Inverse Problems in Image Processing

Effrosyni Kokiopoulou, Martin Plešinger

{effrosyni.kokiopoulou; martin.plesinger}@sam.math.ethz.ch

Welcome to the world of inverse problems!

Outline

- Seminar organization
- Manipulating images in MATLAB
- Motivation & A brief introduction to image deblurring
- Reading assignments

Seminar organization

General info

- Time: **Thursday 15:00–17:00**
- Building: **ML**
- Room: **ML H 34.3**
- First meeting: **September 23, 2010**
- Topic selection: **September 30, 2010**
- First talk: **October 21, 2010**
- Last talk: **December 16, 2010**
- Prerequisities: **Numerical Linear Algebra**
Matrix Computations
MATLAB
- Office hours: **Monday 14:00–15:00**

What do I have to do?

- Regularly **attend** the seminar.
- Give a **successful talk** about the given topic in English.

  Please, meet us for consultation before the talk,

  – 1st approximately **2** weeks before,

  – 2nd approximately **3** days before.

  Send us the presentation slides after the talk.
- Write a short **report** (summary) about the topic (**3** pages).

Optionally

- **MATLAB task**: play with simple 1D problems as well as image deblurring.
- Implement in **MATLAB** some experiments from the reading material.

Send us the **slides** and **report** via e-mail.

Organization

- **2** presentations per week $\Longrightarrow$ **9** weeks

Timetable (Tentative)

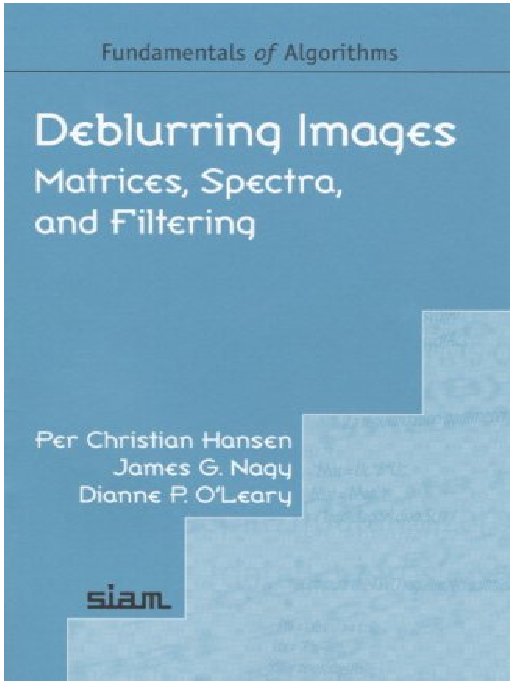
• **1 week** October 21	the first two talks	bachelor st.
• **2 week** October 28		bachelor st.
• **3 week** November 04		bachelor st.
• **4 week** November 11	¿an important date?	BSc/MSc st.
• **5 week** November 18		master st.
• **6 week** November 25		master st.
• **7 week** December 02		master st.
• **8 week** December 09		master st.
• **9 week** December 16	the last two talks	master st.

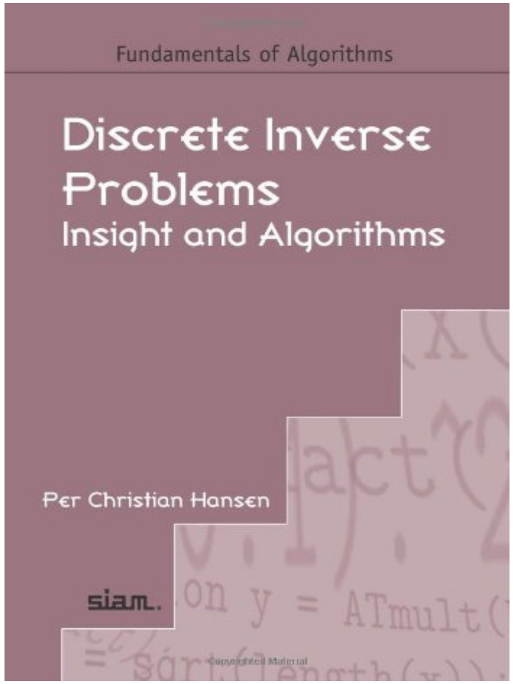
(recommendation only)

Books

Hansen, Nagy, O'Leary: *Deblurring Images, Spectra, Matrices, and Filtering*, SIAM, FA03, 2006 ... recommended for **bachelor** students

Hansen: *Discrete Inverse Problems, Insight and Algorithms*, SIAM, FA07, 2010 ... recommended for **master** students





Books – availability

Legal possibilities:

- Amazon.com (\$130 together, for seriously interested only ;))
- ETHZ library (the second book only)
- Privat

Other possibilities:

- Electronic (pre-version of the first book only)
- Copy

MATLAB code

We recommend:

Regularization Tools

`http://www2.imm.dtu.dk/~pch/Regutools`

by P. C. Hansen,

HNO package

`http://www2.imm.dtu.dk/~pch/HNO`

by P. C. Hansen, J. Nagy, D. O'Leary.

