The Chinese University of Hong Kong, Shenzhen
SDS · School of Data Science
Andre Milzarek · Fall Semester 2024-25

DDA 3005 — Numerical Methods

Course Project

Image Deblurring and QR Factorizations

Problem 1 (Course Project): (approx. *100* pts)
The goal of this course project is to investigate and implement different algorithms (including QR factorizations, least squares models, etc.) to reconstruct blurry images.

Project Description and Tasks. General deblurring problems can be modeled in the following way. Given a blurry image $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ and two blurring kernels $\boldsymbol{A}_\ell \in \mathbb{R}^{n\times n}$ and $\boldsymbol{A}_r \in \mathbb{R}^{n\times n}$, we consider linear systems of the form:

$$\boldsymbol{A}_\ell \boldsymbol{X} \boldsymbol{A}_r = \boldsymbol{B}. \tag{1}$$

We seek to recover the original image $\boldsymbol{X} \in \mathbb{R}^{n\times n}$, i.e., we can set $\boldsymbol{X} = \boldsymbol{A}_\ell^{-1}\boldsymbol{B}\boldsymbol{A}_r^{-1}$ if the matrices $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$ are invertible.

The problem (1) can be used to model the deblurring problem presented in one of our introductory lectures. A typical format for the blurring kernels $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$ is

$$\boldsymbol{A} = \begin{bmatrix} a_n & a_{n-1} & & a_2 & a_1 \\ a_{n+1} & a_n & a_{n-1} & & a_2 \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & a_{n-1} \\ a_{2n-2} & & & a_{n+1} & a_n \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad a_i \in \mathbb{R}, \quad \forall\, i.$$

For instance, if $\boldsymbol{A}$ is symmetric with $a_i \geq 0$ and $a_1 + \cdots + a_n = 1$, then the corresponding blurring kernel computes *weighted averages* of pixels in the matrix $\boldsymbol{X}$ which results in the blurry image $\boldsymbol{B}$. Furthermore, in order to generate *motion-type blurring*, we can consider the specific choice:

$$\begin{bmatrix} a_{n+j} & a_{n+j-1} & \dots & a_{n+j-k+1} \end{bmatrix} = \frac{2}{k(k+1)} \begin{bmatrix} k & k-1 & \dots & 1 \end{bmatrix} \quad \text{and} \quad a_i = 0 \tag{2}$$

for all $i > n+j$ and $i < n+j-k+1$ and suitable $j, k \in \mathbb{N}$. Let us consider two examples:

- We consider $\boldsymbol{A}$ with $n = 5$, $j = 0$, $k = 2$ and $\boldsymbol{B}$ with $n = 5$, $j = 1$, $k = 3$. Then, for $\boldsymbol{A}$, we have $n + j = 5$, $n + j - k + 1 = 5 - 2 + 1 = 4$, and $[a_5,\, a_4] = \frac{1}{3}[2,\, 1]$. This implies that the matrix $\boldsymbol{A}$ is diagonal with an additional superdiagonal (corresponding to a_4). For $\boldsymbol{B}$, we obtain $n + j = 6$, $n + j - k + 1 = 4$, and $[b_6,\, b_5,\, b_4] = \frac{1}{6}[3,\, 2,\, 1]$. Overall, this yields

$$\boldsymbol{A} = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} & 0 & 0 & 0 \\ 0 & \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ 0 & 0 & \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & 0 & 0 & \frac{2}{3} \end{bmatrix} \in \mathbb{R}^{5\times 5} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} \frac{1}{3} & \frac{1}{6} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{6} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{3} & \frac{1}{6} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{3} \end{bmatrix} \in \mathbb{R}^{5\times 5}.$$

Several illustrating examples of the blurring operation (1) are presented below in Figure 1.

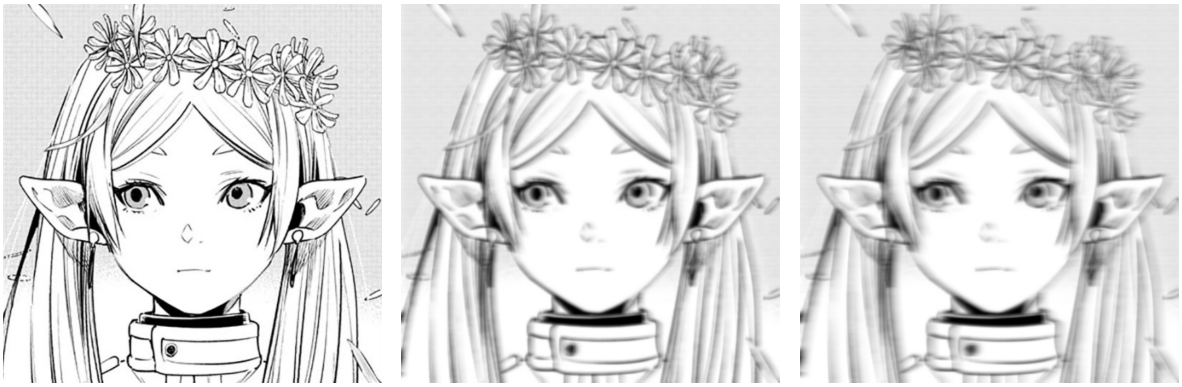

Figure 1: Left: Original image $\boldsymbol{X}$. Middle and Right: Blurred images.

