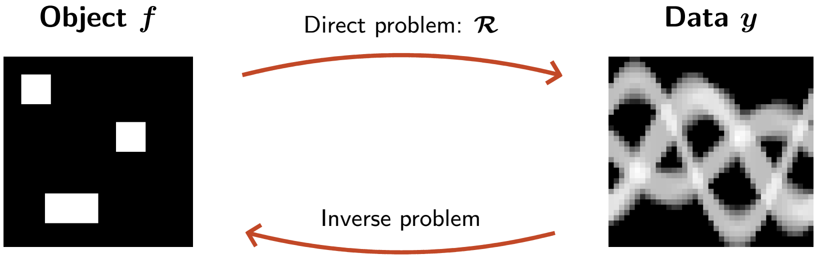
Data y





Direct problem: given object f, determine data y

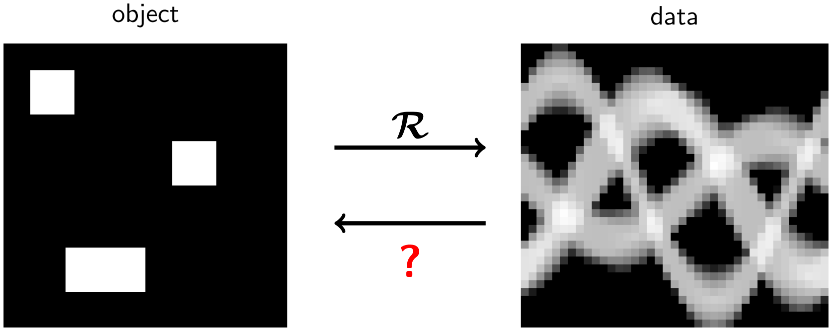
Direct and Inverse Problem: Discrete Version



Direct problem: given object $\boldsymbol{f}$, determine data $\boldsymbol{y}$

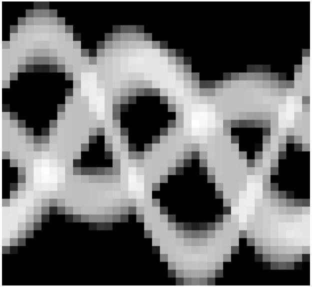
Inverse problem: given noisy data $\boldsymbol{y}^{\delta}$, recover object $\boldsymbol{f}$

How to Solve the Discrete Inverse Problem?



How to Solve the Discrete Inverse Problem?

data

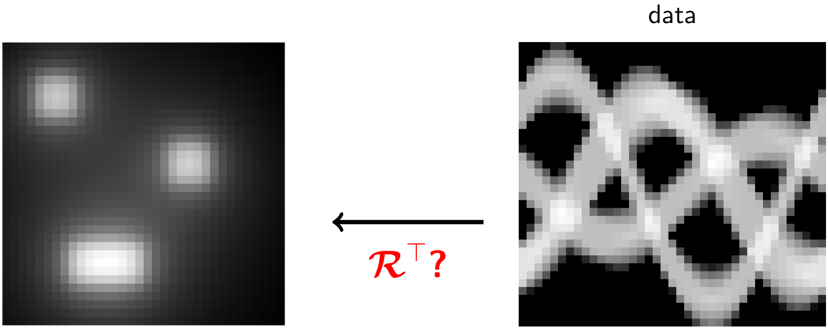


$\mathcal{R}^\top$?

Can we simply use the "discrete adjoint"? For matrices this is simply the transpose and its definition follows directly from that of the inner product:

$$\langle \boldsymbol{\mathcal{R}} \boldsymbol{g}, \boldsymbol{h} \rangle = (\boldsymbol{\mathcal{R}}^\top \boldsymbol{g})^\top \boldsymbol{h} = \boldsymbol{g}^\top \boldsymbol{\mathcal{R}}^\top \boldsymbol{h} = \boldsymbol{g}^\top (\boldsymbol{\mathcal{R}}^\top \boldsymbol{h}) = \langle \boldsymbol{g}, \boldsymbol{\mathcal{R}}^\top \boldsymbol{h} \rangle.$$

How to Solve the Discrete Inverse Problem?

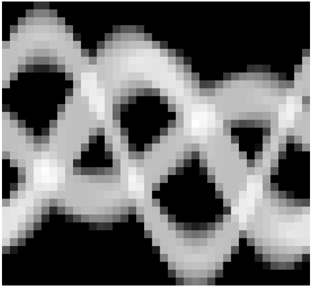


Can we simply use the "discrete adjoint"? For matrices this is simply the transpose and its definition follows directly from that of the inner product:

$$\langle \boldsymbol{\mathcal{R}} \boldsymbol{g}, \boldsymbol{h} \rangle = (\boldsymbol{\mathcal{R}}^\top \boldsymbol{g})^\top \boldsymbol{h} = \boldsymbol{g}^\top \boldsymbol{\mathcal{R}}^\top \boldsymbol{h} = \boldsymbol{g}^\top (\boldsymbol{\mathcal{R}}^\top \boldsymbol{h}) = \langle \boldsymbol{g}, \boldsymbol{\mathcal{R}}^\top \boldsymbol{h} \rangle.$$

How to Solve the Discrete Inverse Problem?

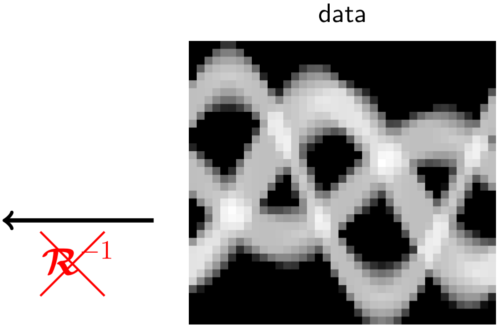
data



$\mathcal{R}^{-1}$?

Can we use the inverse?

How to Solve the Discrete Inverse Problem?



Can we use the inverse? This is in general not even defined ($\mathcal{R}$ is not a square matrix)!

Recall the Singular Value Decomposition (SVD)

If $\boldsymbol{\mathcal{R}} \in \mathbb{R}^{m\times n}$, there exist unitary matrices $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ such that:

$$\boldsymbol{\mathcal{R}} = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^\top = \mathbf{U}\begin{pmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_{\min(m,n)} \end{pmatrix}\mathbf{V}^\top$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(m,n)} \geq 0$ are the so-called **singular values** of $\boldsymbol{\mathcal{R}}$ and if we write:

$$\mathbf{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) \qquad \text{and} \qquad \mathbf{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)$$

the $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are, respectively, the left and right **singular vectors** associated with σ_i, for $i = 1, \ldots, \min(m,n)$.

The SVD can be written also as a sum of matrices of rank equal to 1:

$$\boldsymbol{\mathcal{R}} = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^\top = \sum_{i=1}^{\min(m,n)} \sigma_i\,\boldsymbol{u}_i\boldsymbol{v}_i^\top$$

The Moore-Penrose Pseudoinverse

The Moore-Penrose pseudoinverse is defined to be the matrix:

$$\boldsymbol{\mathcal{R}}^\dagger = \mathbf{V}\,\mathbf{\Sigma}^\dagger\,\mathbf{U}^\top = \mathbf{V}\begin{pmatrix}\Sigma_k^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}\mathbf{U}^\top$$

where

$$\Sigma_k = \begin{pmatrix}\sigma_1 & 0 & \dots & 0\\ 0 & \sigma_2 & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & \sigma_k\end{pmatrix}.$$

In particular, we have:

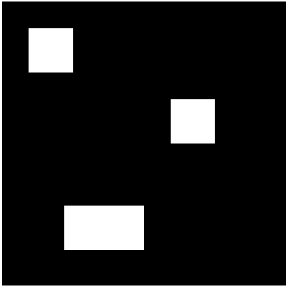
$$\boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y} \perp \ker(\boldsymbol{\mathcal{R}})$$

and

$$\|\boldsymbol{\mathcal{R}}\|_2 = \sigma_1, \quad \|\boldsymbol{\mathcal{R}}^\dagger\|_2 = \frac{1}{\sigma_k} \quad \text{and} \quad \mathrm{cond}(\boldsymbol{\mathcal{R}}) = \|\boldsymbol{\mathcal{R}}\|_2\,\|\boldsymbol{\mathcal{R}}^\dagger\|_2 = \frac{\sigma_1}{\sigma_k}.$$

Can We Use the Pseudoinverse to Solve Our Inverse Problem?

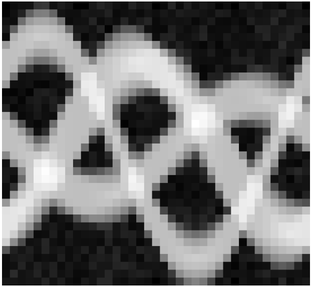
object



Original phantom: values between 0 (black) and 1 (white)

$$\xrightarrow{\mathcal{R}}$$

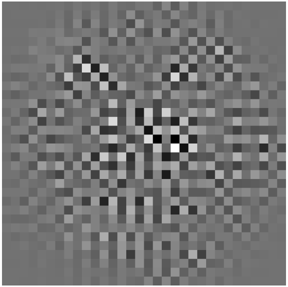
data



Data with 0.1% relative noise

Can We Use the Pseudoinverse to Solve Our Inverse Problem?

object

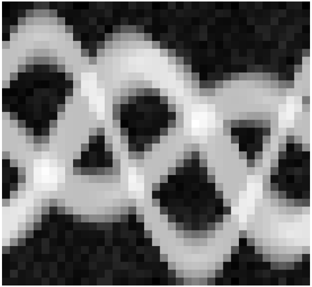


Pseudoinverse reconstruction:
values between -14.9 and 18.5

$$\xrightarrow{\mathcal{R}}$$

$$\xleftarrow[\mathcal{R}^\dagger]{}$$

data



Data with 0.1% relative noise

The Problem is Still Ill-posedness

Put it simply, we have:

- Surjectivity (existence): this fails when $m > n$ (more measurements than image pixels) since $\mathrm{range}(\boldsymbol{\mathcal{R}}) < m$ and $\mathrm{range}(\boldsymbol{\mathcal{R}})^\perp$ is nontrivial.
- Injectivity (uniqueness): this fails when $m < n$ (more unknowns than measurements) since dim$(\ker(\boldsymbol{\mathcal{R}})) > 0$.
- Stability: this fails when the condition number of $\boldsymbol{\mathcal{R}}$ is large.

The Problem is Still Ill-posedness

Put it simply, we have:

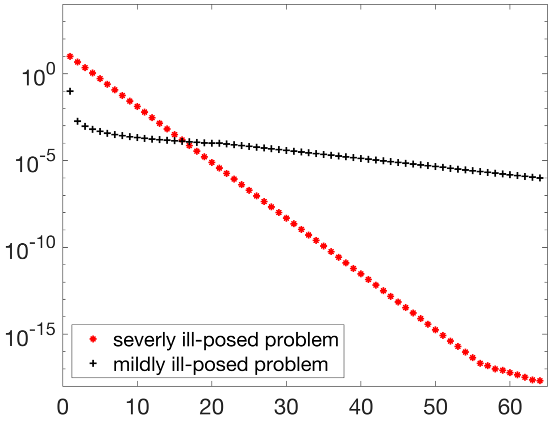
- Surjectivity (existence): this fails when $m > n$ (more measurements than image pixels) since $\mathrm{range}(\boldsymbol{\mathcal{R}}) < m$ and $\mathrm{range}(\boldsymbol{\mathcal{R}})^\perp$ is nontrivial.
- Injectivity (uniqueness): this fails when $m < n$ (more unknowns than measurements) since $\dim(\ker(\boldsymbol{\mathcal{R}})) > 0$.
- Stability: this fails when the condition number of $\boldsymbol{\mathcal{R}}$ is large.

The Moore-Penrose pseudoinverse takes care only of conditions 1 and 2, that is there **always** exists a minimum norm solution given by:

$$\boldsymbol{f}^\dagger = \boldsymbol{\mathcal{R}}^\dagger \boldsymbol{y}^\delta = \sum_{i=1}^{k} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i.$$

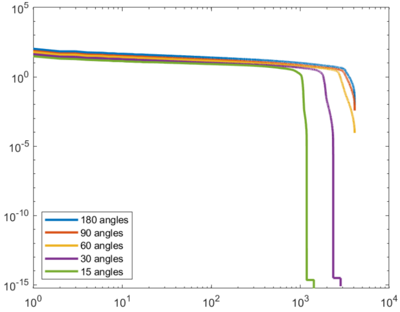
Condition 3 is trickier and requires a deeper understanding of its connection to the SVD.

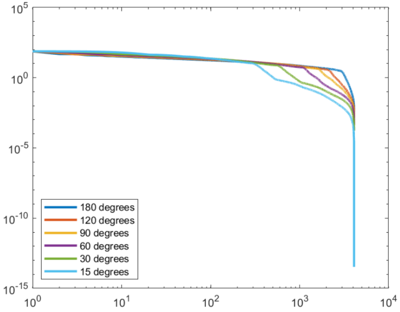
Ill-posedness & SVD



The singular values decay to zero, with no gap in the spectrum. The decay rate determines how difficult the problem is to solve.

