Manipulating signals with the Discrete Fourier Transform

DSP Lab 10

Table of contents

1 Objectives **1**

2 Circular and linear convolution with fft() **1**

3 Exercises **2**

4 Final questions **4**

1 Objectives

- Understand the basic principles of the Discrete Fourier Transform (DFT)
- Use the fft() and ifft() functions in MATLAB to perform DFTs and inverse DFTs on signals
- Understand the relationship between the time domain and frequency domain representations of signals

2 Circular and linear convolution with fft()

In MATLAB, both circular and linear convolution can be computed using the fft function, which computes the fast Fourier transform.

To perform circular convolution, the inputs must first be zero-padded to the length of the longest input, and then the fft is applied to both inputs. The resulting Fourier-transformed signals are multiplied element-wise, and then the inverse Fourier transform is applied to obtain the circular convolution of the inputs. This can be done in a single step using the ifft function, which computes the inverse Fourier transform.

```
% Define the inputs x and y
x = [1 2 3 4];
y = [5 6 7 8];

% Zero-pad the inputs to the length of the longest input
n = max(length(x), length(y));
x = [x, zeros(1, n-length(x))];
y = [y, zeros(1, n-length(y))];

% Compute the circular convolution using the fft and ifft
z = ifft(fft(x) .* fft(y));
```

To perform linear convolution, the inputs must first be zero-padded to the length of the sum of the lengths of the inputs minus one, and then the fft is applied to both inputs. The resulting Fourier-transformed signals are multiplied element-wise, and then the inverse Fourier transform is applied to obtain the linear convolution of the inputs. This can also be done in a single step using the ifft function.

```
% Define the inputs x and y
x = [1 2 3 4];
y = [5 6 7 8];

% Zero-pad the inputs to the length of the sum of the lengths of the inputs minus one
n = length(x) + length(y) - 1;
x = [x, zeros(1, n-length(x))];
y = [y, zeros(1, n-length(y))];

% Compute the linear convolution using the fft and ifft
z = ifft(fft(x) .* fft(y));
```

Note that in both cases, the fft and ifft functions can be replaced by the fft2 and ifft2 functions, respectively, if the inputs are two-dimensional arrays. Additionally, the conv function can be used to compute linear convolution directly, without using the fft function.

3 Exercises

1. In this exercise, we will use the fft() and ifft() functions to manipulate a signal in the frequency domain.

a. Create a vector `x` containing the first 16 elements of a square wave with period 8: [1, 1, 1, 1, -1, -1, -1, -1, … repeat …]
b. Compute the DFT of `x` using the fft() function and store the result in a variable `S`.
c. Set the first 5 coefficients of `S` to 0.
d. Compute the inverse DFT of X using the ifft() function and store the result in a variable `y`.
e. Plot the time domain signals `x` and `y` using the `stem()` function, in a single window, using subplot()
f. Explain how the manipulation of the frequency domain representation of the signal affected the time domain signal.

2. Repeat exercise 1, but this time set the last 8 coefficients of `S` to 0.

3. Generate a 39 samples long **triangular** signal `x` defined as:

- first 10 samples are zeros
- next, `x` increases linearly from `x(10) = 0` up to `x(19) = 4`, then decreases linearly to `x(29) = 0`.
- last 10 samples are 0

a. Plot the signal in the top third of a figure, the magnitude of the DFT coefficients in the middle third, and their phase in the lower third.

b. What is the amplitude of the **third harmonic component** in the signal's spectrum?

c. Concatenate 50 zeros at the end of the signal and redo the exercise. What do you observe?

4. Generate two 10-long random signals `x` and `y`.

a. Perform **linear convolution** with `conv()`.

b. Perform **circular convolution** via the frequency domain, using `fft()` and `ifft()`.

c. Perform **linear convolution** via the frequency domain using the `fft` in N points, with N larger then 19.

d. Which method of linear convolution is is faster, `conv()` or via `fft()`? Use long signals (e.g. length 40000).

4 Final questions

1. How do you expect the amplitudes of the Fourier coefficients to be for:

    - a slow varying signal
    - a rapid varying signal