a) The blurry images shown in Figure 1 are created using kernels following the format (2):

$$\begin{aligned} &\text{for } \boldsymbol{A}_\ell: \quad j = 0,\ k = 12 \quad (\boldsymbol{A}_\ell \text{ is an upper triangular matrix}) \\ &\text{for } \boldsymbol{A}_r: \quad j = 1,\ k \in \{24, 36\} \quad (\boldsymbol{A}_r \text{ is upper triangular with an extra subdiagonal}) \end{aligned}$$

On BB, we have provided a test set of original images $\boldsymbol{X}$ with different sizes n. The images are saved using the format

`n_name_original.png`

to indicate n. Download the test image package from BB. Select several images and construct suitable blurring kernels $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$ as well as the corresponding blurry images $\boldsymbol{B}$. Select at least two different images and create two different blurry versions of the images. You can follow the outlined construction of $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$ (other blurring kernels are also possible).

Hint: The test image package also contains two test files that showcase how to load, read, and write image files in `MATLAB` and `Python`.

b) Design and implement an algorithm in `MATLAB` and `Python` that can solve problem (1) and recover the original image $\boldsymbol{X}$ from a given blurry image $\boldsymbol{B}$ and known blurring kernels $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$. Specifically, develop two suitable codes for solving (1) that are based on

- LU factorizations of $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$
- QR factorizations of $\boldsymbol{A}_\ell$ and $\boldsymbol{A}_r$

and test the methods on the generated problems from part a). For each tested image report your result (i.e., plot your reconstructed image), the required cpu-time, and the relative forward error (using the Frobenius norm). What are your observations? Which version returns better results? Can you explain your observations?

You are allowed to use `MATLAB` or `Python` inbuilt operations to compute the LU factorizations, the QR factorizations, etc.

c) Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a given matrix with $m \geq n$. Implement the (your own) Householder QR factorization $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$ in `MATLAB` or `Python` (with our without column permutations).

You can follow the steps and derivations discussed in lectures L-12 and L-13. It is not necessary to return the full orthonormal matrix $\boldsymbol{Q}$, but your code should be (potentially) adjustable so that it can be used for the application in part b).

d) Repeat the task in part b) using your own implementation of the QR factorization. What are your observations? Does your code return the same or similar results?

Hint: You do not need to beat the inbuilt QR factorizations provided by `MATLAB` or `Python`.

e) The reconstructions obtained in part b) and d) can still be affected by numerical errors and might not be perfect. Based on your results in part b), discuss potential improvements or more robust versions of your algorithm. In particular, consider the least-squares formulation of (1),

$$\min_{\boldsymbol{X}} \|\boldsymbol{A}_\ell \boldsymbol{X} \boldsymbol{A}_r - \boldsymbol{B}\|_F^2,$$

as a potential direction and check whether the blurring kernels $\boldsymbol{A}_\ell$ and/or $\boldsymbol{A}_r$ are (approximately) singular. Explain your adjustments and discuss the modified results.

General Hints and Guidelines:

- You can use the peak-signal-to-noise ratio

$$\texttt{PSNR} = 10 \log_{10}(255^2 n^2 / \|\boldsymbol{X}_{\texttt{rec}} - \boldsymbol{X}\|_F^2)$$

to measure the quality of your reconstructions $\boldsymbol{X}_{\texttt{rec}}$. (If you have rescaled the images to $[0, 1]$, then you do not need the factor "255^2" in `PSNR`).

- You can resize/rescale the images to test your implementation first on simpler and low-dimensional examples.
- It is possible to pad the images, i.e., to extend the image borders with additional white pixels (or other suitable colors). This can result in a smoother/more natural blurred image.

Project Report. This is an individual course project.

A report should be written to summarize the project and to collect and present your different results. The report itself should take no more than 10–15 typed pages plus a possible additional appendix. It should cover the following topics and items:

- What is the project about?
- What have you done in the project? Which algorithmic components have you implemented?
- Summarize your main results and observations.
- Describe your conclusions about the different problems and methods that you have studied.

You can organize your report following the outlined structure in this project description.

Try to be brief, precise and fact-based. Main results should be presented by highly condensed and organized data and not just piles of raw data. To support your summary and conclusions, you can put more selected and organized data into an appendix which is not counted in the page limit.

The deadline for the report submission is **December, 9th, 11:59pm (night)**. Please submit your report (as pdf-document) and all supporting materials and code online using Blackboard.