Applications of matrix computations, Spring 2018 Exercise 6

Lecturer: Samuli Siltanen.
Teaching assistants: Vesa Kaarnioja and Jesse Railo.

Deadline: Monday 07.05. at 10.00. *Sending your solutions late is not allowed!*

The exercises are held in the room C321 of Exactum on Wed 14.15–16.00 (group 1), Thu 12.15–14.00 (group 2), Fri 14.15–16.00 (group 4), and in the computer lab C128 of Exactum on Fri 10.15–12.00 (group 3). You gain credits from this course only by completing the weekly exercises. Participating in the exercise group is not obligatory for completing the course, but it is highly recommended.

Return your solutions as a single PDF file containing your Matlab codes (as text), results, images, comments, and explanations at the Moodle page of the course before the deadline (further instructions are given at the exercise groups if needed). Please include your name, student number, and your exercise group number in the beginning of the PDF file.

1. **Differentiation and Fourier transform.**

   (a) Evaluate the periodic continuous function `PeriodicFun1.m` on a grid with 512 evaluation points in the interval $[0, 2\pi]$. Denote the difference between consecutive grid points by h.

   (b) Compute the convolution of the function with the two-element point spread function $h^{-1}[-1\ 1]^T$. (This is approximate differentiation using a simple *finite difference method*.) Hint: see `doc conv` and `doc conv2`; you can check that your result is correct by comparing it to `diff(f)/h`, where `f` contains the values of `PeriodicFun1` on the grid points.

   (c) Compute the FFT of both the function and its numerical derivative. Divide the latter by the former. Can you see what operation in the frequency domain corresponds to differentiation in the function domain? How does your finding fit with the formulation given at this Wikipedia page?

2. (Corrected 30.4.2018) Let $f\colon [0,1] \to \mathbb{R}$ be a function and $x =$ `linspace(0,1)`. Let `f` $\in \mathbb{R}^n$ be the vector that evaluates f at the points given by x and $h =$ `x(end)-x(end-1)`.

   (a) Create the matrix $A =$ `h*tril(ones(100))` that calculates the discrete integral of f.

   (b) Create the matrix $B =$ `(1/h)*(diag(ones(100,1))-diag(ones(99,1),-1))` that calculates the discrete derivative of f.

   (c) Verify (numerically) that $AB = BA = I$.

   (d) Test your integration and differentiation routines A, A^T, B, and B^T to the exponential function $f(t) = e^t$. Plot your results and compare them with the knowledge that $f'(t) = f(t) = F(t)+C$ for some constant $C \in \mathbb{R}$ and $F' = f$. (Hint: you may need to drop the first and last coordinates from consideration.) Explain how the matrix A^T acts on f? Why do A and B act one way, but A^T and B^T another way?

3. Consider the following harmonic inpainting problem with six pixels:

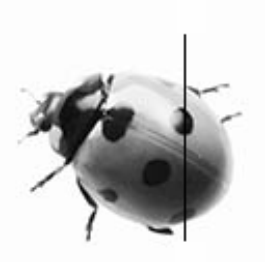
	100	100	
20	x_1	x_4	0
20	x_2	x_5	0
20	x_3	x_6	0
	100	100	

   (a) Write down the six linear equations describing the average equations.

   (b) Form a 6×6 matrix problem $Ax = b$ out of the linear equations.

   (c) Check that your matrix matches with that coming our of the routine `FD_Laplace.m` available at the course website. Read through the code in `FD_Laplace.m` and understand how it works.

4. Download the grayscale images *Ladybug.png* and *Ladybug2.png*. They look like this:

   There are one hundred pixels (in rows 17–116) missing in column 89 of *Ladybug2.png*, seen as a vertical black stripe. Use harmonic inpainting to fill the column, and compare the result to *Ladybug.png*. In other words,

   - call the missing pixel values $x_1, x_2, \ldots, x_{100}$,
   - organize them in a vertical vector $x = [x_1, x_2, \ldots, x_{100}]^T \in \mathbb{R}^{100}$,
   - write a system $Ax = b$ of linear equations requiring that each pixel x_j has average value of its four nearest neighbors (it is a good idea to initialize A as a sparse matrix with the command `A=sparse(100,100)`),
   - solve the 100×100 system using the Matlab command `x = A\x`, and finally
   - insert the inpainted pixel values into the image.

   Note that you can use the knowledge of the locations of the pixels. You do not have to find the "missing" pixels yourself.

5. (a) **Defusing a photobomb, or hippo removal.** The Matlab routine `Photobomb_defuse.m` removes a hippo from the red channel of the image *Photobomb.jpg* as shown here:


Perform hippo removal for the color image *Photobomb.jpg* using the code `Photobomb_defuse.m` as your starting point.

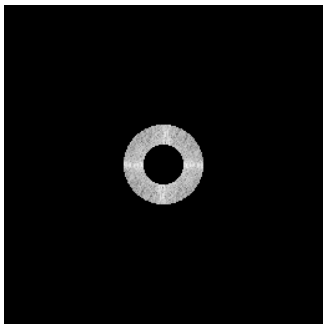
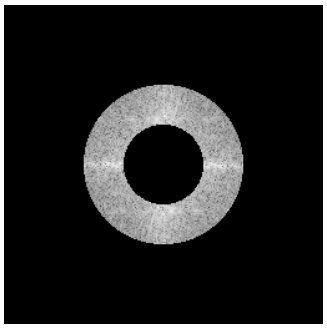
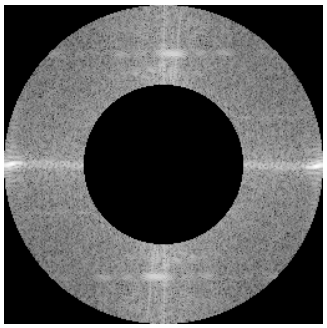
(b) **Defusing a photobomb of your own.** Using your smartphone or any digital camera, take a photograph containing an "unwanted object."

Use finite difference solution of the Poisson equation to remove the object from the photo.

Hint: the removal with this method works best if the background behind the unwanted object is smoothly varying or constant.

6. **Band-pass filtering of images.** Download the image file *LobsterFight1024_1.png* and the Matlab routine `ImageFFT_test.m`. It produces, for example, a low-pass filtered image.

(a) Modify the file `ImageFFT_test.m` so that you get the following "band-pass" filters in the frequency domain.

(b) Choose some small area in the image with a lot of detail. Show, side by side, that area of the original image, the low-pass filtered image, the high-pass filtered image and all the band-passed images of (b).