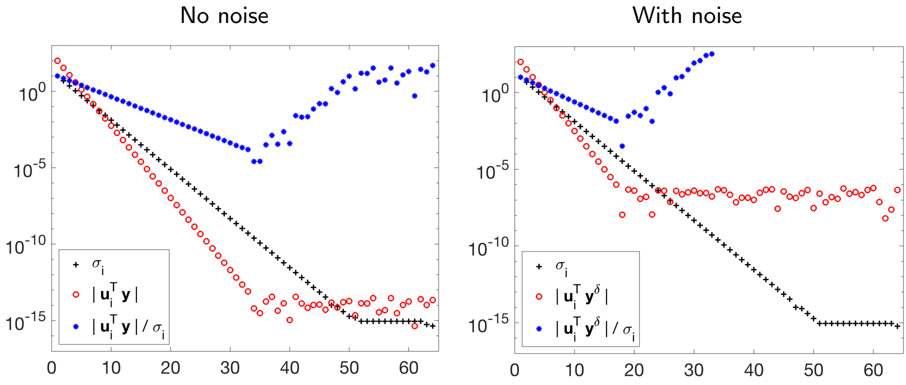
Ill-posedness & SVD: Limited Data Problems





- The discretization of the Radon transform (with dense angular sampling) gives a mildly ill-posed problem.
- Singular values decrease faster with sparse (left) and limited (right) angles.

Ill-posedness & SVD: the Role of Noise



When there is no noise, the singular values σ_i and the coefficients $|\boldsymbol{u}_i^\top \boldsymbol{y}|$ both level off at the machine precision.

When there is noise, the coefficients $|\boldsymbol{u}_i^\top \boldsymbol{y}^\delta|$ level off at the noise level, and only a few SVD components are reliable.

Ill-Conditioned Problems

Discrete ill-posed problems are characterized by system matrices with a large condition number: these problems are called ill-conditioned. This translates into the solution being very **sensitive** to errors in the data.

If $\boldsymbol{y}$ are the noiseless data and $\boldsymbol{y}^{\delta} = \boldsymbol{y} + \boldsymbol{\epsilon}$ the noisy data, classical perturbation theory leads to the bound:

$$\frac{\|\boldsymbol{f}^{\mathrm{gt}} - \boldsymbol{f}\|_2}{\|\boldsymbol{f}^{\mathrm{gt}}\|_2} \leq \mathrm{cond}(\boldsymbol{\mathcal{R}}) \frac{\delta}{\|\boldsymbol{f}^{\mathrm{gt}}\|_2}$$

Since $\mathrm{cond}(\boldsymbol{\mathcal{R}}) = \sigma_1/\sigma_k$ is large, the pseudoinverse solution $\boldsymbol{f}^{\dagger} = \boldsymbol{\mathcal{R}}^{\dagger}\boldsymbol{y}^{\delta}$ can be very far from the true solution $\boldsymbol{f}^{\mathrm{gt}}$.

Illustration of Ill-conditioning of Tomography

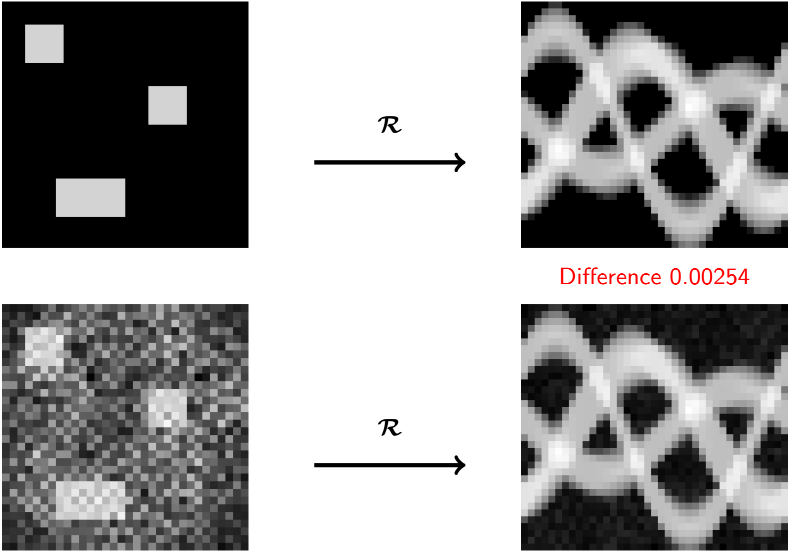


Illustration of Ill-conditioning of Tomography

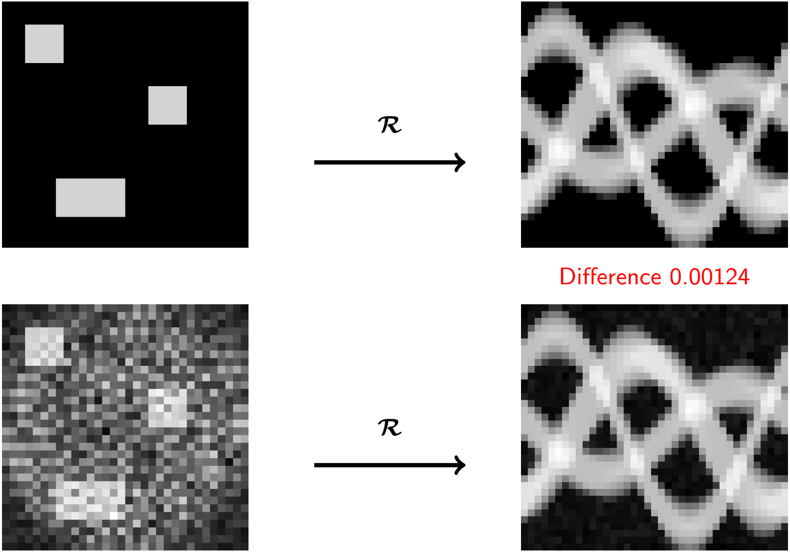
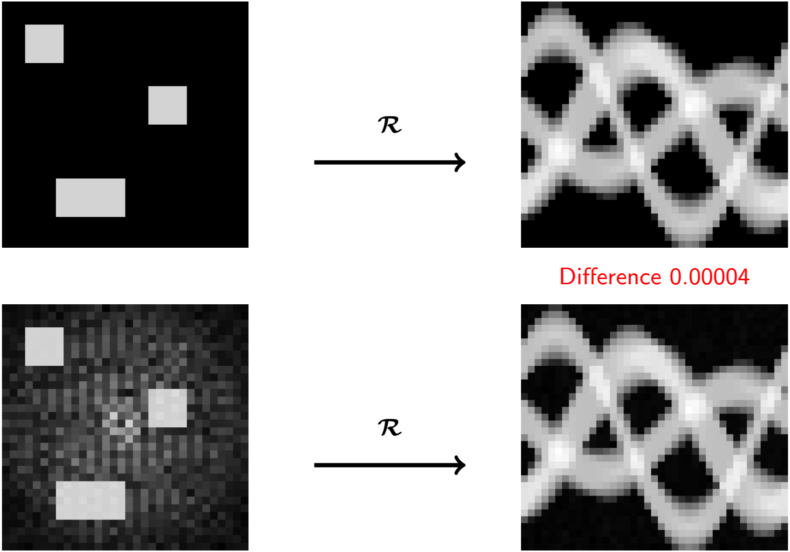
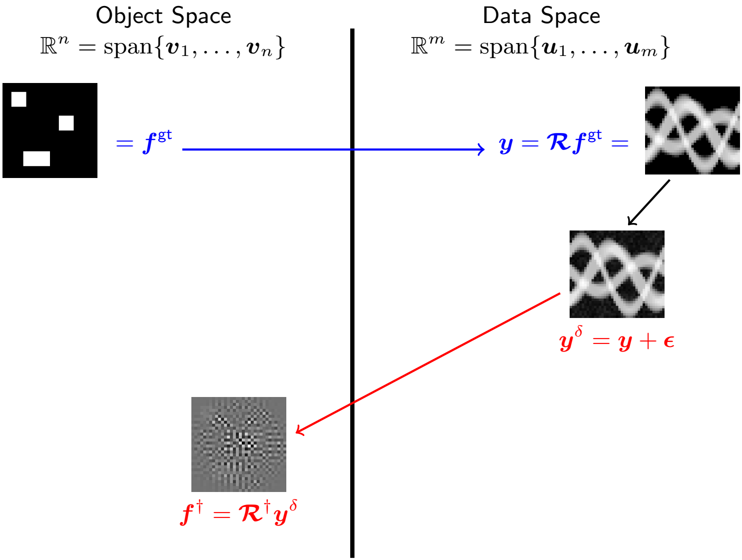


Illustration of Ill-conditioning of Tomography



The Geometry of Ill-Conditioned Problems



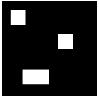
How to Cure Ill-posedness?

Object Space

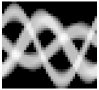
$\mathbb{R}^n = \mathrm{span}\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_n\}$

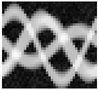
Data Space

$\mathbb{R}^m = \mathrm{span}\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_m\}$



$= \boldsymbol{f}^{\mathrm{gt}} \longrightarrow \boldsymbol{y} = \boldsymbol{\mathcal{R}} \boldsymbol{f}^{\mathrm{gt}} =$





$\boldsymbol{y}^{\delta} = \boldsymbol{y} + \epsilon$

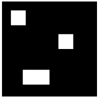
How to Cure Ill-posedness? Robust Solution: Regularization

Object Space

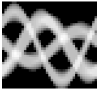
$\mathbb{R}^n = \mathrm{span}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$

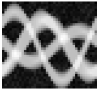
Data Space

$\mathbb{R}^m = \mathrm{span}\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m\}$



$= \boldsymbol{f}^{\mathrm{gt}} \longrightarrow \boldsymbol{y} = \boldsymbol{\mathcal{R}} \boldsymbol{f}^{\mathrm{gt}} =$





$\boldsymbol{y}^\delta = \boldsymbol{y} + \epsilon$

$\Gamma_{\alpha(\delta)}(\boldsymbol{y}^\delta)$

We need to define a family of continuous functions $\Gamma_\alpha : \mathbb{R}^m \to \mathbb{R}^n$ so that the reconstruction error $\|\Gamma_{\alpha(\delta)}(\boldsymbol{y}^\delta) - \boldsymbol{f}^{\mathrm{gt}}\|_2$ vanishes asymptotically as $\delta \to 0$.

Regularization Theory of Inverse Problems

A classic strategy is the theory of regularization of inverse problems, which applies to more general inverse problems with any (linear) forward operator $\boldsymbol{\mathcal{A}} : X \to Y$. It requires two ingredients:

(1) A family of regularization functionals $\Gamma_\alpha : \mathbb{R}^m \to \mathbb{R}^n$ varying with $\alpha > 0$:

$$\Gamma_\alpha \boldsymbol{y} \to \boldsymbol{\mathcal{A}}^\dagger \boldsymbol{y} \quad \text{as } \alpha \to 0, \qquad \forall \boldsymbol{y} \in \mathbb{R}^m.$$

We only focus on **linear** regularization functionals, which can therefore be represented by a matrix in $\mathbb{R}^{m \times n}$.

(2) A suitable parameter choice rule $\alpha = \alpha(\delta)$ ensuring

$$\Gamma_{\alpha(\delta)} \boldsymbol{y}^\delta \to \boldsymbol{\mathcal{A}}^\dagger \boldsymbol{y} \quad \text{as } \delta \to 0, \qquad \forall \boldsymbol{y}^\delta \ : \ \|\boldsymbol{y}^\delta - \boldsymbol{y}\| \le \delta \quad \forall \boldsymbol{y} \in \mathbb{R}^m$$

A desirable property is that Γ_α are **more stable** than $\boldsymbol{\mathcal{A}}^\dagger$: $\|\Gamma_\alpha\| \le \|\boldsymbol{\mathcal{A}}^\dagger\| \; \forall \alpha > 0$.

Remark

A parameter choice $\alpha = \alpha(\delta)$ is defined a priori, since it holds for any $\boldsymbol{y}$ and any perturbation $\boldsymbol{y}^\delta$. There are also a posteriori (heuristic) rules $\alpha = \alpha(\delta, \boldsymbol{y}^\delta)$.

Regularization 101: Back to SVD

In the following we always assume $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{R}}$, but the regularization strategies we introduce apply to more general inverse problems with any (linear) forward operator $\boldsymbol{\mathcal{A}} : \mathbb{R}^n \to \mathbb{R}^m$.

Regularization 101: Back to SVD

In the following we always assume $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{R}}$, but the regularization strategies we introduce apply to more general inverse problems with any (linear) forward operator $\boldsymbol{\mathcal{A}} : \mathbb{R}^n \to \mathbb{R}^m$.

Recall that the Moore-Penrose pseudoinverse solution is given by

$$\boldsymbol{f}^\dagger = \sum_{i=1}^{k} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i.$$

When dealing with **noisy data**, we have:

$$\boldsymbol{u}_i^\top \boldsymbol{y}^\delta = \boldsymbol{u}_i^\top \boldsymbol{y} + \boldsymbol{u}_i^\top \epsilon \approx \begin{cases} \boldsymbol{u}_i^\top \boldsymbol{y} & |\boldsymbol{u}_i^\top \boldsymbol{y}| > |\boldsymbol{u}_i^\top \epsilon| \\ \boldsymbol{u}_i^\top \epsilon & |\boldsymbol{u}_i^\top \boldsymbol{y}| < |\boldsymbol{u}_i^\top \epsilon| \end{cases}$$

Notice that the "noisy" components $|\boldsymbol{u}_i^\top \boldsymbol{y}^\delta|$ are those for which $|\boldsymbol{u}_i^\top \boldsymbol{y}|$ is small and they correspond to the smallest singular values σ_i.