Google "*Per Christian Hansen*" > go to "*Stuff to Download*" > open "*Regularization Tools*" and "*Debluring Images*".

Manipulating images in MATLAB

What is an image?

An image is a vector (matrix or tensor) from a *real* vector space

$$X = \Bigg($$

$$\Bigg) \in \mathbb{R}^{m \times n \times d},$$

where m, n are numbers of rows and columns of the image, i.e. the *height* and *width* in pixels, resp., d is the dimension of a *color space*.

	image color scheme	*color space*	*dimension*
•	grayscale	$[0, 1]$ or $[0, 255]$	$d = 1$
•	RGB	$[0, 1]^3$ or $[0, 255]^3$	$d = 3$

Image Basics

- Images can be color, grayscale or binary.
- Grayscale intensity image: 2D array where each entry contains the intensity value of the corresponding pixel.

There are many types of image file formats:

- GIF (Graphics Interchange Format)
- JPEG (Joint Photographic Experts Group)
- PNG (Portable Network Graphics)
- TIFF (Tagged Image File Format)

Images can be also stored using the "MAT-file" format.

Reading, Displaying and Writing Images

The command `imfinfo` displays information about the images stored in the data file:

```
info = imfinfo('cameraman.tif');
```

The command `imread` loads an image in MATLAB:

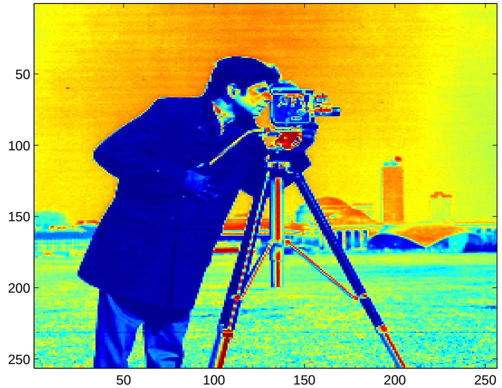
```
I = imread('cameraman.tif');
```

Images can be displayed by three commands:

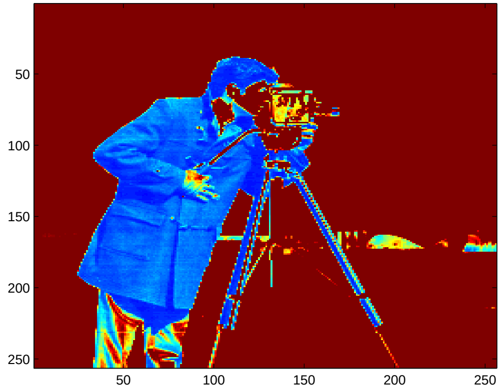
`imshow`, `imagesc`, and `image`.

Display of Images

imshow

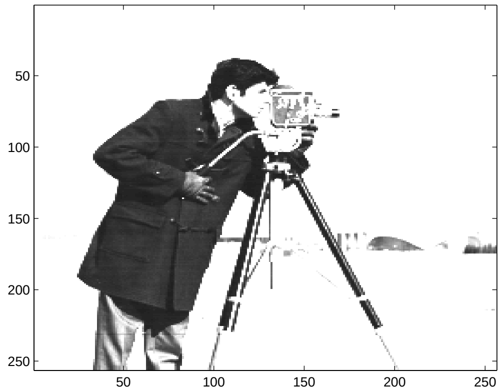


imagesc



image

imagesc, colormap(gray)



image, colormap(gray)

Display of Images (cont'd)

- `imshow` renders images more accurately in terms of size and color
- `image` and `imagesc` display images with a false colormap
- `image` does not provide a proper scaling of the pixel values
- `imagesc` should be combined with `axis image`

Writing of Images

The `imwrite` command writes an image to a file:

```
imwrite(I, 'image.jpg');
```

Arithmetic on Images

- Integer representation of images can be limiting.
- **Common practice**: convert the image to double precision, process it, and convert it back to the original format.
- The function `double` does the conversion: `Id = double(I);`
- When the input image has double entries, `imshow` expects values in [0,1].
- Use `imshow(Id,[])` to avoid unexpected results!
- The function `rgb2gray` can be used to convert color images to grayscale intensity images.

