Skills Class Signal & Image Processing – Fourier analysis

Pietro Bonizzi & Joël Karel

Fourier analysis of signals

Fourier analysis is one of the most successful techniques in engineering. It underlies a great part of technological development in the last century, especially where mathematical modeling and signal processing is involved.

Question 2.1

Create a sequence `x` sampled at 10 Hz in $[-10, 10]$ by

```
>> x=-10:0.1:10;
```

Calculate the function values $\cos(2\pi x)$ and $\sin(2\pi x)$. Calculate the fast Fourier transform (*fft*) for each of these sequences of function values and plot them.

What happens?

Question 2.2

Examine the Matlab documentation of the *fft* function. Create a function *plotfft* that displays the spectrum of a given fast Fourier transform. Note that the discrete Fourier transform is periodic in frequency. The Matlab *fft* method returns a complex vector with at the left side the zero frequency (sum) and in the middle the highest possible frequency (Nyquist frequency).

The Fourier transform consists of complex numbers. It is not convenient to consider these in order to analyze the frequency content of a signal. Therefore you should consider it in polar coordinates. The squared absolute values of these Fourier coefficients represent the energies of the various harmonic components which constitute the signal. Plot them on screen. Such a plot is similar to what is called a amplitude spectrum. You can look at the documentation of the *fft* function for some help of how to do this (Remark: the discrete Fourier series are defined in a slightly different manner: the coefficients are scaled). The phase spectrum gives information as well: plot the phase and amplitude spectra too. What information?

Also important is to display the amplitude and phase spectra of a signal versus frequencies. This allows an interpretation of the frequency content of a signal consistent with the specific sampling frequency used during the analog/digital conversion of the signal. Given an N-point Fourier transform of a digital signal sampled with a sampling frequency of F_s, think of how you could construct a consistent frequency vector.

Display the two Fourier transform sequences with the *plotfft* function. Explain the appearance of each of these functions in the Fourier domain and discuss their differences.

Question 2.3

Repeat 2.1 and 2.2, but then with a sampling period of 1.1. What happens?

Question 2.4

In this exercise we will be looking at ideal filtering in the Fourier domain. A signal is converted to this domain and its frequency components are analyzed. Then the unwanted frequencies are removed and the signal is converted back to the time domain.

Load the dataset in the file `dataset1.mat`. The data is in the vector `y` (`x` contains the sample times) and the sample frequency is 1000 Hz. Plot the data and calculate its Fourier transform.

The signal in the dataset consists of two harmonics; one frequency of interest and one power line interference. Estimate the frequencies of these harmonics from the amplitude spectrum. Determine a cutoff frequency for ideal high-pass filtering that removes the power line interference. Next, apply this filter to the Fourier domain data. Plot the spectrum of the filtered Fourier-domain data.

Question 2.5

Calculate the inverse Fourier transform of the filtered data with the ifft-function. Plot the original signal, the filtered signal and the difference between the two and check that you have successfully separated the two harmonics. You now have performed filtering in the Fourier domain!

Fourier analysis of images

In MATLAB, the following programs are available for Fourier analysis of images:

fft2	2D Fast Fourier Transform
ifft2	2D Inverse Fast Fourier Transform
fftshift	Swap opposite image quadrants
ifftshift	Undo the effects of the fftshift command
abs	Takes the magnitude of a complex image
angle	Takes the phase of a complex image
real	Takes the real part of a complex image
imag	Takes the imaginary part of a complex image

Question 2.6

Load the image `lena_gray.tif` and convert it to double. Display it by using `imagesc`. Now try to do the Fourier Transform using *fft2*. Try to show the resulting image using `imagesc`:

```
>> im_fft = fft2(im);
>> spectrum = fftshift(abs(im_fft));
>> imagesc(spectrum)
>> colormap gray
```

What happens and why?

Question 2.7

The values of power spectrum of the transformed image have a very dominant peak for low frequencies. Where can this peak be found in the power spectrum, and what is it usually called? This makes images in the frequency domain difficult to view. One way around this is to take the logarithm of the image before viewing it. Remember to use *fftshift* to get the optical form.

Since the Fourier transformed image is a complex image we can visualize both magnitude, phase, real and imaginary part separately. Use `log` to enhance the contrast where it is hard to see (to avoid “log of 0” warnings you can add 1 to transformed image).

```
>> imagesc(fftshift(log(abs(im_fft)+1)))
```

etc...

Comment on the results.

Question 2.8

To illustrate the importance of the phase image we now will try to change the magnitude and phase between two images.

Take a look at the two images, `lena_gray.tif`, and `peppers_gray.tif` using imagesc.

```
>> im1 = double(imread('lena_gray.tif'));
>> im2 = double(imread('peppers_gray.tif'));
```

Perform the Fourier Transform on both images

```
>> im1_fft = fft2(im1);
>> im2_fft = fft2(im2);
```

Split the information into its magnitude and phase parts:

```
>> im1M = abs(im1_fft);
>> im1P = angle(im1_fft); ...etc
```

Combine the two images so that the magnitude of `im1` is combined with the phase of `im2` and vice versa. This can be done by

```
>> im12_ftt = im1M.*(cos(im2P)+1i*sin(im2P));
>> im21_fft = im2M.*(cos(im1P)+1i*sin(im1P));
```

Now perform the Inverse Fourier Transform and look at the real part of the resulting images, as done before. Comment the results. Where is the information about shape located, in the phase or in the magnitude?