Regularization 101: Truncated SVD (TSVD)

The simplest regularization technique is to discard the SVD coefficients corresponding to the smallest singular values. This is called truncated SVD:

$$\boldsymbol{f}^{\mathsf{TSVD}} = \sum_{i=1}^{r} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i$$

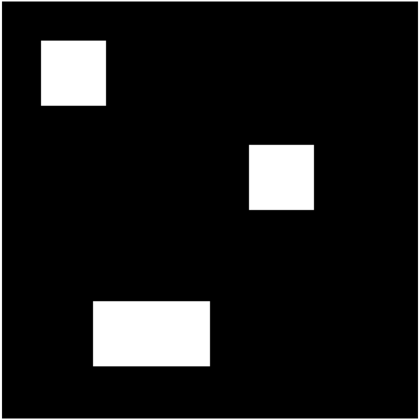
where the truncation parameter r is dictated by the coefficients $|\boldsymbol{u}_i^\top \boldsymbol{y}^\delta|$, **not** the singular values.

In practice, r has to be chosen as the index i where $|\boldsymbol{u}_i^\top \boldsymbol{y}^\delta|$ start to "level off" due to the noise.

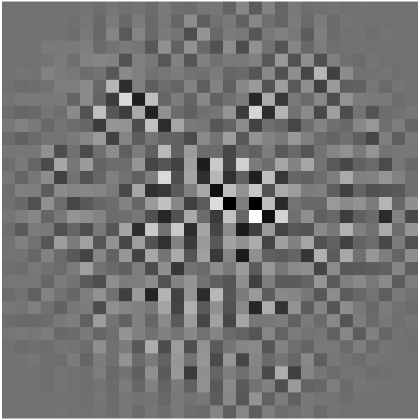
It is clear that the condition number for the TSVD solution is:

$$\frac{\sigma_1}{\sigma_r} < \frac{\sigma_1}{\sigma_k} = \mathrm{cond}(\boldsymbol{\mathcal{R}}).$$

Naive Reconstruction (Moore-Penrose Pseudoinverse)

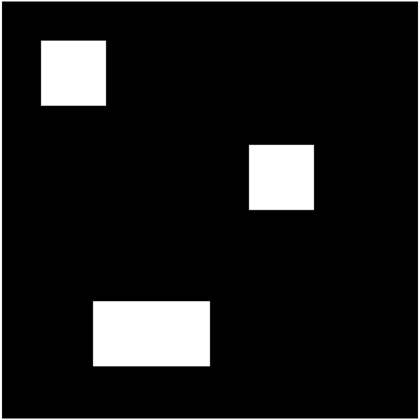


Original phantom

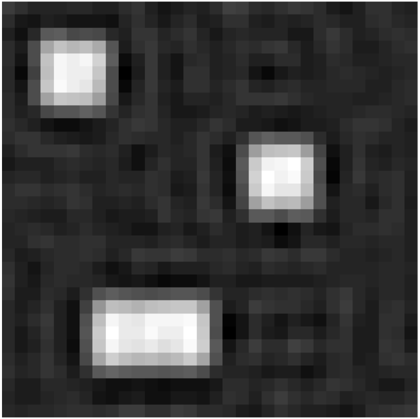


$f^\dagger$: RE $= 100\%$

Truncated SVD

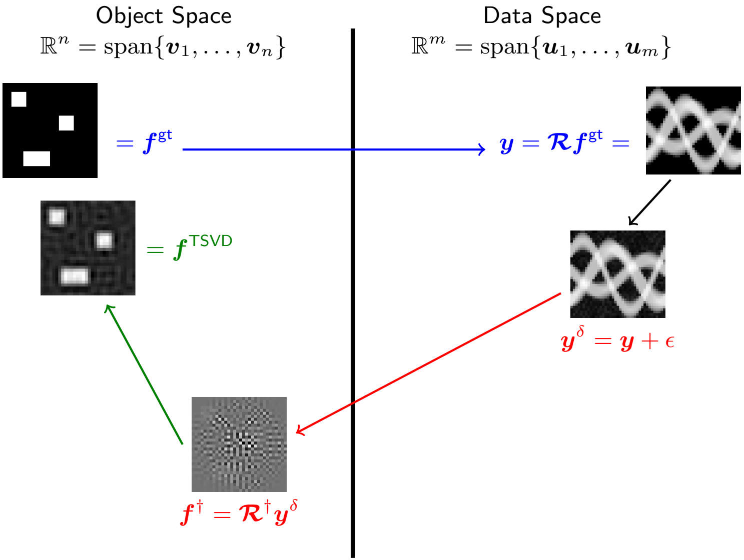


Original phantom



$\boldsymbol{f}^{\mathrm{TSVD}}$: RE $= 35\%$

Where TSVD Stands in The Geometry of Ill-Conditioned Problems



TSVD as Spectral Filtering

We can regard the TSVD also as the result of a filtering operation, namely:

$$\boldsymbol{f}^{\mathsf{TSVD}} = \sum_{i=1}^{r} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i = \sum_{i=1}^{\min(m,n)} \phi_i^{\mathsf{TSVD}} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i$$

where r is the truncation parameter and

$$\phi_i^{\mathsf{TSVD}} = \begin{cases} 1 & i = 1, \ldots, r \\ 0 & \text{elsewhere} \end{cases}$$

are the **filter factors** associated with the method.

These are called **spectral filtering** methods because the SVD basis can be regarded as a spectral basis, since the vectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are the eigenvectors of $\boldsymbol{\mathcal{R}}^\top \boldsymbol{\mathcal{R}}$ and $\boldsymbol{\mathcal{R}}\boldsymbol{\mathcal{R}}^\top$.

Another Filtering Strategy: Tikhonov Regularization

Let's now consider the following filter factors:

$$\phi_i^{\mathsf{TIKH}} = \begin{cases} \frac{\sigma_i^2}{\sigma_i^2+\alpha^2} & i = 1, \dots, \min(m, n) \\ 0 & \text{else} \end{cases}$$

which yield the reconstruction method:

$$\boldsymbol{f}^{\mathsf{TIKH}} = \sum_{i=1}^{\min(m,n)} \phi_i^{\mathsf{TIKH}} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i = \sum_{i=1}^{\min(m,n)} \frac{\sigma_i \left(\boldsymbol{u}_i^\top \boldsymbol{y}^\delta\right)}{\sigma_i^2 + \alpha^2} \boldsymbol{v}_i.$$

This choice of the filters result in a regularization technique called **Tikhonov regularization** and $\alpha > 0$ is the so-called regularization parameter.

The parameter α acts in the same way as the parameter r in the TSVD method: it controls which/how many SVD components we want to damp or filter.

Tikhonov Regularization: Minimization Formulation

- Recall that the Moore-Penrose pseudoinverse solution is given by:

$$\boldsymbol{f}^\dagger = \sum_{i=1}^{m} \frac{\boldsymbol{u}_i^\top \boldsymbol{y}^\delta}{\sigma_i} \boldsymbol{v}_i = \underset{\boldsymbol{f}}{\operatorname{argmin}} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta \right\|_2^2$$

where the last step states that is equivalent to finding the least squares solution. We want $\left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta \right\|_2^2$ small, but also avoid that becomes 0!

- Let's look also at the norm of the Moore-Penrose solution $\boldsymbol{f}^\dagger$:

$$\|\boldsymbol{f}^\dagger\|_2^2 = \sum_{i=1}^{m} \frac{(\boldsymbol{u}_i^\top \boldsymbol{y}^\delta)^2}{\sigma_i^2}.$$

This can become unrealistically large when the magnitude of the noise in some direction $\boldsymbol{u}_i$ greatly exceeds the magnitude of the singular value σ_i.

Tikhonov Regularization: Minimization Formulation

- Recall that the Moore-Penrose pseudoinverse solution is given by:

$$\boldsymbol{f}^{\dagger} = \sum_{i=1}^{m} \frac{\boldsymbol{u}_i^{\top}\boldsymbol{y}^{\delta}}{\sigma_i}\boldsymbol{v}_i = \operatorname*{argmin}_{\boldsymbol{f}} \left\|\boldsymbol{\mathcal{R}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\right\|_2^2$$

  where the last step states that is equivalent to finding the least squares solution. We want $\left\|\boldsymbol{\mathcal{R}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\right\|_2^2$ small, but also avoid that becomes 0!

- Let's look also at the norm of the Moore-Penrose solution $\boldsymbol{f}^{\dagger}$:

$$\|\boldsymbol{f}^{\dagger}\|_2^2 = \sum_{i=1}^{m} \frac{(\boldsymbol{u}_i^{\top}\boldsymbol{y}^{\delta})^2}{\sigma_i^2}.$$

  This can become unrealistically large when the magnitude of the noise in some direction $\boldsymbol{u}_i$ greatly exceeds the magnitude of the singular value σ_i.

Tikhonov regularization combines these two desiderata by looking at solving the minimization problem:

$$\boldsymbol{f}^{\mathsf{TIKH}} = \operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \left\{\left\|\boldsymbol{\mathcal{R}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\right\|_2^2 + \alpha\left\|\boldsymbol{f}\right\|_2^2\right\}.$$

Tikhonov Regularization

In particular, by looking at the minimization problem

$$\boldsymbol{f}^{\text{TIKH}} = \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\left\{ \left\|\boldsymbol{\mathcal{R}}\boldsymbol{f} - \boldsymbol{y}^{\delta}\right\|_2^2 + \alpha\left\|\boldsymbol{f}\right\|_2^2 \right\}$$

we notice that:

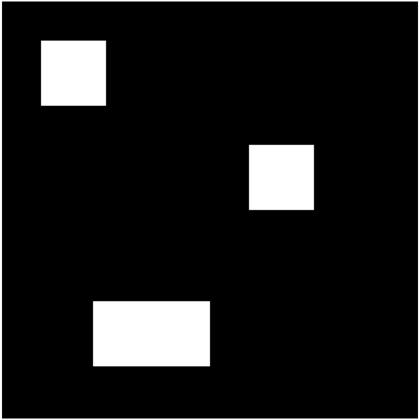
- By selecting $\alpha = 0$ we retrieve the Moore-Pensore solution $\boldsymbol{f}^{\dagger}$.
- By taking $\alpha \to \infty$, the solution of the minimization problem tends to $\boldsymbol{f} = \mathbf{0}$: Tikhonov regularization penalizes solutions with large norms.

Therefore, by tuning the regularization parameter $\alpha > 0$ we can ensure that:

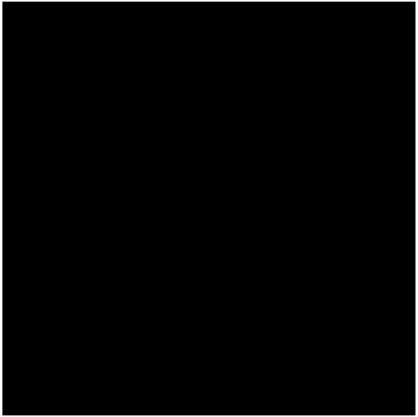
- The residual $\boldsymbol{\mathcal{R}}\boldsymbol{f}^{\text{TIKH}} - \boldsymbol{y}^{\delta}$ is small, but it does not become zero.
- The norm of the solution $\boldsymbol{f}^{\text{TIKH}}$ is bounded.

In general, choosing the regularization parameter α is not a trivial task and there is no rule of thumbs. Usually, it is a combination of good heuristics and prior knowledge of the noise in the observations.