Summary: A very short guide to manipulating images in MATLAB

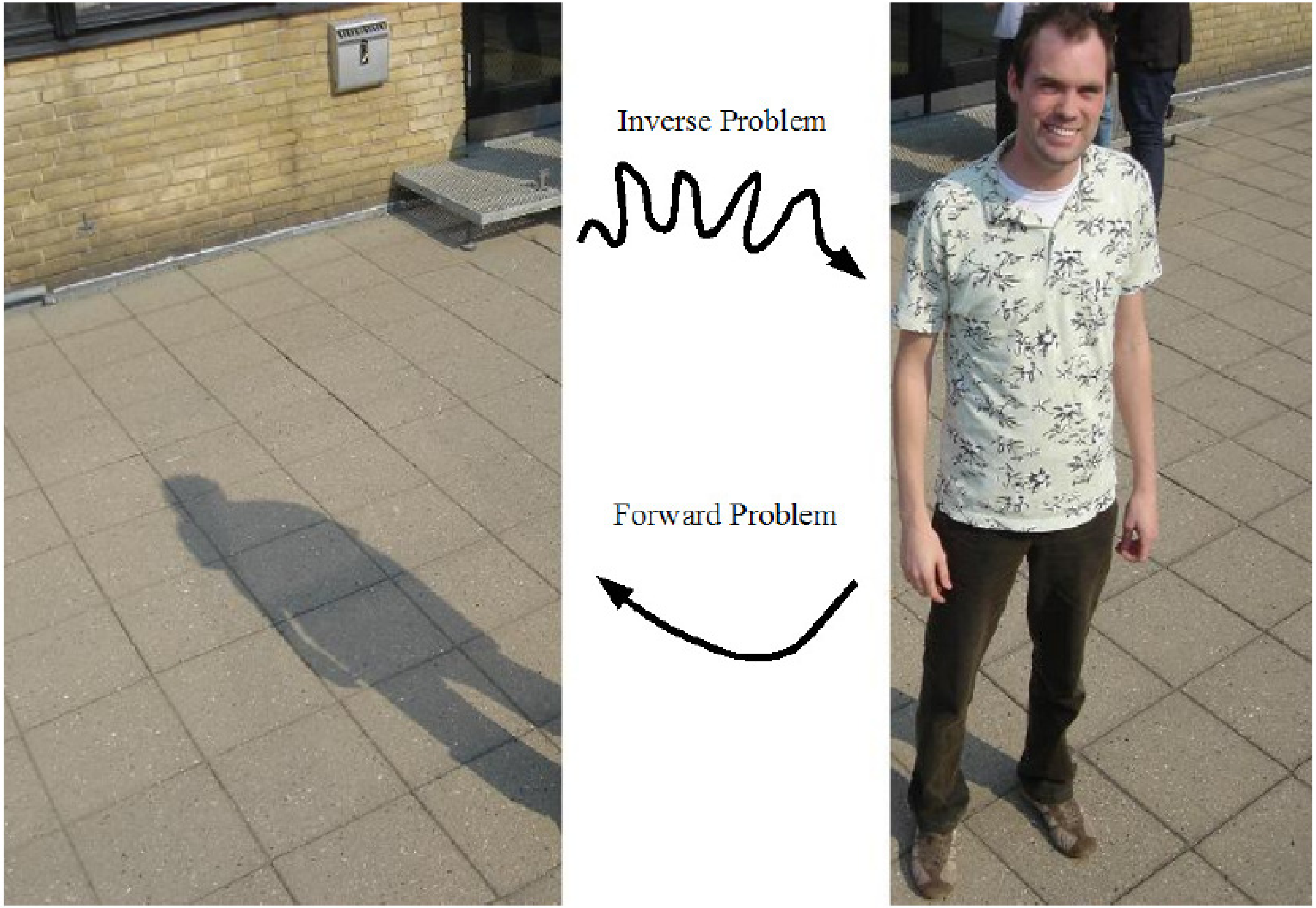
```
X = imread('file.bmp');     % opens an image in MATLAB
X = double(X);              % converts X to doubles

% X is a standard matrix (2D array) in grayscale case
% X is a 3D array in color case, three 2D arrays (for R, G, and B)

colormap gray;              % changes the color scheme in MATLAB
imshow(X);                  % plots the (real) matrix X as an image
imagesc(X);                 % ditto

% the first function is better, but available only with the
% Image Processing Toolbox.

% Convention: min(min(X)) is black, max(max(X)) is white.
```

Further information

- MATLAB (online) help,
- Image Processing Toolbox help,
- Chapter 2 of the first book [H., N., O'L., Deblurring Images],
- Regularization Tools Manual (available online),
- consultations.

