Inverse problems course, Exercise 2 (for the week starting on February 6, 2017)
University of Helsinki
Department of Mathematics and Statistics
Samuli Siltanen, Minh Mach, Santeri Kaupinmäki and Alexander Meaney
Related book sections (Mueller & Siltanen 2012): 2.1.2, 2.3.4, 4.1

Theoretical exercises:

T1. Let $\psi : [-a, a] \to \mathbb{R}$ be a non-negative continuous point spread function (PSF) whose integral over $[-a, a]$ equals one. Let $f : \mathbb{R} \to \mathbb{R}$ be a nicely behaving function that has value zero for all $x \leq 0$ and $x \geq 1$. The continuum convolution is defined by

$$(\psi * f)(x) = \int_{-a}^{a} \psi(x') f(x - x')\, dx'. \tag{1}$$

Let $n > 2$ and take $\Delta x = 1/n$. Choose evaluation points for the interval $[0, 1]$:

$$x_j = (j - 1)\Delta x \quad \text{for } j = 1, 2 \ldots, n. \tag{2}$$

Then $f(x)$ is represented by the vector $\mathbf{f} \in \mathbb{R}^n$ containing values at grid points:

$$\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n]^T = [f(x_1), f(x_2), \ldots, f(x_n)]^T \in \mathbb{R}^n. \tag{3}$$

Also, we define a computational representation vector for the PSF:

$$\mathbf{p} = [\mathbf{p}_{-\nu}, \mathbf{p}_{-\nu+1}, \ldots, \mathbf{p}_{-1}, \mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{\nu-1}, \mathbf{p}_\nu]^T \tag{4}$$

with $\mathbf{p}_j = \psi(j\Delta x)$ for $j = -\nu, \ldots, \nu$. Discrete convolution is given by

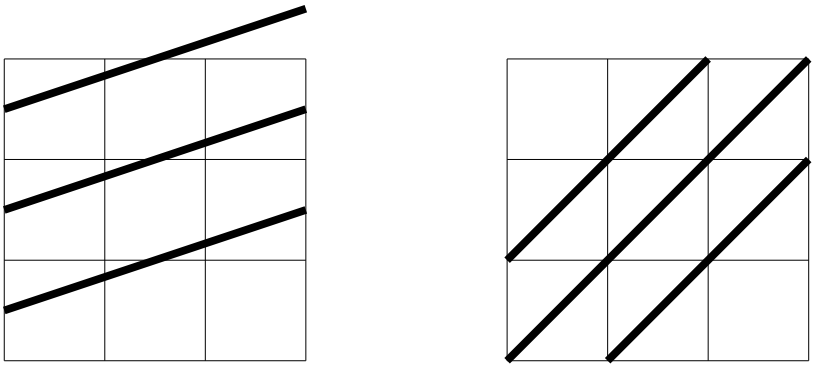
$$(\mathbf{p} * \mathbf{f})_j = \sum_{\ell=-\nu}^{\nu} \mathbf{p}_\ell\, \mathbf{f}_{j-\ell}, \tag{5}$$

where $\mathbf{f}_{j-\ell}$ is defined using zero extension for the cases $j - \ell < 1$ and $j - \ell > n$.

(a) What is a good choice for ν when n is given?

(b) Explain why $(\psi * f)(x_j) \approx \Delta x (\mathbf{p} * \mathbf{f})_j$ with the approximation getting better as $n \to \infty$. You do not have to give a rigorous mathematical proof. Study the discussion in Section 2.1.2 of the textbook and convince yourself (and the teaching assistant) that the claim holds.

T2. Thin lines depict pixels and thick lines X-rays in this image:



Give a numbering to the nine pixels ($f \in \mathbb{R}^9$) and to the six X-rays ($m \in \mathbb{R}^6$), and construct the matrix A for the measurement model $m = Af$. The length of the side of a pixel is one.

Hint: see Section 2.3.4 of the textbook.

Matlab exercises:

M1. Consider equations $x_1 + x_2 = 1$, $x_2 = -2$ and $-\frac{1}{3}x_1 + x_2 = -2$.

(a) Write the equations in the matrix form $Ax = y$. (That is, specify the elements in the 3×2 matrix A and the vector $y \in \mathbb{R}^3$.)

(b) Use Matlab to compute the singular value decomposition $A = UDV^T$.

(c) Read Section 4.1 of the textbook. Using the result of (b), construct D^+ and the minimum norm solution $x^+ := VD^+U^Ty$ in Matlab. Draw the three lines specified by the equations and the point x^+ in the (x_1, x_2)-plane. Discuss the result.

M2. **Computational version of Exercise T1 above.** Let f and ψ be defined by the Matlab routines *targetf.m* and *PSF.m* written in the lectures.

(a) Choose n and use the routine *deco02_data_comp.m* to evaluate $(\psi * f)(x_j)$ for all x_j defined in (2). Call the resulting vector $\widetilde{\mathbf{m}} \in \mathbb{R}^n$ and plot it using blue dots.

(b) Using the same n than in (a), compute $\Delta x(\mathbf{p} * \mathbf{f})_j$ defined in (5) in vector form $A\mathbf{f}$, including the multiplication by the constant Δx. In other words, use the command `convmtx` to construct the convolution matrix A.

Call the resulting vector $\mathbf{m} = A\mathbf{f}$ and plot it (to the same plot than $\widetilde{\mathbf{m}}$; remember to use the command `hold on`) using red dots. Are the blue and red dots close to each other?

(c) Define *relative square norm error* between the vectors $\mathbf{m} \in \mathbb{R}^n$ and $\widetilde{\mathbf{m}} \in \mathbb{R}^n$ by the formula

$$\frac{\|\widetilde{\mathbf{m}} - \mathbf{m}\|_2}{\|\widetilde{\mathbf{m}}\|_2} \cdot 100\%,$$

where $\|\mathbf{m}\|_2 = (\mathbf{m}_1^2 + \mathbf{m}_2^2 + \ldots + \mathbf{m}_n^2)^{1/2}$ is the usual Euclidean square norm. In Matlab you can use the command `norm` to compute it.

How large do you have to take n in (a) and (b) to make the relative square norm error between $\mathbf{m}$ and $\widetilde{\mathbf{m}}$ smaller than 1%? How about 0.1%?