Influence of the Choice of α in Tikhonov Regularization

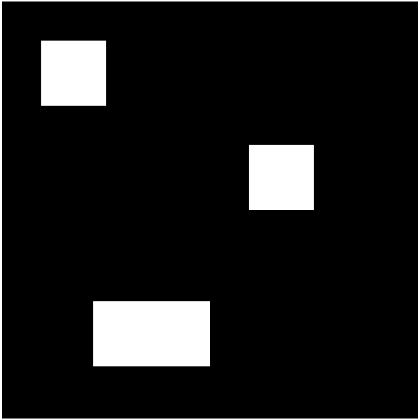


Original phantom

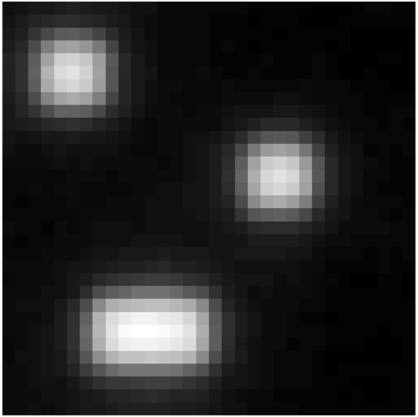


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10^3$

Influence of the Choice of α in Tikhonov Regularization

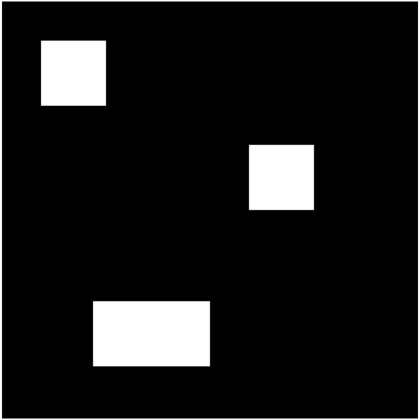


Original phantom

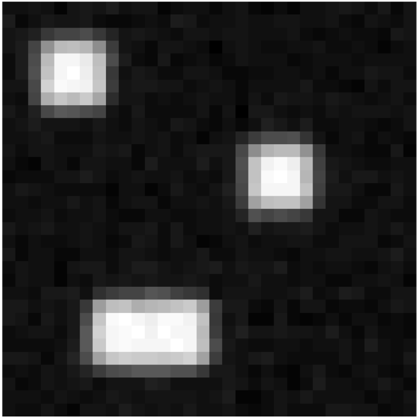


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10^2$

Influence of the Choice of α in Tikhonov Regularization

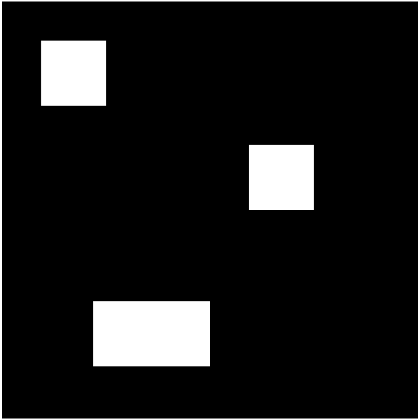


Original phantom

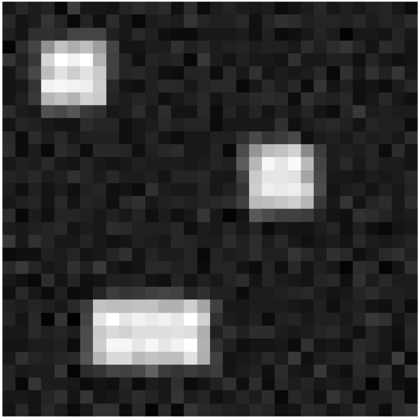


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10$

Influence of the Choice of α in Tikhonov Regularization

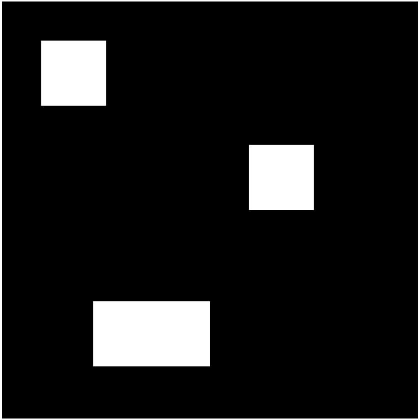


Original phantom

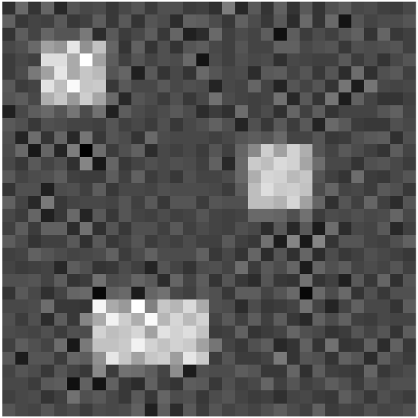


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 1$

Influence of the Choice of α in Tikhonov Regularization

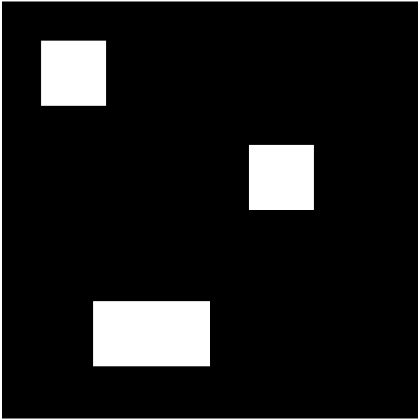


Original phantom

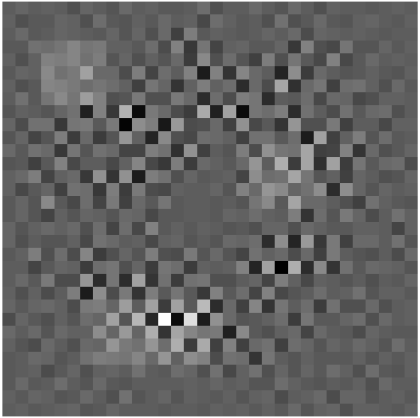


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10^{-1}$

Influence of the Choice of α in Tikhonov Regularization

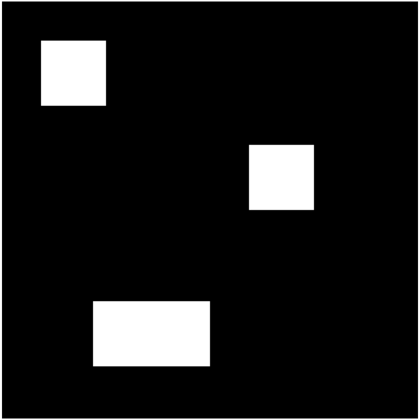


Original phantom

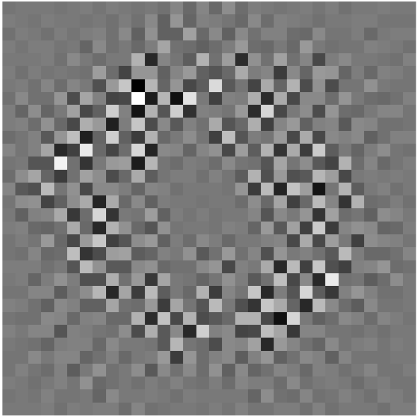


$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10^{-2}$

Influence of the Choice of α in Tikhonov Regularization



Original phantom



$\boldsymbol{f}^{\mathsf{TIKH}}$: $\alpha = 10^{-3}$

Normal Equation and Stacked Form for Tikhonov Regularization

The Tikhonov solution can be also formulated as a linear least squares problem:

$$\boldsymbol{f}^{\mathsf{TIKH}} = \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \; \frac{1}{2}\left\{ \left\| \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \sqrt{\alpha}\,\boldsymbol{I}_n \end{bmatrix} \boldsymbol{f} - \begin{bmatrix} \boldsymbol{y}^\delta \\ \boldsymbol{0} \end{bmatrix} \right\|_2^2 \right\}.$$

This is called **stacked form**. If we denote by $\widetilde{\boldsymbol{\mathcal{R}}} = \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \sqrt{\alpha}\,\boldsymbol{I}_n \end{bmatrix}$ and $\widetilde{\boldsymbol{y}}^\delta = \begin{bmatrix} \boldsymbol{y}^\delta \\ \boldsymbol{0} \end{bmatrix}$ then the least square solution of the stacked form satisfies the normal equations:

$$\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{\mathcal{R}}}\,\boldsymbol{f} = \widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{y}}^\delta.$$

It is easy to check that $\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{\mathcal{R}}} = \boldsymbol{\mathcal{R}}^\top \boldsymbol{\mathcal{R}} + \alpha\,\boldsymbol{I}_n$ and $\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{y}}^\delta = \boldsymbol{\mathcal{R}}^\top \boldsymbol{y}^\delta$, which yields the following result.

Normal Equation and Stacked Form for Tikhonov Regularization

The Tikhonov solution can be also formulated as a linear least squares problem:

$$\boldsymbol{f}^{\text{TIKH}} = \operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \frac{1}{2}\left\{ \left\| \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \sqrt{\alpha}\,\boldsymbol{I}_n \end{bmatrix} \boldsymbol{f} - \begin{bmatrix} \boldsymbol{y}^\delta \\ \boldsymbol{0} \end{bmatrix} \right\|_2^2 \right\}.$$

This is called **stacked form**. If we denote by $\widetilde{\boldsymbol{\mathcal{R}}} = \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \sqrt{\alpha}\,\boldsymbol{I}_n \end{bmatrix}$ and $\widetilde{\boldsymbol{y}}^\delta = \begin{bmatrix} \boldsymbol{y}^\delta \\ \boldsymbol{0} \end{bmatrix}$ then the least square solution of the stacked form satisfies the normal equations:

$$\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{\mathcal{R}}}\,\boldsymbol{f} = \widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{y}}^\delta.$$

It is easy to check that $\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{\mathcal{R}}} = \boldsymbol{\mathcal{R}}^\top \boldsymbol{\mathcal{R}} + \alpha\,\boldsymbol{I}_n$ and $\widetilde{\boldsymbol{\mathcal{R}}}^\top \widetilde{\boldsymbol{y}}^\delta = \boldsymbol{\mathcal{R}}^\top \boldsymbol{y}^\delta$, which yields the following result.

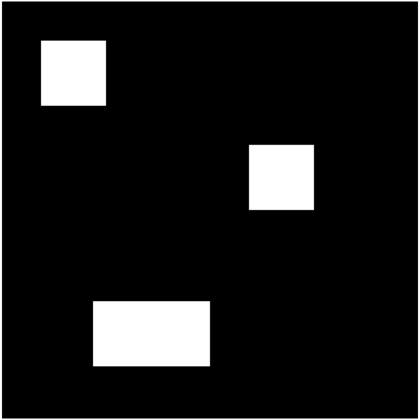
Theorem [Tikhonov regularization]

The one-parameter family of functionals $\{\Gamma_\alpha\}_{\alpha>0}$ defined by

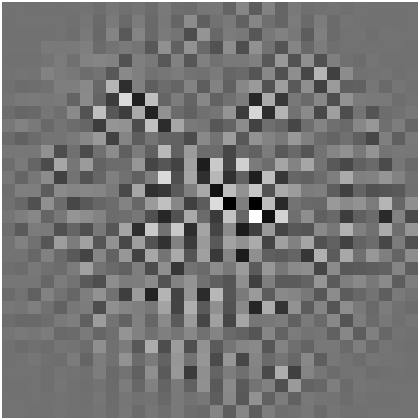
$$\Gamma_\alpha = (\boldsymbol{\mathcal{R}}^\top \boldsymbol{\mathcal{R}} + \alpha\,\boldsymbol{I}_n)^{-1}\boldsymbol{\mathcal{R}}^\top \quad \Leftrightarrow \quad \boldsymbol{f}^{\text{TIKH}} = (\boldsymbol{\mathcal{R}}^\top \boldsymbol{\mathcal{R}} + \alpha\,\boldsymbol{I}_n)^{-1}\boldsymbol{\mathcal{R}}^\top \boldsymbol{y}^\delta$$

is a linear regularization algorithm (i.e., it satisfies conditions (1) and (2)).

Naive Reconstruction (Moore-Penrose Pseudoinverse)

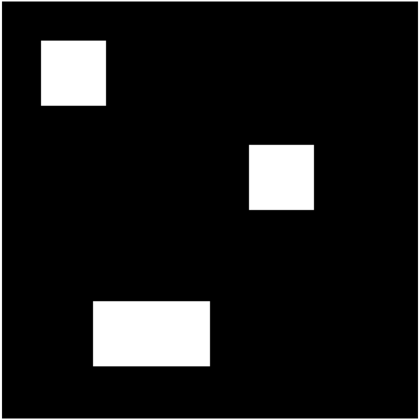


Original phantom

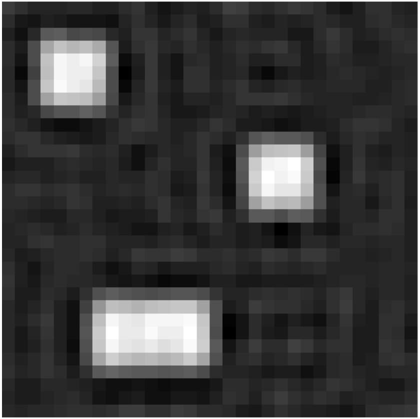


$\boldsymbol{f}^\dagger$: RE = 100%

Truncated SVD Regularization

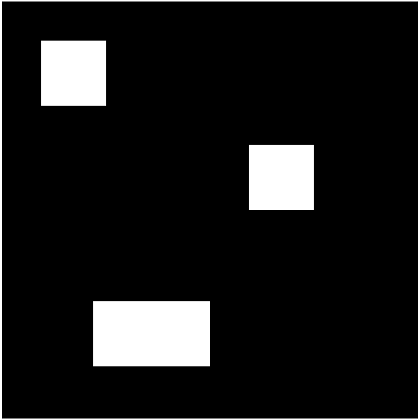


Original phantom

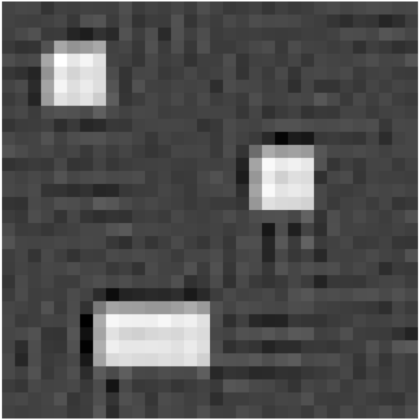


$\boldsymbol{f}^{\mathsf{TSVD}}$: RE $= 35\%$

Tikhonov Regularization

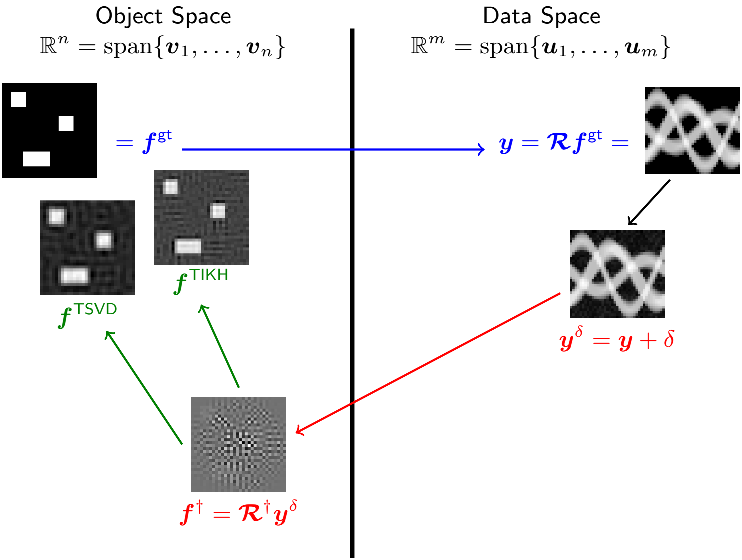


Original phantom



$\boldsymbol{f}^{\mathsf{TIKH}}$: RE = 32%

Where Tikhonov Solution Stands in The Geometry of Ill-Conditioned Problems



Variational Regularization

In general, a minimization problem of the form:

$$\Gamma_\alpha(\boldsymbol{y}^\delta) = \underset{f\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta \right\|_2^2 + \alpha \, \text{Reg}(\boldsymbol{f}) \right\}$$

is called variational formulation.