Motivation

A gentle start



[Kjøller, Master Thesis, DTU Lyngby, 2007]

Another examples

• *Computer tomography* (CT) maps a 3D object to ℓ X-ray pictures,

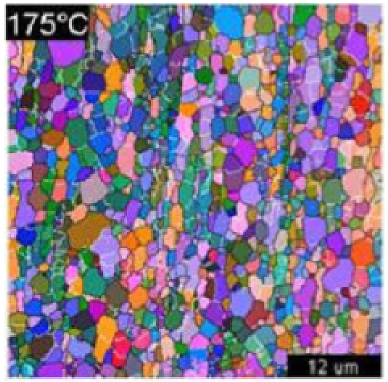
$$\mathcal{A}(\cdot) \equiv$$  $$: \mathbb{R}^{M \times N \times K} \longrightarrow \bigotimes_{j=1}^{\ell} \mathbb{R}^{m \times n}$$

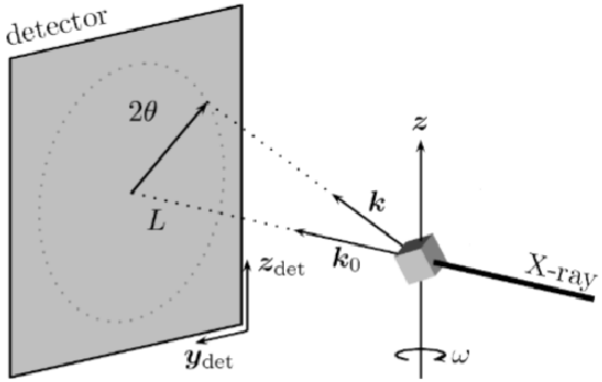
is a mapping (not opeartor) of MNK voxels, to ℓ-times mn pixels.

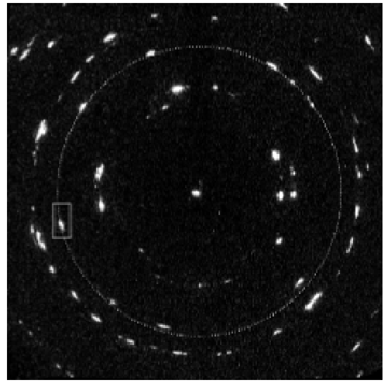
Boundary problem, from real analysis we know that it is (under some assumptions) uniquely solvable. In numerical computations (rounding errors, other sources of noise, e.g. electronic noise on semiconductors PN-junctions in transistors) it is difficult to solve (*Medicine*).

We lose some important information.

• *Transmision tomography* in *crystalographics* is used, e.g., for reconstruction of orientation-distribution-function (ODF) of grains in a polycrystalline material,

$\mathcal{A}\Bigg($ 

$\Bigg) \equiv$ 

$= \Bigg($  $\Bigg).$

the observation (right-hand side) is a vector of difractograms (analogous to the previous example).

- *Seismic tomography* in *geology* is used for reconstruction of boundaries or cracks in tectonic plates in localities with high tectonic activity and frequent earthquakes. A model example of situation in San Francisco craton [Hansen, AIRtools, DTU Lyngby]:

- Reading *bar codes*:

A brief introduction to image deblurring

Linear operators on a vector space $\mathbb{R}^{m\times n\times d}$

Consider for simplicity the grayscale color scheme, thus $d = 1$.

A simple linear operator

$$\mathcal{A}(X) = B, \qquad X, B \in \mathbb{R}^{m\times n} \quad \text{(real matrices)}$$

is, e.g., a rotation, a reflection, a shift operator, etc (all of them are easily invertible, in principle), but there are "ugly" operators, e.g.,

$\mathcal{A}\Bigg($  $\Bigg) =$ 

the *blurring operator*.

The blurring operator is linear and in purely mathematical sense still invertible, but there are *fundamental difficulties* with the *real practical computations* of this **inverse problem**.

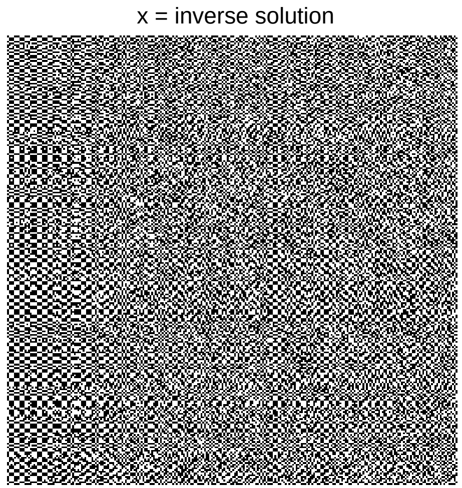
Naive solution of an equation with the blurring operator

Since the operator is linear we can rewrite the problem as a system of linear algebraic equations

$$Ax = b, \qquad A \in \mathbb{R}^{nm \times nm}, \quad x, b \in \mathbb{R}^{nm}$$

and solve it: True x and blured b image, and *naive solution* $A^{-1}b$:

,


,


,

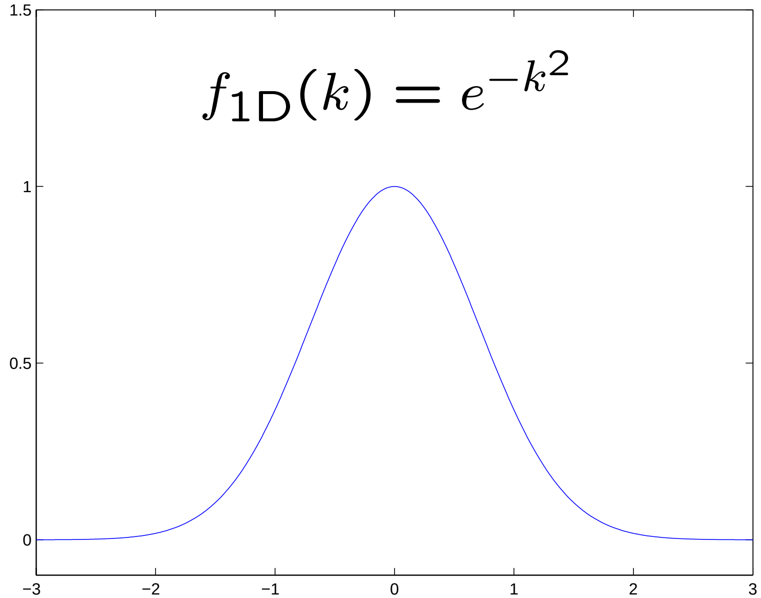
(example by James Nagy, Emory University, Atlanta).

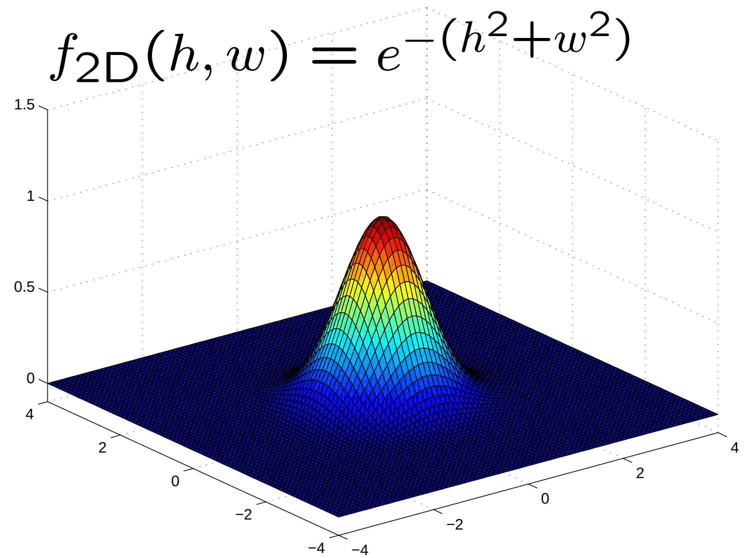
Question: What did it happen **???**

Blurring operator – Gaußian blur

The simplest case is the *Gaußian blurring operator* (used in the previous example).

First recall the Gauß function in 1D and 2D:

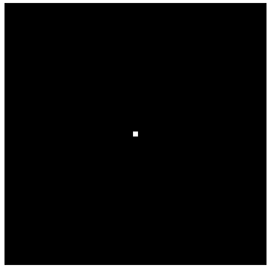
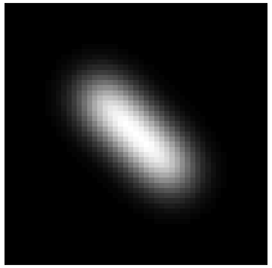




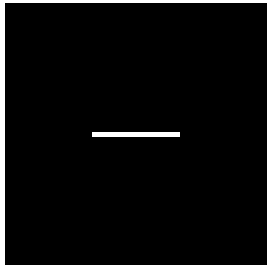
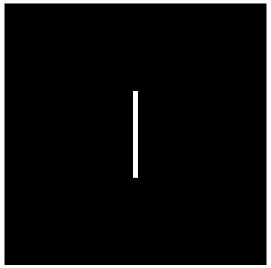
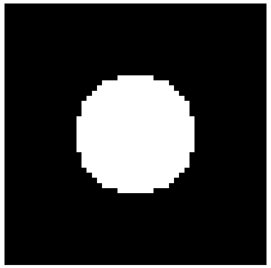
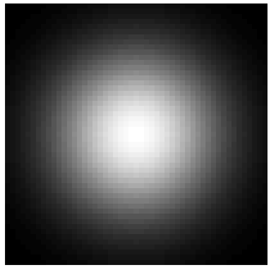
Note that the 2D Gauß function is separable, i.e. can be written as a product of two functions of one variable $f_{2\text{D}}(h, w) = e^{-(h^2+w^2)} = e^{-h^2} e^{-w^2} = f_{1\text{D}}(h) f_{1\text{D}}(w)$.

