Signal and Image Processing - Assignment 2

Assigned March 18$^{\text{th}}$ **Due April 14$^{\text{th}}$ 23:55**

Time you spent on this assignment: ____ hours

You need to submit a PDF report and one Python ($\geq$ v.3.6) function or script for each task, all in the same folder. Make sure that each file is named according to the task. You can also choose to upload a Jupyter Notebook containing both the functions and the report. Please write (at least) your name and student ID into each file you submit. Feel free to define any other auxiliary functions.

The maximum amount of points awarded for this assignment is **15 points**.

1 Image Enhancement [3 Points]

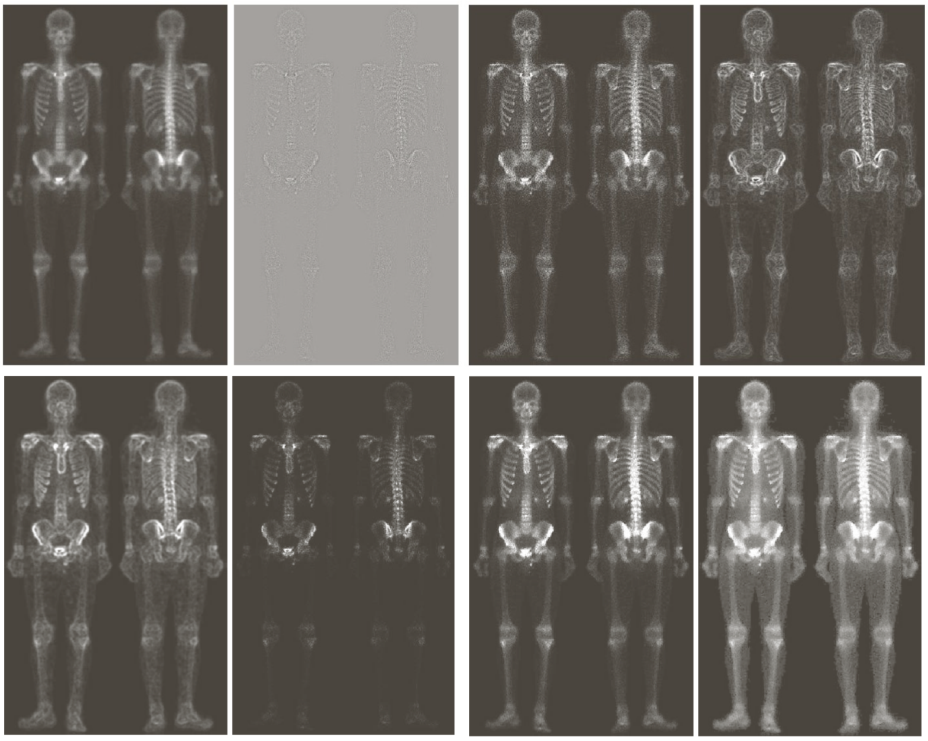


FIGURE 3.43
(a) Image of whole body bone scan. (b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b). (d) Sobel gradient of (a). (e) Sobel image smoothed with a 5×5 averaging filter. (f) Mask image formed by the product of (c) and (e). (g) Sharpened image obtained by the sum of (a) and (f). (h) Final result obtained by applying a power-law transformation to (g). Compare (g) and (h) with (a). (Original image courtesy of G.E. Medical Systems.)

Figure 1: Image Enhancement (high-res image in slides)

Download the original image (a) in Figure 1 from `http://vda.univie.ac.at/Teaching/SIP/18s/Lab2/343a.tif`. Start by gamma correcting the image by a factor of 0.6. Then generate steps (b) through (h) by applying the respective filters in Python. Choose a suited factor for the power-law transform in the last step to visually match the resulting image (h) from above. The resulting program should output 8 images in order, starting from the corrected image (a).

For this section you are allowed to use all built-in functions of SciPy, OpenCV and scikit-image.