Main ingredients:

- Data fidelity (or data fitting) term: $\left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta \right\|_2^2$ keeps the estimation of the solution close to the data under the forward physical system
- Regularization parameter: $\alpha > 0$ controls the trade-off between a good fit and the requirements from the regularization
- Regularization term: $\text{Reg}(\boldsymbol{f})$ incorporates **a priori** information or assumptions on the unknown $\boldsymbol{f}$. A **non** exhaustive list:
  - Tikhonov regularization: $\|\boldsymbol{f}\|_2^2$
  - Generalized Tikhonov regularization: $\|\boldsymbol{\mathcal{L}} \boldsymbol{f}\|_2^2$
  - Compress sensing or sparse regularization: $\|\boldsymbol{f}\|_0$ or $\|\boldsymbol{f}\|_1$ or $\|\boldsymbol{\Psi} \boldsymbol{f}\|_1$
  - Indicator functions of constraints sets: $\iota_{\mathbb{R}_+}(\boldsymbol{f})$
  - A combination of the above

Sparse Regularization

Finding the sparsest solution:

$$\operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \left\{ \frac{1}{2}\|\boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta\|_2^2 + \alpha \|\boldsymbol{\Psi} \boldsymbol{f}\|_0 \right\}$$

is known as Compress Sensing (CS). The problem above is NP-hard, since it requires a combinatorial search of exponential size for considering all possible supports.

Under certain conditions on $\boldsymbol{\Psi} \boldsymbol{f}$ (and the forward operator), replacing ℓ^0 norm with ℓ^1 norm yields "similar" results. This relaxation leads to a convex problem:

$$\operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \left\{ \frac{1}{2}\|\boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta\|_2^2 + \alpha \|\boldsymbol{\Psi} \boldsymbol{f}\|_1 \right\}.$$

which is at the basis of optimization-based methods for CS.

About the Convex Relaxation

The formulation

$$\operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \alpha \|\boldsymbol{\Psi} \boldsymbol{f}\|_1 \right\}.$$

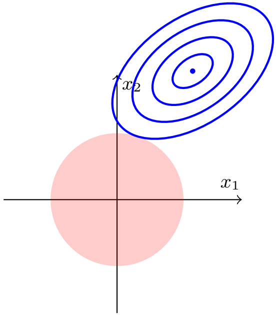
it is more easily solvable, but still nonsmooth. Also, it is convex, but not strictly convex. So why not using Tikhonov regularization?

About the Convex Relaxation

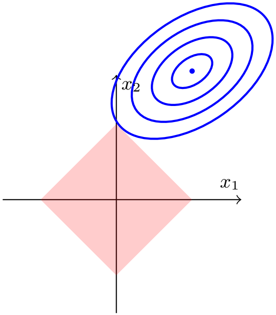
The formulation

$$\operatorname*{argmin}_{\boldsymbol{f}\in\mathbb{R}^n} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^\delta \right\|_2^2 + \alpha \|\boldsymbol{\Psi} \boldsymbol{f}\|_1 \right\}.$$

it is more easily solvable, but still nonsmooth. Also, it is convex, but not strictly convex. So why not using Tikhonov regularization?



$|x_1|^2 + |x_2|^2 =$ const



$|x_1| + |x_2| =$ const

(Constrained) Wavelet-based Regularization

If we take $\boldsymbol{\Psi} = \boldsymbol{\mathcal{W}}$ as the matrix associated with a certain wavelet transform, the variational formulation:

$$\boldsymbol{f}^{\mathsf{WLET}} = \underset{\boldsymbol{f} \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \alpha \left\| \boldsymbol{\mathcal{W}} \boldsymbol{f} \right\|_1 \right\}$$

promotes sparsity on the wavelet coefficients.

(Constrained) Wavelet-based Regularization

If we take $\boldsymbol{\Psi} = \boldsymbol{\mathcal{W}}$ as the matrix associated with a certain wavelet transform, the variational formulation:

$$\boldsymbol{f}^{\mathsf{WLET}} = \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \alpha \left\| \boldsymbol{\mathcal{W}} \boldsymbol{f} \right\|_1 \right\}$$

promotes sparsity on the wavelet coefficients.

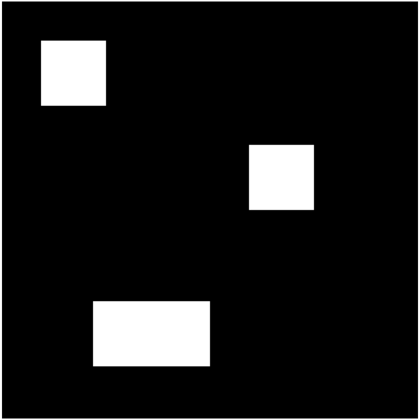
In many cases, it is beneficial to include in the model a nonnegativity constraint:

$$\underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \iota_{\mathbb{R}_{\geq 0}}(\boldsymbol{f}) \right\} \qquad \text{or} \qquad \underset{\boldsymbol{f}\geq 0}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 \right\},$$

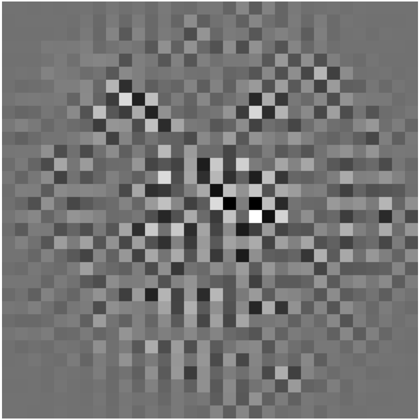
where the inequality is meant component-wise. The nonnegative constraint can also be coupled with other regularisers:

$$\boldsymbol{f}_{+}^{\mathsf{WLET}} = \underset{\boldsymbol{f}\geq 0}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \alpha \left\| \boldsymbol{\mathcal{W}} \boldsymbol{f} \right\|_1 \right\}$$

Naive Reconstruction (Moore-Penrose Pseudoinverse)

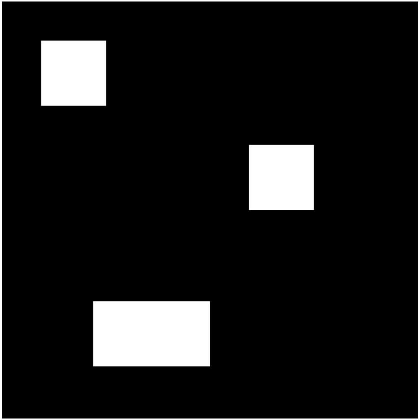


Original phantom

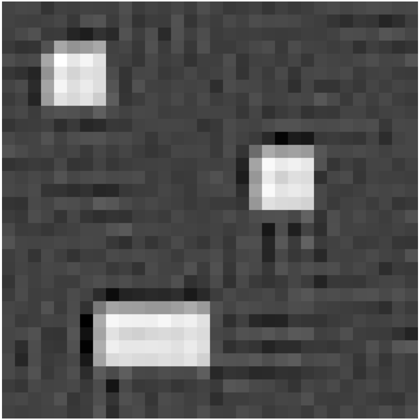


$\boldsymbol{f}^{\dagger}$: RE = 100%

Tikhonov Regularization

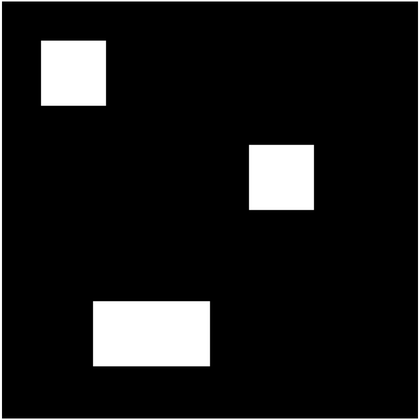


Original phantom

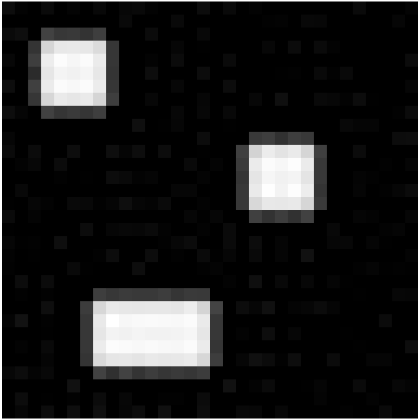


$\boldsymbol{f}^{\mathsf{TIKH}}$: RE = 32%

Nonnegativity Constrained Tikhonov Regularization

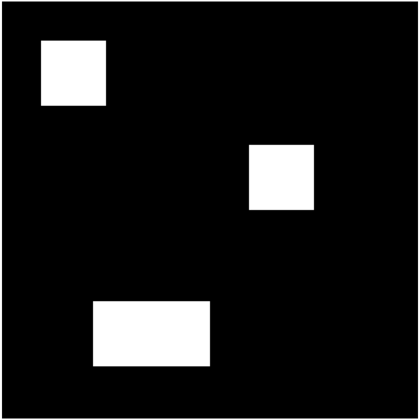


Original phantom

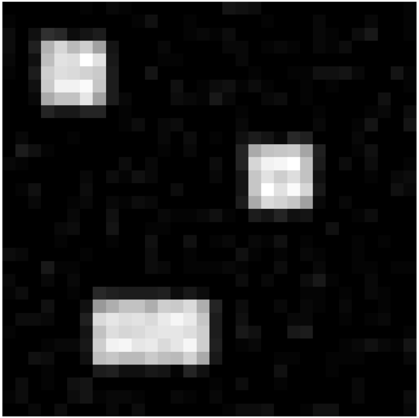


$\boldsymbol{f}_{+}^{\mathsf{TIKH}}$: RE $= 13\%$

Nonnegativity Constrained Wavelet-based Regularization

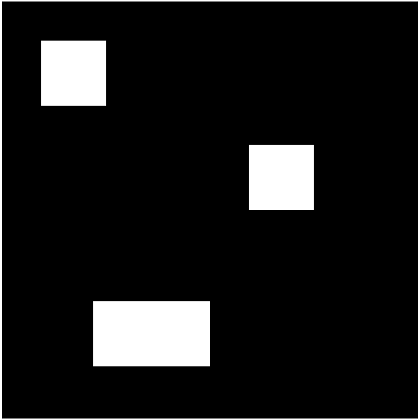


Original phantom

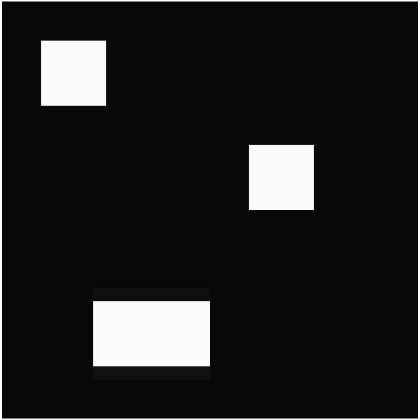


f_+^{WLET}: RE = 26%

Nonnegativity Constrained Total Variation Regularization



Original phantom

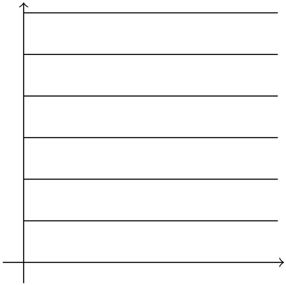


$\boldsymbol{f}_{+}^{\mathrm{TV}}$: RE = 3%

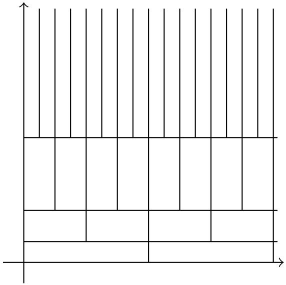
A Bit About Wavelets

Wavelets arose in 1980s to overcome some of the limitations of Fourier analysis. They are a very common choice in CS approaches since they still model images quite adequately.

Similarly to Fourier series, the idea is to “break” a signal into **building blocks**, but unlike Fourier series the building blocks are localized not only in the frequency domain but also in the space domain.



Time-frequency plane for the Fourier Transform.



Time-frequency plane for the wavelet transform.