Point–Spread–Function

The *point-spread-function (PSF)* is a characteristic of a (linear) blurring operator and illustrates how the opearator acts on a single pixel

$\mathcal{A}\Bigg($  $\Bigg) =$  $\equiv PSF_{\mathcal{A}}(h, w) \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$,

where h, w is from *height* and *width*. Examples of PSFs:

horizontal motion blur	vertical motion blur	out-of-focus blur	**Gaußian blur**
 ,	 ,	 ,	 .

Gaußian blur is given by the Gaußisan PSF which is nothing else than the 2D Gauß function (black $\sim$ 0, white $\sim$ 1).

Relatioship between $\mathcal{A}$ and $PSF_{\mathcal{A}}$

The PSF construction from the operator $\mathcal{A}(\cdot) \Longrightarrow PSF_{\mathcal{A}}(w,h)$ is clear from the previous slide.

The (grayscale) image can be represented as a 2D function $X = X(w,h) = x_{w,h}$ (color of the pixel at position (w,h)), the action of $\mathcal{A}$ can be obtained by the *2D convolution* of X with the $PSF_{\mathcal{A}}$.

In (our) *discrete* and *finite* case, recall that

$$X \in \mathbb{R}^{m\times n}, \qquad PSF_{\mathcal{A}} \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$$

are matrices, it is

$$\mathcal{A}(X) = \sum_{\eta=-\ell}^{\ell}\sum_{\nu=-\ell}^{\ell} X(w-\nu, h-\eta) PSF_{\mathcal{A}}(\ell+1+\nu, \ell+1+\eta).$$

Problem: Entries $x_{w,h}$ are not defined for $w = -\ell+1,\ldots,0$, and $m+1,\ldots,m+\ell$ and $w = -\ell+1,\ldots,0$, and $n+1,\ldots,n+\ell$ **!!!**

Question: Is it possible to construct the matrix $\mathcal{A}$ from the PSF **???**

Boundary conditions

The action of the (*inverse*) operator is not defined close to boundary of the image. There are several possibilities, typically:

zero boundary

periodic boundary

reflexive boundary

which involve structure of the matrix in the linear system.

Boundary conditions – 1D problem

PSF is a Töplitz matrix

$$b \equiv \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} p_3 & p_2 & p_1 & & \\ p_4 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 \\ & & p_5 & p_4 & p_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \equiv A_{PSF}\, x,$$

with the boundary condition

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \left[\begin{array}{cc|ccccc|cc} p_5 & p_4 & p_3 & p_2 & p_1 & & & & \\ & p_5 & p_4 & p_3 & p_2 & p_1 & & & \\ & & p_5 & p_4 & p_3 & p_2 & p_1 & & \\ & & & p_5 & p_4 & p_3 & p_2 & p_1 & \\ & & & & p_5 & p_4 & p_3 & p_2 & p_1 \end{array}\right] \left[\begin{array}{c} ? \\ ? \\ \hline x \\ \hline ? \\ ? \end{array}\right].$$

Note: The row $[p_1, p_2, p_3, p_4, p_5]$ is, e.g., the discretized 1D Gauß function $f_{1\mathrm{D}}(k)$.

Boundary conditions – matrix structure

Clearly,

zero boundary: $\longrightarrow [0, 0 | x_1, x_2, x_3, x_4, x_5 | 0, 0]^T$,
periodic boundary: $\longrightarrow [x_4, x_5 | x_1, x_2, x_3, x_4, x_5 | x_1, x_2]^T$,
reflexive boundary: $\longrightarrow [x_2, x_1 | x_1, x_2, x_3, x_4, x_5 | x_5, x_4]^T$.

The linear system $b = A\,x$, where $A = A_{PSF} + M$, and for

zero boundary periodic boundary reflexive boundary

$$M \equiv 0, \qquad M \equiv \begin{bmatrix} & & & p_5 & p_4 \\ & & & & p_5 \\ & & & & \\ p_1 & & & & \\ p_2 & p_1 & & & \end{bmatrix}, \quad M \equiv \begin{bmatrix} p_4 & p_5 & & & \\ p_5 & & & & \\ & & & & \\ & & & & p_1 \\ & & & p_1 & p_2 \end{bmatrix}.$$

Back to the naive solution

There are fundamental difficulties with solving the linear problem:

• Action of the blurring operator $\mathcal{A}$ is realized by convolution with very smooth 2D Gauß function, the operator has a *smoothing property*.