We will test this task by calling `python task1.py` or in the respective section of the Jupyter Notebook, so make sure this displays all (properly labelled) figures from this task.

2 Convolution [2 Points]

Define your own 1-dimensional convolution function with the name `conv1`, which takes two timesignals as input and returns the convolved timesignal as output. The function must be able to handle timesignals of any lengths.

For this task, you are only allowed to use the NumPy library. The use of `numpy.convolve` as well as the use of the SciPy or any other library is prohibited.

We will test this by calling `from task2 import conv1` and then `conv1(x, y)`, where x and y are two arbitrary length time signals. In Jupyter you are supposed to define x and y yourself and we will change the values to test it.

Hint Don't confuse convolution with correlation!

3 LTI Systems [2 Points]

The input-output pair shown in Figure 2 is given for a stable LTI-system.

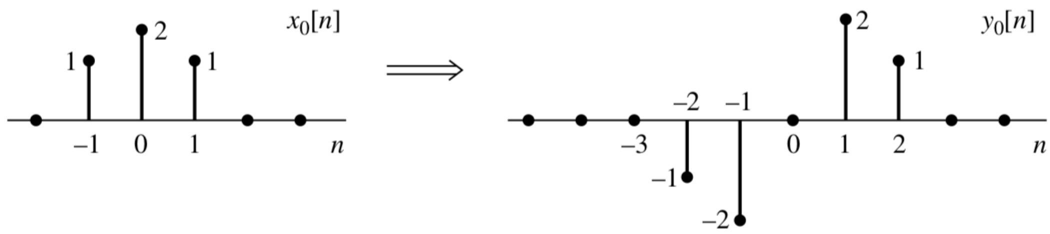


Figure 2: Input and output for a stable LTI-system

(a) Calculate the impulse response of the system. State the steps of the calculation in your report.

(b) Determine the response to the input $x_1[n]$ in Figure 3:

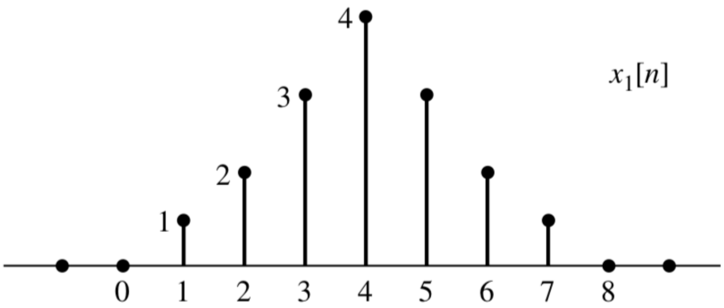


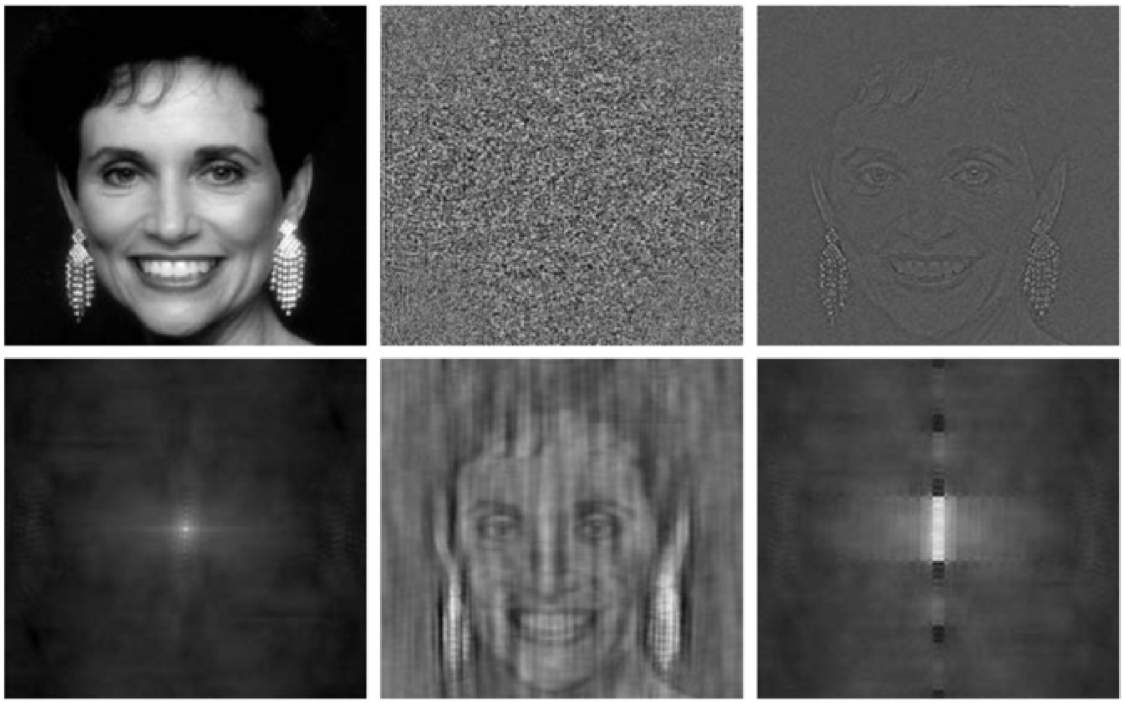
Figure 3: Input for the stable LTI system

For this task, you should use your own convolution function from section 2 and the same restrictions apply.

Show the results for a and b as a stem-plot. Make sure your stem-plots use the correct indices.

We will test this by calling `python task3.py` or by running the respective section of the Jupyter Notebook.

4 2D Discrete Fourier Transform [4 Points]



a b c
d e f

FIGURE 4.27 (a) Woman. (b) Phase angle. (c) Woman reconstructed using only the phase angle. (d) Woman reconstructed using only the spectrum. (e) Reconstruction using the phase angle corresponding to the woman and the spectrum corresponding to the rectangle in Fig. 4.24(a). (f) Reconstruction using the phase of the rectangle and the spectrum of the woman.

Figure 4: Figure 4.27 from the *Gonzalez & Woods* book (3$^{\text{rd}}$ edition).

The input images can be downloaded from the website:
`http://vda.univie.ac.at/Teaching/SIP/19s/Lab2/424a.png`
`http://vda.univie.ac.at/Teaching/SIP/19s/Lab2/427a.png`

Given the two input images produce the same output as shown in Figure 4, namely all 6 panels of that figure, as denoted in the caption. Be aware that the caption does not state all requirements to produce these images and you might need to experiment a bit.

For this task you are only allowed to use the NumPy library.

We will test this by calling `python task4.py` or by running the respective section of the Jupyter Notebook. Make sure the figures are labelled correctly.

5 Image Enhancement in Frequency Domain [4 Points]

Read the image `blurry-moon.tif` and sharpen it using unsharp masking. Use a Gaussian lowpass filter of your choice in the frequency domain for the blurring step. Then, improve the sharpness of your result using high-boost filtering.
[You should be able to achieve a result close to the right one in Figure 5]

Your outputs should at least show: **(a)** the original image, **(b)** the Gaussian frequency filter, **(c)** the blurred image, **(d)** the unsharp mask, **(e)** the enhanced image with unsharp masking applied, and **(f)** the enhanced image with high-boost filtering applied.

The image can be downloaded from the website:
`http://vda.univie.ac.at/Teaching/SIP/19s/Lab2/blurry-moon.tif`

Figure 5: [left] Original, blurry image; [right] enhanced image; Figure 4.56 from the *Gonzalez & Woods* book (4^{th} edition).

We will test this by calling `python task5.py` or by running the respective section of the Jupyter Notebook. Make sure the figures are labelled correctly.

Hint You might have a closer look at the *Summary of steps for filtering in the frequency domain* as well as equation (4-116) in the *Gonzalez & Woods* book (4^{th} edition).