Building a Wavelet System in 1D

Different families of wavelets can be generated by considering different "parents":

- The scaling function $\phi \in L^2(\mathbb{R}^2)$, a low-pass filter, provides a rougher version of the signal itself
- The (mother) wavelet $\psi \in L^2(\mathbb{R}^2)$, a high-pass filter, describes the details in the signal

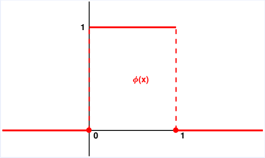
Building a Wavelet System in 1D

Different families of wavelets can be generated by considering different "parents":

- The scaling function $\phi \in L^2(\mathbb{R}^2)$, a low-pass filter, provides a rougher version of the signal itself
- The (mother) wavelet $\psi \in L^2(\mathbb{R}^2)$, a high-pass filter, describes the details in the signal

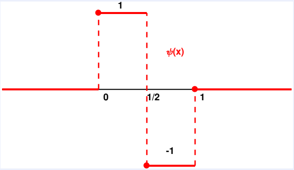
An example

Haar scaling function:



$$\phi(x) = \begin{cases} 1 & 0 < x < 1 \\ 0 & \text{else} \end{cases}$$

Haar wavelet:



$$\psi(x) = \begin{cases} 1 & 0 \leq x < \frac{1}{2} \\ -1 & \frac{1}{2} \leq x < 1 \end{cases}$$

Building a Wavelet System in 1D

A wavelet system is generated by applying to both "parents" two operators:

- Isotropic dilation: $D_M\psi(x) = 2^{-\frac{j}{2}}\psi(2^j x)$, with $j \in \mathbb{Z}$ **scaling** parameter
- Translation: $T_k\psi(x) = \psi(x-k)$, with $k \in \mathbb{Z}$ **location** parameter

Building a Wavelet System in 1D

A wavelet system is generated by applying to both "parents" two operators:

- Isotropic dilation: $D_M \psi(x) = 2^{-\frac{j}{2}} \psi(2^j x)$, with $j \in \mathbb{Z}$ **scaling** parameter
- Translation: $T_k \psi(x) = \psi(x - k)$, with $k \in \mathbb{Z}$ **location** parameter

Then, the elements of a wavelet system are given by:

$$\psi_{jk}(x) = \left\{ T_k D_j \psi(x) = 2^{-\frac{j}{2}} \psi(2^j x - k) \ : \ (j,k) \in \mathbb{Z} \times \mathbb{Z} \right\}$$

and similarly for the scaling function. The wavelets coefficients are the result of the wavelet transform:

$$\mathcal{W} : f \longrightarrow \mathcal{W} f(j,k) = \langle f, \psi_{jk} \rangle$$

and set $V_j = \text{span}\ \{\phi_{jk},\ k \in \mathbb{Z}\}$ and $W_j = \text{span}\ \{\psi_{jk},\ k \in \mathbb{Z}\}$, 1D wavelet analysis relies on the decomposition:

$$f_{j+1} = f_j + w_j, \quad \text{where } f_j = \sum_{k \in \mathbb{Z}} \langle f, \phi_{jk} \rangle \phi_{jk} \text{ and } w_j = \sum_{k \in \mathbb{Z}} \langle f, \psi_{jk} \rangle \psi_{jk}.$$

Building a Wavelet System in 1D

A wavelet system is generated by applying to both "parents" two operators:

- Isotropic dilation: $D_M\psi(x) = 2^{-\frac{j}{2}}\psi(2^j x)$, with $j \in \mathbb{Z}$ **scaling** parameter
- Translation: $T_k\psi(x) = \psi(x-k)$, with $k \in \mathbb{Z}$ **location** parameter

Then, the elements of a wavelet system are given by:

$$\psi_{jk}(x) = \left\{T_k D_j \psi(x) = 2^{-\frac{j}{2}}\psi(2^j x - k) \;:\; (j,k) \in \mathbb{Z}\times\mathbb{Z}\right\}$$

and similarly for the scaling function. The wavelets coefficients are the result of the wavelet transform:

$$\mathcal{W} : f \;\longrightarrow\; \mathcal{W}f(j,k) = \langle f, \psi_{jk}\rangle$$

and set $V_j = \mathrm{span}\,\{\phi_{jk},\; k \in \mathbb{Z}\}$ and $W_j = \mathrm{span}\,\{\psi_{jk},\; k \in \mathbb{Z}\}$, 1D wavelet analysis relies on the decomposition:

$$f_{j+1} = f_j + w_j, \quad \text{where } f_j = \sum_{k\in\mathbb{Z}} \langle f, \phi_{jk}\rangle \phi_{jk} \text{ and } w_j = \sum_{k\in\mathbb{Z}} \langle f, \psi_{jk}\rangle \psi_{jk}.$$

In practical applications these are computed using the language of filters, with convolutions and downsampling and upsampling operations.

2D Wavelets

By considering tensor products, from the scaling and wavelet functions we get 1 scaling function but 3 wavelet functions (horizontal, vertical & diagonal):

$$\Phi(\boldsymbol{x}) = \phi(x_1)\phi(x_2),$$

and

$$\Psi^1(\boldsymbol{x}) = \phi(x_1)\psi(x_2), \quad \Psi^2(\boldsymbol{x}) = \psi(x_1)\phi(x_2), \quad \Psi^3(\boldsymbol{x}) = \psi(x_1)\psi(x_2).$$

Similarly to the 1D case, one defines the approximation space $\mathbf{V}_j$ and the wavelet spaces $\mathbf{W}_j^1$, $\mathbf{W}_j^2$, and $\mathbf{W}_j^3$ as the following linear span in $L^2(\mathbb{R}^2)$:

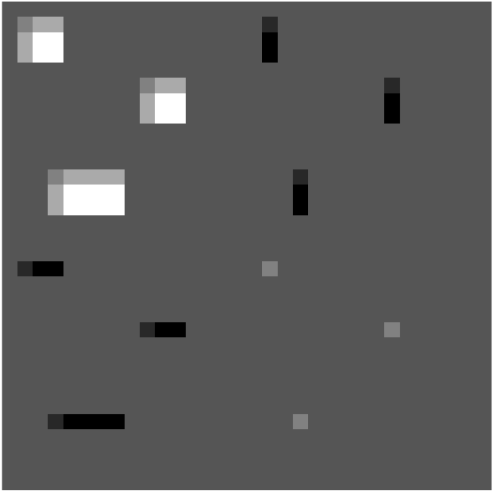
$$\mathbf{V}_j = \text{span}\,\{\Phi_{j\boldsymbol{k}},\ \boldsymbol{k} \in \mathbb{Z}^2\}, \qquad \mathbf{W}_j^\lambda = \text{span}\,\{\Psi_{j\boldsymbol{k}}^\lambda,\ \boldsymbol{k} \in \mathbb{Z}^2\}$$

where $\lambda = 1, 2, 3$. Therefore, 2D wavelet analysis relies on the decomposition $\mathbf{V}_{j+1} = \mathbf{V}_j \oplus \mathbf{W}_j^1 \oplus \mathbf{W}_j^2 \oplus \mathbf{W}_j^3$, which yields:

$$f_{j+1} = f_j + \sum_{\lambda=1}^{3} w_j^\lambda, \quad \text{where } f_j = \sum_{\boldsymbol{k}\in\mathbb{Z}^2} \langle f, \Phi_{j\boldsymbol{k}}\rangle \Phi_{j\boldsymbol{k}} \text{ and } w_j^\lambda = \sum_{\boldsymbol{k}\in\mathbb{Z}^2} \langle f, \Psi_{j\boldsymbol{k}}^\lambda\rangle \Psi_{j\boldsymbol{k}}^\lambda.$$

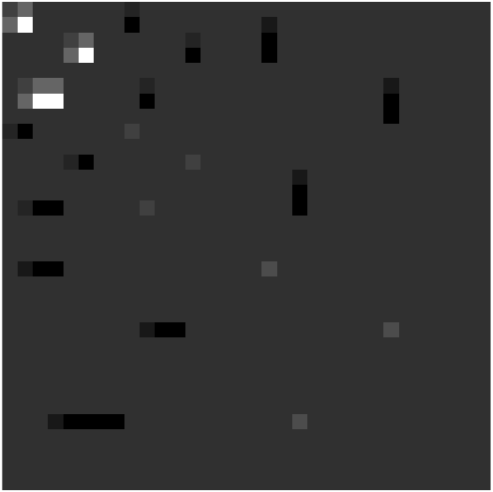
Then, one iterates on f_j as in the 1D case. Filter-wise, this amounts to applying the 1D filters first on the rows and then on the columns of the 2D signal.

An Example: Haar Wavelet Transform of the Square Phantom



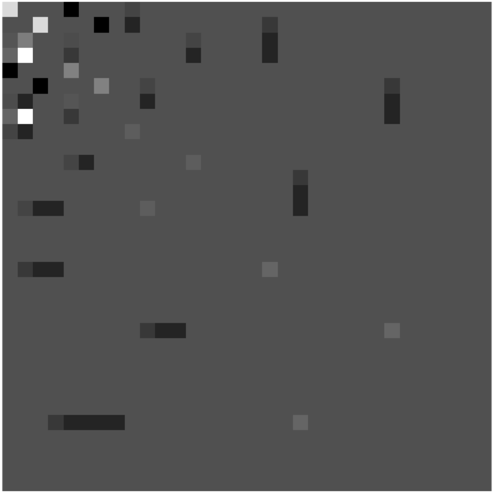
1-level Haar wavelet transform

An Example: Haar Wavelet Transform of the Square Phantom



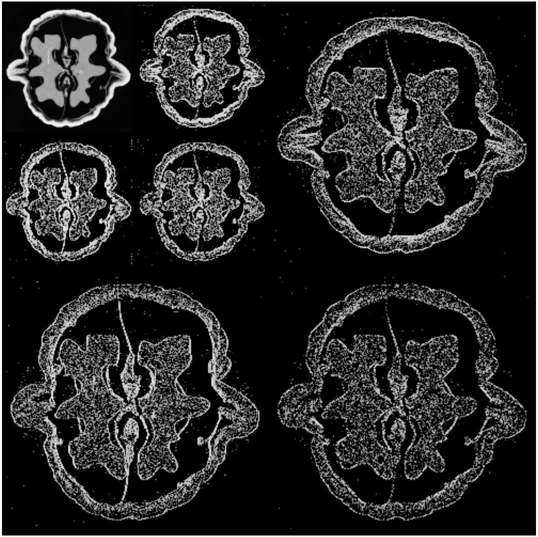
2-level Haar wavelet transform

An Example: Haar Wavelet Transform of the Square Phantom



3-level Haar wavelet transform

An Example: Haar Wavelet Transform of a Walnut



Sparse Image Representation With Wavelets

Our goal: Use a wavelet representation of signals to extract their key features efficiently - a few coefficients carry most of the information. How to quantify it?

Sparse Image Representation With Wavelets

Our goal: Use a wavelet representation of signals to extract their key features efficiently - a few coefficients carry most of the information. How to quantify it?

Approximation error - linear and nonlinear

Consider a sequence of approximations U_N of a space U (either $L^2[0,1]$ or $L^2[0,1]^2$), and an o.n.b. $\{\varphi_i\}_{i\in\mathbb{Z}}$ of U such that $\{\varphi_i\}_{i=1}^N$ is an o.n.b. for U_N (e.g., wavelet multiresolutions from max scale J)

- Linear approximation: $f_N^{\mathsf{l}} = \sum_{i=1}^N \langle f, \varphi_i\rangle\varphi_i \rightsquigarrow \mathsf{e_l}(N,f) = \|f - f_N^{\mathsf{l}}\|^2$
- Nonlinear approximation: $f_N^{\mathsf{n}} = \sum_{i\in\Lambda_N} \langle f, \varphi_i\rangle\varphi_i \rightsquigarrow \mathsf{e_n}(N,f) = \|f - f_N^{\mathsf{n}}\|^2$ where Λ_N are the indices of the N biggest coefficients $|\langle f, \varphi_i\rangle|$, $i\in\mathbb{Z}$

Sparse Image Representation With Wavelets

Our goal: Use a wavelet representation of signals to extract their key features efficiently - a few coefficients carry most of the information. How to quantify it?