• The (example) right-hand side B is represented by *smooth function*.

• While solving the linear system, i.e. evaluation $\mathcal{A}^{-1}(B)$, we invert a smoothing operator, apply it on a smooth function, and we want to obtain an image X which is *typically discontiuous function*. Recall

b = blurred, noisy image

$B =$  ,

x = true image

$X =$ 

.

Inverse problems $\equiv$ Troubles!

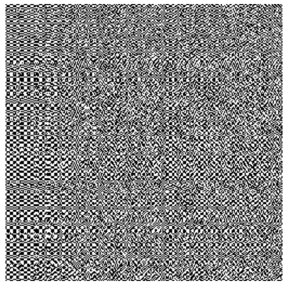
• The problem is *typically very sensitive* to small perturbations (e.g., smooth B, smooth $\mathcal{A}$, nonsmooth solution).

• But B is *always* corrupted by rounding errors (noise). Thus we have system of equations

$$Ax = b + e^{\text{exact}}, \qquad \text{where} \quad e \in \mathbb{R}^{mn} \quad \text{is unknown,}$$

and we want to find $x^{\text{true}} \equiv A^{-1}b$. The naive solution ilustrates the catastophical impact of noise

x = inverse solution

$$X^{\text{naive}} \equiv \mathcal{A}^{-1}(B^{\text{exact}} + E) =$$  .

Instead of the solution we see the *amplified noise* only.

• Condition number of A is *very large*, e.g. $\kappa(A) \approx 10^{100}$, i.e. A is close to singular, we can easily *lose some important information*.

The effect of inverted noise

We have seen that

$$x^{\text{naive}} = A^{-1}b = A^{-1}b^{\text{exact}} + A^{-1}e.$$

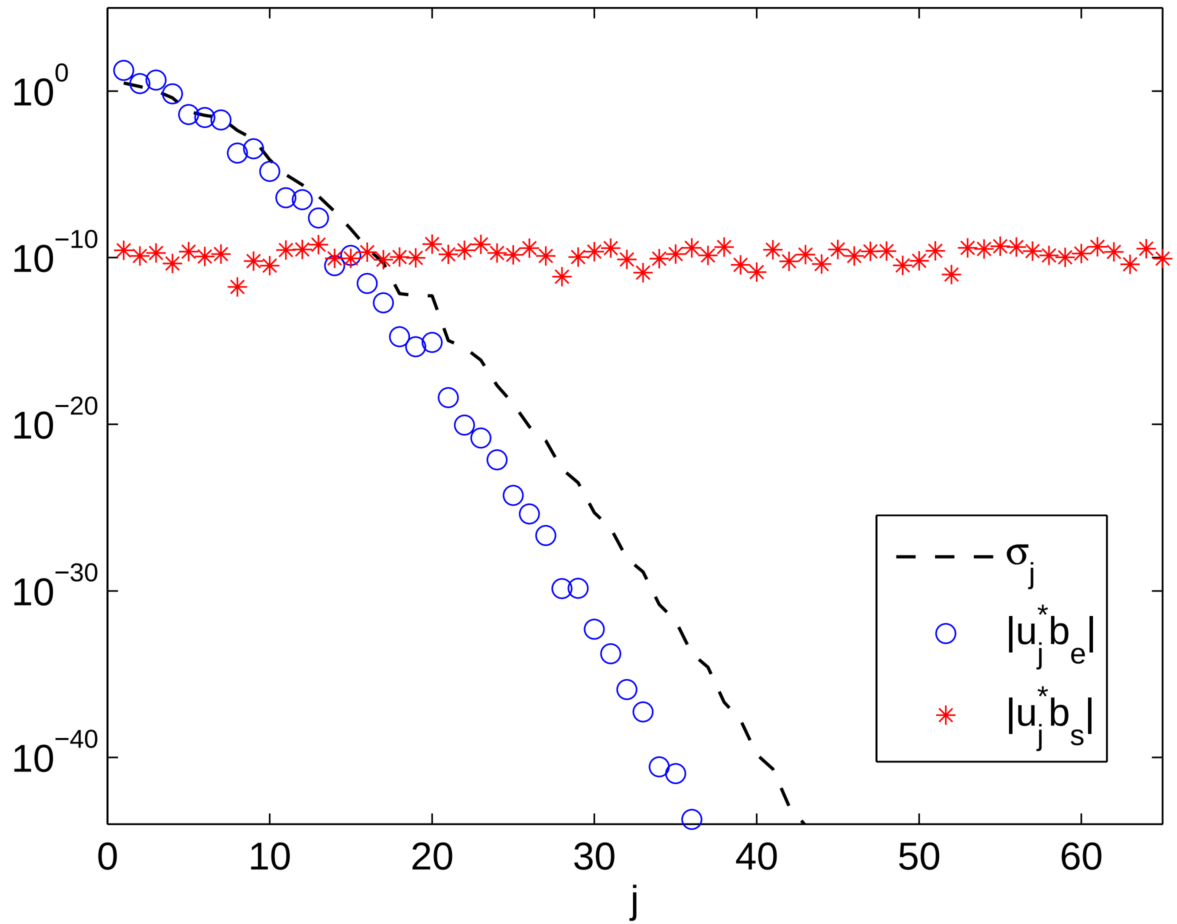
Using the SVD of $A = U\Sigma V^T$, $U = [u_1, \ldots, u_N]$, $U^{-1} = U^T$, $V = [v_1, \ldots, v_N]$, $V^{-1} = V^T$, $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_N)$, $\sigma_1 \geq \ldots \geq \sigma_N \geq 0$,

