

Homework 4

Daniel Sawyer

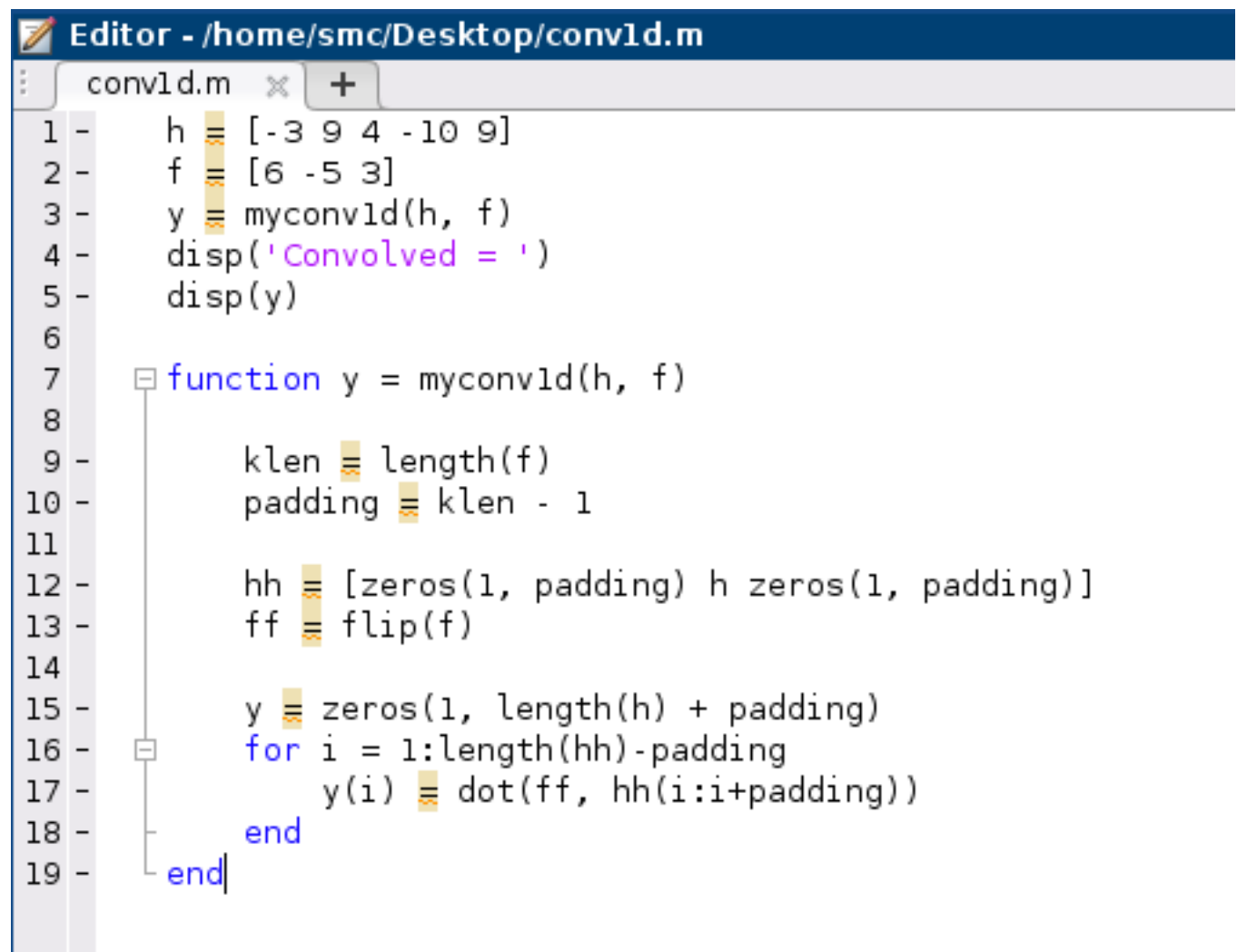
April 7, 2021

Exercise 1

1D and 2D Convolutions

a) Write your own Matlab function to implement 1D discrete convolution, between two vectors:

For the matlab script I calculate the padding needed then run the convolution. Code is pretty simple.



```
Editor - /home/smc/Desktop/conv1d.m
conv1d.m x +
1 - h = [-3 9 4 -10 9]
2 - f = [6 -5 3]
3 - y = myconv1d(h, f)
4 - disp('Convolved = ')
5 - disp(y)
6
7 - function y = myconv1d(h, f)
8
9 -     klen = length(f)
10 -     padding = klen - 1
11
12 -     hh = [zeros(1, padding) h zeros(1, padding)]
13 -     ff = flip(f)
14
15 -     y = zeros(1, length(h) + padding)
16 -     for i = 1:length(hh)-padding
17 -         y(i) = dot(ff, hh(i:i+padding))
18 -     end
19 - end
```

$$h = \begin{bmatrix} -3 \\ 9 \\ 4 \\ -10 \\ 9 \end{bmatrix}, \text{ and } f = \begin{bmatrix} 6 \\ -5 \\ 3 \end{bmatrix}$$

To compute by hand you will need to flip f and slide it over h like so:

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 3 & -5 & 6 & & & & & & \end{bmatrix} = [-18]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 3 & -5 & 6 & & & & & \end{bmatrix} = [-18 \quad 69]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 0 & 3 & -5 & 6 & & & & \end{bmatrix} = [-18 \quad 69 \quad -30]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 0 & 0 & 3 & -5 & 6 & & & \end{bmatrix} = [-18 \quad 69 \quad -30 \quad -53]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & -5 & 6 & & \end{bmatrix} = [-18 \quad 69 \quad -30 \quad -53 \quad 116]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & -5 & 6 & \end{bmatrix} = [-18 \quad 69 \quad -30 \quad -53 \quad 116 \quad -75]$$

$$\frac{h}{f} = \begin{bmatrix} 0 & 0 & -3 & 9 & 4 & -10 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & -5 & 6 \end{bmatrix} = [-18 \quad 69 \quad -30 \quad -53 \quad 116 \quad -75 \quad 27]$$

$$\therefore y = h * f = [-18 \quad 69 \quad -30 \quad -53 \quad 116 \quad -75 \quad 27]$$

b) In this question, you will compute the convolution between an image of your choice, and the Gaussian filter. The PSF for the Gaussian filter is:

$$H = \frac{1}{320} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

For this problem, we have to load the image in matlab, convolve with the 2D kernel H , and display the results. The code is fairly simple and in this case I used a border as the kernel is pretty small being 5×5 . To calculate the border you use $border = (kernel_size - 1)/2$ and in this case $border = (5 - 1)/2 = 2$ as you will not notice a 2 pixel border at all with images. The Gaussian kernel provides what is called Gaussian smoothing or blur and is a way to remove noise/detail from in image in order to extract features like lines and shapes. Once convolved, it looks slightly blurrier than the original. Below is the code, original image, and blurred/convolved image.

```
Editor - /home/smc/Desktop/conv2d.m
conv2d.m
1 - img_path_in = '~/Desktop/lenna_bw.png';
2 - img_path_out = '~/Desktop/output_conv2d.png';
3 - h = imread(img_path_in);
4 - k = [1 4 6 4 1; 4 16 24 16 4; 6 24 36 24 6; 4 16 24 16 4; 1 4 6 4 1] / 320;
5
6 - y = myconv2d(h, k);
7 - imwrite(y, img_path_out);
8
9 - function y = myconv2d(h, k)
10 -     padding = (length(k) - 1) / 2;
11 -     hdims = size(h);
12 -     h = double(h);
13 -     k = double(k);
14
15 -     y = double(zeros(hdims));
16 -     p = padding;
17 -     for i = 1+p:hdims(1)-p
18 -         for j = 1+p:hdims(2)-p
19 -             hs = h(i-p:i+p, j-p:j+p);
20 -             y(i,j) = sum(hs .* k, 'all');
21 -         end
22 -     end
23
24 -     y = uint8(y);
25 - end
```

Lenna Original



Lenna Convolved



Exercise 2

Digital Holography

Digital holographic reconstruction: Inverse Write Matlab scripts to complete the following questions. Submit both your Matlab scripts as well as the output results. The object, $f2$, located at $z_1 = 50\text{ mm}$ is given in the mat-file `02.mat`. Use the following parameters throughout this problem for forming the hologram: $\lambda = 0.5\mu\text{m}$, $N = 1000$, $\Delta = \delta = 5\mu\text{m}$. Consider the object `o2` is present.

a) **Simulate the hologram using the discrete forward model ($y = Ax$) derived from Homework #3. (If you didn't manage that part, the solution has been posted and you can use the accompanying code).**

The forward model code:

```

1  function H = H_system(D, d, zed)
2  -     M = 1000;
3  -     N = 1000;
4  -     lambda = 0.5 * 10^-6;
5  -     z = zed * 10^-3;
6
7  -     Delta = D * 10^-6;
8  -     delta = d * 10^-6;
9
10 -     dm = 1/(M * delta)
11 -     dn = 1/(N * Delta)
12
13 -     m = [-M/2:1:(M/2)-1] .* dm;
14 -     n = [-N/2:1:(N/2)-1] .* dn;
15
16 -     [mm, nn] = meshgrid(m, n);
17
18 -     H = exp((-1 * sqrt(-1) * pi * lambda * z) .* (mm.^2 + nn.^2));
19
20 -     P = sinc(Delta * mm) .* sinc(delta * mm)
21
22 -     H_sys = H .* P .* Q;
23
24 -     figure;
25 -     imagesc(real(H_sys));
26
27 -     figure;
28 -     surf(abs(real(H_sys)), 'edgecolor', 'none');
29 - end

```

b) Next, we will consider holographic reconstruction using the same parameters as in (e). What is the condition number of the imaging matrix A?

I was unable to calculate the condition number in time for the project but my guess is that it would be pretty small.

c) Finally, we will investigate the effect of Gaussian noise on reconstruction accuracy. Gaussian noise is commonly encountered in imaging, and is used to model random phenomena due to unmodelled hardware imperfections. The fundamental assumption is that the noise is independent of the measured intensity. The simplest case is the white Gaussian noise (WGN), which assumes the mean as the noise-free image I noisefree, and the same variance n std across all pixels.

Didnt get a chance to finish this one either but the deconvolution result would return a signal denoised since obtaining it using the foward method introduces noise due to the convolution. The least square solution would give a pretty good results and im not sure about the Tikhonov.