Approximation error - linear and nonlinear

Consider a sequence of approximations U_N of a space U (either $L^2[0,1]$ or $L^2[0,1]^2$), and an o.n.b. $\{\varphi_i\}_{i\in\mathbb{Z}}$ of U such that $\{\varphi_i\}_{i=1}^N$ is an o.n.b. for U_N (e.g., wavelet multiresolutions from max scale J)

- Linear approximation: $f_N^{\mathsf{l}} = \sum_{i=1}^N \langle f, \varphi_i\rangle \varphi_i \rightsquigarrow \mathsf{e}_{\mathsf{l}}(N,f) = \|f - f_N^{\mathsf{l}}\|^2$
- Nonlinear approximation: $f_N^{\mathsf{n}} = \sum_{i\in\Lambda_N} \langle f, \varphi_i\rangle \varphi_i \rightsquigarrow \mathsf{e}_{\mathsf{n}}(N,f) = \|f - f_N^{\mathsf{n}}\|^2$ where Λ_N are the indices of the N biggest coefficients $|\langle f, \varphi_i\rangle|$, $i \in \mathbb{Z}$

In 1D, wavelets provide very good approximation results, especially for regular functions. Let $\{\varphi_i\}$ be generated by a wavelet with q vanishing moments:

- If $f \in \mathrm{Lip}^\alpha([0,1])$, with $1/2 < \alpha < q$, then $\mathsf{e}_{\mathsf{l}}(N,f) = O(N^{-2\alpha})$
- If f has K discontinuities in $[0,1]$ and is $\mathrm{Lip}^\alpha([0,1])$ between them, with $1/2 < \alpha < q$, then:

$$\mathsf{e}_{\mathsf{l}}(N,f) = O(KN^{-1}), \qquad \mathsf{e}_{\mathsf{n}}(N,f) = O(N^{-2\alpha})$$

Cartoon-like Images and Limitations of Wavelet Representation

What about in 2D? Wavelets are still good at approximating regular images:

$$f \in \mathsf{Lip}^{\alpha}([0,1]^2),\ 1/2 < \alpha < q \quad \Rightarrow \quad \mathsf{e_l}(N,f) = O(N^{-\alpha})$$

Cartoon-like Images and Limitations of Wavelet Representation

What about in 2D? Wavelets are still good at approximating regular images:

$$f \in \mathsf{Lip}^{\alpha}([0,1]^2),\ 1/2 < \alpha < q \quad \Rightarrow \quad \mathsf{e_l}(N,f) = O(N^{-\alpha})$$

A natural concept of piecewise regular functions in 2D are cartoon-like images: piecewise $\mathcal{C}^2$ regular, with jumps along $\mathcal{C}^2$-smooth curves.

$$f \ \ \text{cartoon-like} \quad \Rightarrow \quad \mathsf{e_n}(N,f) = O(N^{-1}). \qquad \text{Why?}$$

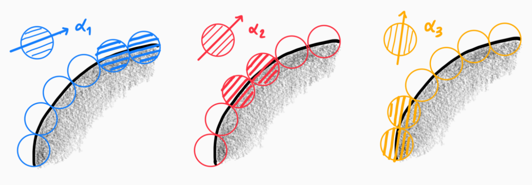
Cartoon-like Images and Limitations of Wavelet Representation

What about in 2D? Wavelets are still good at approximating regular images:

$$f \in \mathrm{Lip}^{\alpha}([0,1]^2),\ 1/2 < \alpha < q \quad \Rightarrow \quad \mathrm{e_l}(N,f) = O(N^{-\alpha})$$

A natural concept of piecewise regular functions in 2D are cartoon-like images: piecewise $\mathcal{C}^2$ regular, with jumps along $\mathcal{C}^2$-smooth curves.

$$f \ \ \text{cartoon-like} \quad \Rightarrow \quad \mathrm{e_n}(N,f) = O(N^{-1}). \qquad \text{Why?}$$



At scale 2^j, a discontinuity along an edge is captured by $\sim 2^{-j}$ coefficients: we could sample less frequently in the relevant direction.

$\rightsquigarrow$ Problem: the support of wavelets is isotropic and does not depend on the direction itself!

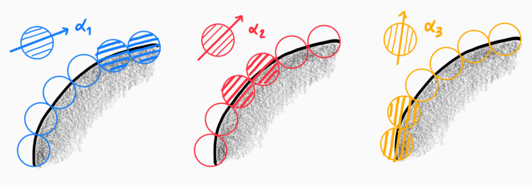
Cartoon-like Images and Limitations of Wavelet Representation

What about in 2D? Wavelets are still good at approximating regular images:

$$f \in \mathsf{Lip}^{\alpha}([0,1]^2),\ 1/2 < \alpha < q \quad \Rightarrow \quad \mathsf{e_l}(N,f) = O(N^{-\alpha})$$

A natural concept of piecewise regular functions in 2D are cartoon-like images: piecewise $\mathcal{C}^2$ regular, with jumps along $\mathcal{C}^2$-smooth curves.

$$f \ \text{ cartoon-like} \quad \Rightarrow \quad \mathsf{e_n}(N,f) = O(N^{-1}). \qquad \text{Why?}$$



At scale 2^j, a discontinuity along an edge is captured by $\sim 2^{-j}$ coefficients: we could sample less frequently in the relevant direction.

$\rightsquigarrow$ Problem: the support of wavelets is isotropic and does not depend on the direction itself!

Can we do better? Not with wavelets! Optimal rate: $O(N^{-2})$, by piecewise linear interpolation on adaptive grids. This is for another day!

Wavelet-based Regularization: Putting Everything Together

Let's circle back to the (contrained) wavelet-based regularization:

$$\boldsymbol{f}^{\mathsf{WLET}} = \underset{\boldsymbol{f} \geq 0}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{R}} \boldsymbol{f} - \boldsymbol{y}^{\delta} \right\|_2^2 + \alpha \left\| \boldsymbol{\mathcal{W}} \boldsymbol{f} \right\|_1 \right\}$$

Despite not providing optimal multiscale approximations, wavelets are a very common choice in imaging tasks since they still model images quite adequately and are computationally very efficient.

Indeed, wavelet **coefficients** come with **different magnitudes** so that:

- The smallest coefficients are associated with noise
- The largest ones are associated with edges and images dominant features

The ℓ^1 norm suppresses the small coefficients in favor of the largest ones, yielding a powerful regularization strategy.

How to Solve the Minimization Problem? Convex Optimization!

Let's consider minimization problems of the form:

$$\boldsymbol{f}^* \in \operatorname*{argmin}_{\boldsymbol{f} \in \mathbb{R}^n} J(\boldsymbol{f}), \qquad \text{with } \; J(\boldsymbol{f}) := D(\boldsymbol{f}) + \varphi(\boldsymbol{f}), \tag{1}$$

where, denoted $\overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$, we have:

- $D \colon \mathbb{R}^n \to \overline{\mathbb{R}}$ prescribed metric of the fidelity of $\boldsymbol{\mathcal{A}f}$ w.r.t. measurement $\boldsymbol{y}$. For example: $D(\boldsymbol{f}) = \frac{1}{2}\|\boldsymbol{\mathcal{A}f} - \boldsymbol{y}\|_2^2$
- $\varphi(\boldsymbol{f}) : \mathbb{R}^n \to \overline{\mathbb{R}}$ regularization term

How to Solve the Minimization Problem? Convex Optimization!

Let's consider minimization problems of the form:

$$\boldsymbol{f}^* \in \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}}\, J(\boldsymbol{f}), \qquad \text{with} \;\; J(\boldsymbol{f}) := D(\boldsymbol{f}) + \varphi(\boldsymbol{f}), \tag{1}$$

where, denoted $\overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$, we have:

- $D\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ prescribed metric of the fidelity of $\boldsymbol{\mathcal{A}}\boldsymbol{f}$ w.r.t. measurement $\boldsymbol{y}$. For example: $D(\boldsymbol{f}) = \frac{1}{2}\|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}\|_2^2$
- $\varphi(\boldsymbol{f}) : \mathbb{R}^n \to \overline{\mathbb{R}}$ regularization term

We require D and φ to be convex, lower semi-continuous, proper, and coercive.

A function $\phi : \mathbb{R}^n \to \overline{\mathbb{R}}$ is:

- convex if $\phi(\alpha\boldsymbol{f}_1 + (1-\alpha)\boldsymbol{f}_2) \leq \alpha\phi(\boldsymbol{f}_1) + (1-\alpha)\phi(\boldsymbol{f}_2)$ for all $\alpha \in [0,1]$
- lower semi-continuous (l.s.c.) at a point $\boldsymbol{f}_0$ if for every sequence $\boldsymbol{f}^{(k)} \to \boldsymbol{f}_0$ in $\mathbb{R}^n$, $\liminf_{n\to\infty} \phi(\boldsymbol{f}^{(k)}) \geq \phi(\boldsymbol{f}_0)$
- proper if the effective domain $\operatorname{dom}\phi := \{\boldsymbol{f} \in \mathbb{R}^n \mid \phi(\boldsymbol{f}) < +\infty\} \neq \emptyset$
- coercive if for all $\boldsymbol{f}^{(k)}$ with $\|\boldsymbol{f}^{(k)}\| \to \infty$, we have that $\phi(\boldsymbol{f}^{(k)}) \to \infty$

$\Gamma_0(\mathbb{R}^n)$ denotes the class of convex, proper, and l.s.c. functions from $\mathbb{R}^n \to \overline{\mathbb{R}}$.

Existance of Minimizers

It is easy to show that, if $D, \varphi \in \Gamma_0(\mathbb{R}^n)$ are coercive, then the minimization problem (1) admits a solution.

Existance of Minimizers

It is easy to show that, if $D, \varphi \in \Gamma_0(\mathbb{R}^n)$ are coercive, then the minimization problem (1) admits a solution.

An alternative requirement on J is requiring L-smoothness, i.e., J has L-Lipschitz gradient:

$$\exists\ L > 0 \ : \quad \|\nabla J(\boldsymbol{f}_1) - \nabla J(\boldsymbol{f}_2)\|_2 \leq L\|\boldsymbol{f}_1 - \boldsymbol{f}_2\|_2$$

which, under convexity, is equivalent to:

$$J(\boldsymbol{f}_1) \leq J(\boldsymbol{f}_2) + \langle \nabla J(\boldsymbol{f}_2), \boldsymbol{f}_2 - \boldsymbol{f}_1 \rangle + \frac{L}{2}\|\boldsymbol{f}_1 - \boldsymbol{f}_2\|_2^2$$

Theorem

Let J be proper, convex, L-smooth and coercive. Then, J admits a minimiser. All local minimisers are global minimisers.

Variational Problems with Smooth Regularizers

The Gradient Descent (GD) algorithm is a classic choice for the solution of L-smooth minimization problems (i.e., for (L-)smooth J).

Algorithm Gradient Descent algorithm

Require: $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$ (initial guess), $\tau \in (0, 2/L)$, ϱ (tolerance)

```
for k = 1, ..., K do
    f^(k+1) = f^(k) − τ∇J(f^(k))
    if ‖f^(k+1) − f^(k)‖ < ϱ then
        Stop and return f^(k+1).
    end if
end for
```

Variational Problems with Smooth Regularizers

The Gradient Descent (GD) algorithm is a classic choice for the solution of L-smooth minimization problems (i.e., for (L-)smooth J).

Algorithm Gradient Descent algorithm

Require: $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$ (initial guess), $\tau \in (0, 2/L)$, ϱ (tolerance)

```
for k = 1, ..., K do
    f^(k+1) = f^(k) − τ∇J(f^(k))
    if ‖f^(k+1) − f^(k)‖ < ϱ then
        Stop and return f^(k+1).
    end if
end for
```

Theorem (convergence of GD)

Let J be L-smooth, $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$ and $\tau \in (0, 2/L)$. There holds $\boldsymbol{f}^{(k)} \to \boldsymbol{f}^*$, where $(\boldsymbol{f}^{(k)})_{k \in \mathbb{N}}$ is the sequence generated by GD, and for the function values:

$$J(\boldsymbol{f}^{(k)}) - J(\boldsymbol{f}^*) \leq \frac{\|\boldsymbol{f}^{(k)} - \boldsymbol{f}^*\|_2^2}{2\tau k}$$

Variational Problems with Smooth Regularizers

The Gradient Descent (GD) algorithm is a classic choice for the solution of L-smooth minimization problems (i.e., for (L-)smooth J).

Algorithm Gradient Descent algorithm

Require: $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$ (initial guess), $\tau \in (0, 2/L)$, ϱ (tolerance)

for $k = 1, \ldots, K$ **do**
  $\boldsymbol{f}^{(k+1)} = \boldsymbol{f}^{(k)} - \tau \nabla J(\boldsymbol{f}^{(k)})$
  if $\|\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)}\| < \varrho$ **then**
    Stop and return $\boldsymbol{f}^{(k+1)}$.
  end if
end for

Example: Tikhonov regularization. The problem is quadratic:

$$J^{\mathsf{TIKH}}(\boldsymbol{f}) := \frac{1}{2}\|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}\|_2^2 + \alpha\|\boldsymbol{\mathcal{L}}\boldsymbol{f}\|_2^2 \qquad \text{with} \quad \boldsymbol{\mathcal{L}} = \{\boldsymbol{I}_n, \nabla\}$$

and the GD iteration reads as:

$$\boldsymbol{f}^{(k+1)} = \boldsymbol{f}^{(k)} - \tau \left(\boldsymbol{\mathcal{A}}^\top(\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}) + \alpha \boldsymbol{\mathcal{L}}^\top \boldsymbol{\mathcal{L}}\boldsymbol{f} \right) \quad \text{for} \quad k = 0, 1, \ldots$$

Variational Problems with Non-smooth Regularizers

Let's now assume the regularizer φ to be **non-smooth**.

Definition (proximal operator)

For a convex function $\phi : \mathbb{R}^n \to \overline{\mathbb{R}}$ we have

$$\mathbf{prox}_{\phi}(\boldsymbol{f}) = \operatorname*{argmin}_{\boldsymbol{v}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{v} - \boldsymbol{f}\|_2^2 + \phi(\boldsymbol{v})$$

For $\phi \in \Gamma_0(\mathbb{R}^n)$, $\mathbf{prox}_{\phi}(\cdot)$ is well-defined, single-valued and **non-expansive**.

Variational Problems with Non-smooth Regularizers

Let's now assume the regularizer φ to be **non-smooth**.

Definition (proximal operator)

For a convex function $\phi : \mathbb{R}^n \to \overline{\mathbb{R}}$ we have

$$\mathbf{prox}_{\phi}(\boldsymbol{f}) = \operatorname*{argmin}_{\boldsymbol{v} \in \mathbb{R}^n} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{f}\|_2^2 + \phi(\boldsymbol{v})$$

For $\phi \in \Gamma_0(\mathbb{R}^n)$, $\mathbf{prox}_{\phi}(\cdot)$ is well-defined, single-valued and **non-expansive**.