$$A^{-1}b = V\Sigma^{-1}U^T e = \sum_{i=1}^{N} \frac{u_i^T b}{\sigma_i} v_i, \qquad A^{-1}e = \sum_{i=1}^{N} \frac{u_i^T e}{\sigma_i} v_i, \qquad N = mn.$$

Consider the quantities $\frac{u_i^T e}{\sigma_i}$: when $i \longrightarrow N$ this quantity is divided by σ_i and the contribution of v_i in x^{naive} gets magnified!

The singular vectors corresponding to smallest σ_i represent high frequency information.

The effect of inverted noise — SVD



Don't panic!

We have a plan B!

Using "clever" methods (regularization, spectral filtering) survive the problem

b = blurred, noisy image

$B =$  $\longrightarrow$ $X^{\text{approx}} =$

659 iterations

.

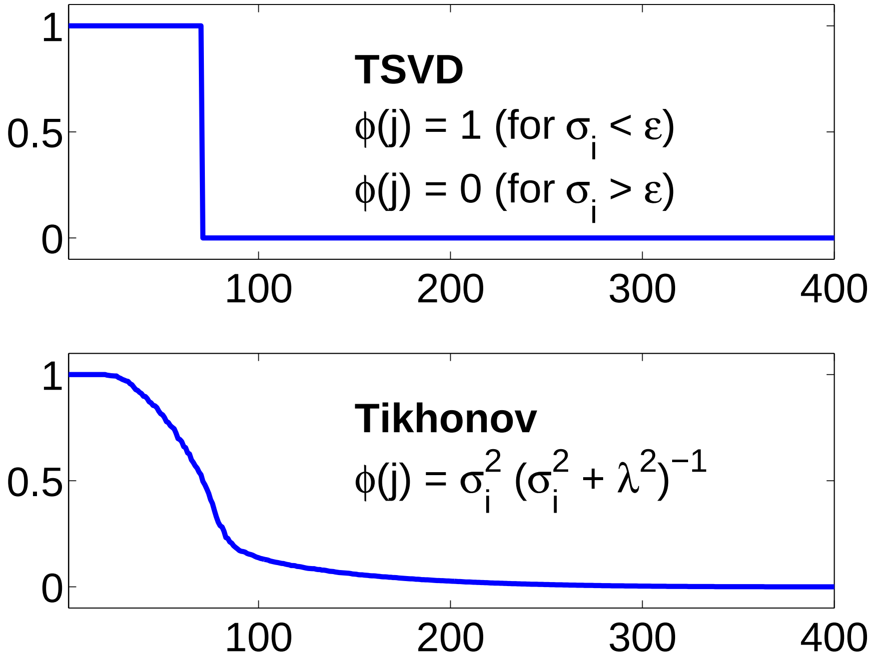
Main idea of regularization methods: Filter out the components corresponding to the small σ_i.

Spectral filtering

Instead of the naive solution use

$$x^{\text{approx}} = \sum_{i=1}^{N} \varphi(i) \frac{u_i^T b}{\sigma_i} v_i = \sum_{i=1}^{N} \varphi(i) \frac{u_i^T (b^{\text{exact}} + e)}{\sigma_i} v_i$$

where $\varphi(i)$ is a **filter** function



Fredholm integral equation of the first kind

Generic form (1D):

$$\int_0^1 \mathcal{K}(s,t) f(t)\,dt = g(s), \quad 0 \le s \le 1.$$

Inverse problem: Given the "kernel" $\mathcal{K}$ and g, compute f (typically g is smooth and f has discontinuities!)

Deconvolution problem: special case of the above

$$\int_0^1 h(s-t) f(t)\,dt = g(s), \quad 0 \le s \le 1.$$

Singular value expansion (SVE) of $\mathcal{K}$: Important analysis tool.

Reading assignments topics

See the list of topics.