Example: Important proximity operators

- Consider the indicator ι_C of a convex set C (e.g., $C = \mathbb{R}^n_+$):

$$\mathbf{prox}_{\iota_C}(\boldsymbol{f}) = \operatorname*{argmin}_{\boldsymbol{v} \in \mathbb{R}^n} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{f}\|_2^2 + \iota_C(\boldsymbol{v}) = \operatorname*{argmin}_{\boldsymbol{v} \in C} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{f}\|_2^2 = \mathrm{proj}_C(\boldsymbol{f})$$

This is the Euclidean projection on S!

Variational Problems with Non-smooth Regularizers

Let's now assume the regularizer φ to be **non-smooth**.

Definition (proximal operator)

For a convex function $\phi : \mathbb{R}^n \to \overline{\mathbb{R}}$ we have

$$\mathbf{prox}_{\phi}(\boldsymbol{f}) = \operatorname*{argmin}_{\boldsymbol{v} \in \mathbb{R}^n} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{f}\|_2^2 + \phi(\boldsymbol{v})$$

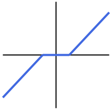
For $\phi \in \Gamma_0(\mathbb{R}^n)$, $\mathbf{prox}_{\phi}(\cdot)$ is well-defined, single-valued and **non-expansive**.

Example: Important proximity operators

- Soft-thresholding is the proximity operator of the ℓ^1 norm:

  $$S_\gamma(\boldsymbol{f}) = (\boldsymbol{f} - \gamma)\chi_{[\gamma,\infty)} + (\boldsymbol{f} + \gamma)\chi_{(-\infty,-\gamma]}$$

  [this is meant to be applied component-wise]



- Many more proximity operators: https://proximity-operator.net

Proximal Gradient Descent

A generalization of GD is the Proximal Gradient Descent (PGD) method, which alternates between a proximal step on φ and a gradient descent step on D.

Algorithm Proximal Gradient Descent algorithm

Require: $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$ (initial guess), $\gamma \in (0, 1/L_D)$, ϱ (tolerance)

for $k = 1, \ldots, K$ **do**
  $\boldsymbol{f}^{(k+1)} = \mathbf{prox}_{\gamma\varphi}(\boldsymbol{f}^{(k)} - \gamma \nabla J(\boldsymbol{f}^{(k)}))$
  if $\|\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)}\| < \varrho$ **then**
    Stop and return $\boldsymbol{f}^{(k+1)}$.
  end if
end for

Proximal Gradient Descent – Convergence

Theorem (e.g., [Beck & Teboulle 2009])

Let $\boldsymbol{f}^* \in \operatorname{argmin}_{\boldsymbol{f} \in \mathbb{R}^n} J(\boldsymbol{f})$ and $\boldsymbol{f}^{(0)} \in \mathbb{R}^n$ the starting point. Assumptions:

- $D \in \Gamma_0(\mathbb{R}^n)$ differentiable with L_D-Lipschitz gradient
- $\varphi \in \Gamma_0(\mathbb{R}^n)$, possibly non-smooth
- $\gamma \in (0, 1/L_D)$

Then, the iterates generated by PGD verify the following properties:

1. $(J(\boldsymbol{f}^{(k)}))_{k \in \mathbb{N}}$ is non-increasing
2. The sequence $(J(\boldsymbol{f}^{(k)}) - J(\boldsymbol{f}^*))_{k \in \mathbb{N}}$ goes to zero at a rate $\mathcal{O}(1/k)$
3. The sequence $\boldsymbol{f}^{(k)}$ converges to a minimizer $\boldsymbol{f}^*$ of the problem

A Special Instance of PGD: Wavelet-based Regularization & ISTA

Let $\boldsymbol{\mathcal{W}}$ be an orthogonal wavelet transform. Recall wavelet-based regularization:

$$\boldsymbol{f}^{\mathsf{WLET}} = \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2}\,\|\boldsymbol{\mathcal{A}}\boldsymbol{f}-\boldsymbol{y}\|_2^2 + \alpha\|\boldsymbol{\mathcal{W}}\boldsymbol{f}\|_1 \right\}$$

$\boldsymbol{f}^{\mathsf{WLET}}$ is a special case for which the proximal operator (associated with 1-norm regularization) is easy and fast to compute, because it is given by the soft-thresholding operator: $S_\gamma(\boldsymbol{w}) = (\boldsymbol{w}-\gamma)\chi_{[\gamma,\infty)} + (\boldsymbol{w}+\gamma)\chi_{(-\infty,-\gamma]}$.

A Special Instance of PGD: Wavelet-based Regularization & ISTA

Let $\mathcal{W}$ be an orthogonal wavelet transform. Recall wavelet-based regularization:

$$\boldsymbol{f}^{\mathsf{WLET}} = \underset{\boldsymbol{f}\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2}\|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}\|_2^2 + \alpha \|\boldsymbol{\mathcal{W}}\boldsymbol{f}\|_1 \right\}$$

$\boldsymbol{f}^{\mathsf{WLET}}$ is a special case for which the proximal operator (associated with 1-norm regularization) is easy and fast to compute, because it is given by the soft-thresholding operator: $S_\gamma(\boldsymbol{w}) = (\boldsymbol{w} - \gamma)\chi_{[\gamma,\infty)} + (\boldsymbol{w} + \gamma)\chi_{(-\infty,-\gamma]}$.

Algorithm Iterative Soft-Thresholding Algorithm

Require: $\boldsymbol{f}^{(0)} \in \mathrm{dom}(J)$, $\boldsymbol{\mathcal{W}}$ orthogonal, τ (step-size), ϱ (tolerance)

```
for k = 1, ..., K do
    f^(k+1) = W^T S_{α/τ}( W( f^(k) − (1/τ) A^T A f^(k) + (1/τ) A^T y ) )
    if ||f^(k+1) − f^(k)|| < ϱ then
        Stop and return f^(k+1).
    end if
end for
```

for $k = 1, \ldots, K$ **do**

$\quad \boldsymbol{f}^{(k+1)} = \boldsymbol{\mathcal{W}}^\top \mathcal{S}_{\alpha/\tau}\left(\boldsymbol{\mathcal{W}}\left(\boldsymbol{f}^{(k)} - \frac{1}{\tau}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{A}}\boldsymbol{f}^{(k)} + \frac{1}{\tau}\boldsymbol{\mathcal{A}}^\top\boldsymbol{y}\right)\right)$

$\quad$ **if** $\|\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)}\| < \varrho$ **then**

$\quad\quad$ Stop and return $\boldsymbol{f}^{(k+1)}$.

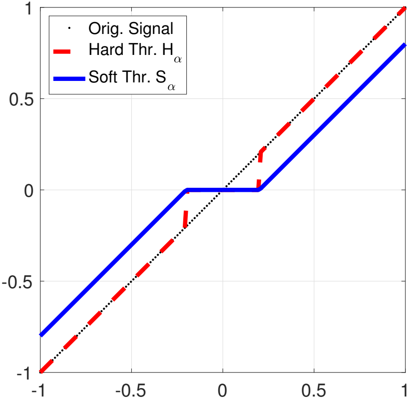
$\quad$ **end if**

end for

Hard- and Soft-Thresholding

$$S_\alpha(x) = \begin{cases} x+\alpha & \text{if } \ x < -\alpha \\ 0 & \text{if } \ |x| \le \alpha \\ x-\alpha & \text{if } \ x > \alpha \end{cases}$$

$$H_\alpha(x) = \begin{cases} x & \text{if } \ x < -\alpha \\ 0 & \text{if } \ |x| \le \alpha \\ x & \text{if } \ x > \alpha \end{cases}$$



Connection: Hard-thresholding arises when considering the minimisation problem

$$\underset{\boldsymbol{w}\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\boldsymbol{w}-\boldsymbol{v}\|_2^2 + \alpha \|\boldsymbol{w}\|_0 \right\}$$

where the ℓ^0 semi-norm counts the number of non-zeros components in $\boldsymbol{w}$.

Constrained PGD: Primal-Dual Fixed Point

Recall the nonnegativity constrained wavelet-based regularization:

$$\boldsymbol{f}_{+}^{\mathsf{WLET}} = \operatorname*{argmin}_{\boldsymbol{f}\geq 0} \left\{ \frac{1}{2} \|\boldsymbol{\mathcal{A}}\boldsymbol{f} - \boldsymbol{y}\|_2^2 + \alpha \|\boldsymbol{\mathcal{W}}\boldsymbol{f}\|_1 \right\}$$

There are many variants of ISTA (or PGD) to extend it to non-orthogonal bases (or frames), or to include the nonnegativity constraint.

Constrained PGD: Primal-Dual Fixed Point

Recall the nonnegativity constrained wavelet-based regularization:

$$\boldsymbol{f}_+^{\mathsf{WLET}} = \underset{\boldsymbol{f}\geq 0}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\boldsymbol{\mathcal{A}} \boldsymbol{f} - \boldsymbol{y}\|_2^2 + \alpha \|\boldsymbol{\mathcal{W}} \boldsymbol{f}\|_1 \right\}$$

There are many variants of ISTA (or PGD) to extend it to non-orthogonal bases (or frames), or to include the nonnegativity constraint.

Primal-Dual Fixed Point (PDFP) algorithm

Select $\boldsymbol{f}^{(0)}$ and choose $\lambda < 1/\lambda_{\max}(\|\boldsymbol{\mathcal{W}}^\top \boldsymbol{\mathcal{W}}\|)$ and $0 < \gamma < 2/L_D$. Then, the PDFP update is:

$$\begin{cases} \boldsymbol{d}^{(k+1)} &= \operatorname{proj}_{\mathbb{R}_+^n} \left(\boldsymbol{f}^{(k)} - \gamma(\boldsymbol{\mathcal{A}}^\top \widetilde{\boldsymbol{\mathcal{A}}} \boldsymbol{f}^{(k)} - \boldsymbol{\mathcal{A}}^\top \boldsymbol{y}) - \lambda \boldsymbol{\mathcal{W}}^\top \boldsymbol{v}^{(i)} \right), \\ \boldsymbol{v}^{(k+1)} &= \left(\boldsymbol{I} - S_{\alpha \frac{\gamma}{\lambda}} \right) \left(\boldsymbol{\mathcal{W}} \boldsymbol{d}^{(k+1)} + \boldsymbol{v}^{(k)} \right), \\ \boldsymbol{f}^{(k+1)} &= \operatorname{proj}_{\mathbb{R}_+^n} \left(\boldsymbol{f}^{(k)} - \gamma(\boldsymbol{\mathcal{A}}^\top \widetilde{\boldsymbol{\mathcal{A}}} \boldsymbol{f}^{(k)} - \boldsymbol{\mathcal{A}}^\top \boldsymbol{y}) - \lambda \boldsymbol{\mathcal{W}}^\top \boldsymbol{v}^{(k+1)} \right) \end{cases}$$

PDFP can be applied also when in place of $\boldsymbol{\mathcal{W}}$ we use a linear $\boldsymbol{\mathcal{L}}$ associated with a non-orthogonal basis or a frame.

Summary & Outlook

What we learned today:

- Discretization and ill-conditioning of the tomographic problem
- TSVD and variational regularization
- A very small wavelet tour of signal processing
- Gradient descent (GD) and its non-smooth variant (PGD, ISTA, PDFP)

What I do not have time to talk about:

- Continuous regularization theory of inverse problems (convergence, rate of convergence, . . .)
- Choice of the regularization parameter (a priori, a posteriori rules)
- There is a whole zoo of first-order optimization methods: we barely scratched the surface!

Up next:

- Nods to machine and deep learning
- Learned reconstruction methods for inverse problems

Some References

P.C. Hansen, J.S. Jørgensen and W.R.B. Lionheart (Eds.)
Computed Tomography: Algorithms, Insight, and Just Enough Theory
Society for Industrial and Applied Mathematics, 2021

A. Boggess and F.J. Narcowich
A first course in wavelets with Fourier analysis
John Wiley & Sons, 2009

Regularization of inverse problems:

- Bertero & Boccacci, *Introduction to inverse problems in imaging*, 1998
- Engl, Hanke & Neubauer, *Regularization of inverse problems*, 1996
- Hansen, *Discrete inverse problems: insight and algorithms*, 2010
- Mueller & Siltanen, *Linear and nonlinear inverse problems with practical applications*, 2012
- Scherzer, Grasmair, Grossauer, Haltmeier & Lenzen, *Variational methods in imaging*, 2009

Wavelets:

- Mallat, *A wavelet tour of signal processing*, 1999.

Some References

C. Dossal, S. Hurault and N. Papadakis
Optimization with First Order Algorithms, preprint, 2024

Books:

- Boyd & Vandenberghe, Convex optimization, 2004
- Nocedal & Wright, Numerical optimization, 2006
- Rockafellar, Convex optimization, 1996

Papers:

- Beck & Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, 2009
- Chambolle & Dossal, *On the convergence of the iterates of the "fast iterative shrinkage/ thresholding algorithm"*, 2015
- Chambolle & Pock, *An introduction to continuous optimization for imaging*, 2016
- P. Chen, J. Huang, X. Zhang, *A primal-dual fixed-point algorithm for minimization of the sum of three convex separable functions*, 2016
- Daubechies, Defrise & De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, 2004