Lab 1: Answers – Introduction to Signal Processing: Fourier Series, Fourier Transform, and Convolution

Part 1: Basic signal operations

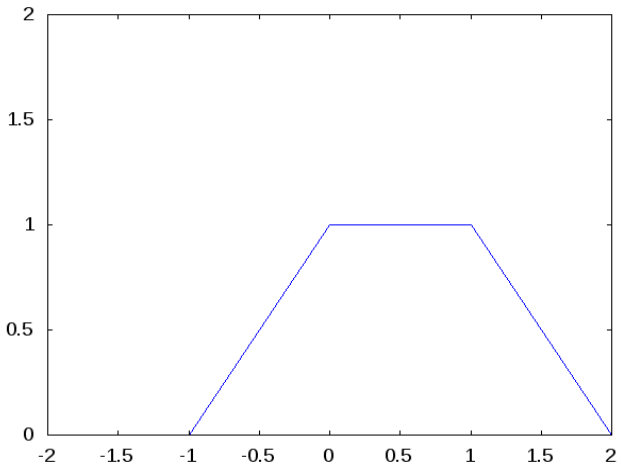
2. **(Q)**: What shape is it?

A: square.

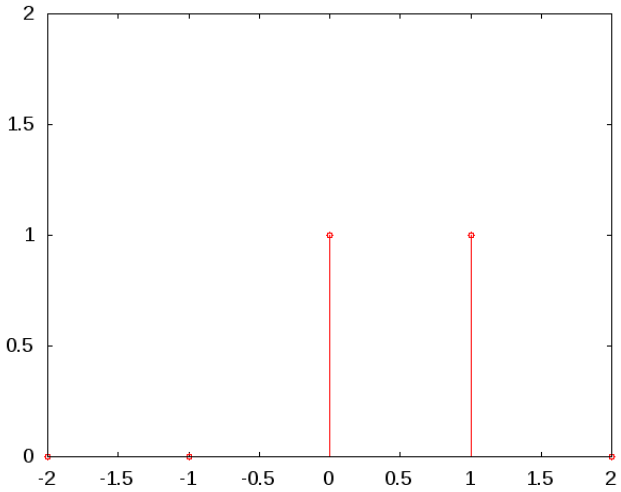
3. **(S)**: In Matlab, a discrete time signal is the same as a vector. Thus, the above signal can be represented as `f = [0 0 1 1 0]` for `x=[-2:2]`. Using Matlab's function `plot(x,y)` and `stem(x,y)`, plot this signal. What difference can you see between those two functions? Which one would you choose to plot discrete time signals? Why?

A:

`plot(x,y)`



`stem(x,y)`



A: `plot()` interpolates using a first-degree polynomial between points, whereas stem() does not. Thus, for discrete time signals, `stem()` is better as it won't introduce interpolation errors (for this example, the square becomes a trapezoid when using `plot()`).

4. **(Q)**: Now consider the signal

$$g(n) = \begin{cases} 1, & if\ n = -1, 0 \\ 0, & otherwise \end{cases}$$

What is the relationship between `f(n)` and `g(n)`? Write `g(n)` as a function of `f(n)` (2 possibilities).

A:

`g(n) = f(-n) (symmetry)`

or

`g(n) = f(n+1) (time shift).`

5. **(Q)**: What would be the resulting signal `h(n)=f(n)+g(n)`?

A: triangle symmetric around `n = 0`.

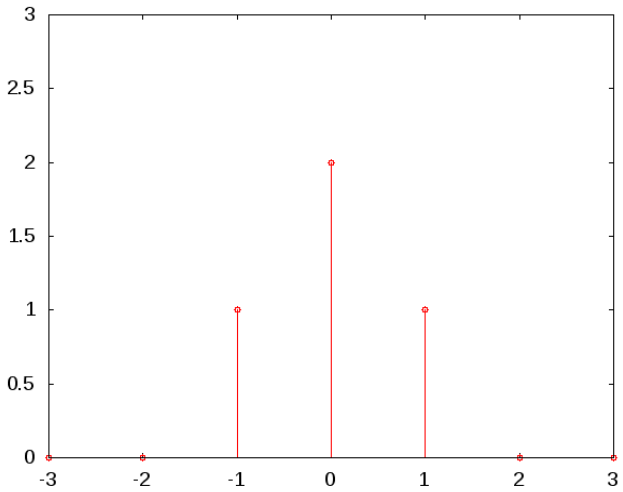
$$h[n] = \begin{cases} 2, & if\ n = 0 \\ 1, & if\ n = -1, 1 \\ 0, & otherwise \end{cases}$$

6. **(S)**: Implement this addition in Matlab. Plot the signal for `x=[-3:3]`.

A:

```
close all;

x = -3:3;
f = [0 0 0 1 1 0 0];
g = [0 0 1 1 0 0 0];
h = f + g;
stem(x,h);
axis ([min(x) max(x) 0 3]);
```

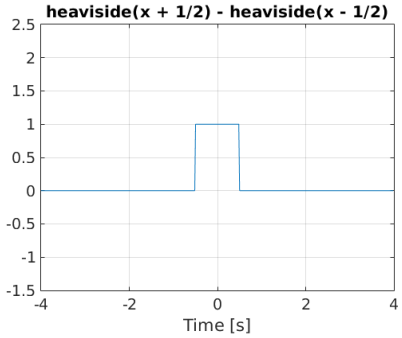


Part 2: Fourier Transform

7. **(S)**: First generate the symbolic variable `x`. For that, you will need to type the command `x = sym('x')`. Symbolic variables allow Matlab to do the Fourier Transform analytically. Using the function `rect(x)` provided with the lab, generate a rectangle function centered in 0 and of width 1. Use `plot_function(f)` to plot the function.

A:

```
syms x;
f = rect(x);
plot_function(f);
```

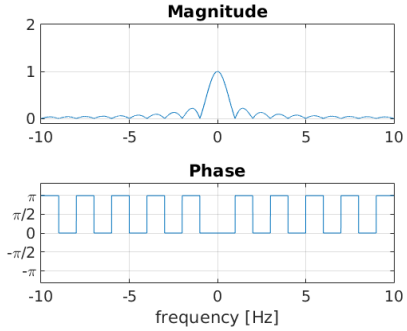


8. **(S)**: Use both `fourier` function and `plot_fourier` function to plot the magnitude of the Fourier Transform of `rect(x)`. What function is it? *Hint: The magnitude is always positive.*

A:

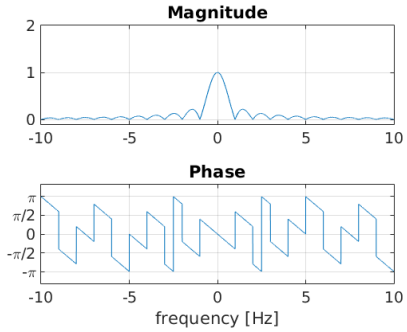
```
plot_fourier(fourier(f));
```

It is the absolute value of a sinc because magnitude is always positive. The phase is changing to reflect the fact that in reality the Fourier transform has negative values.



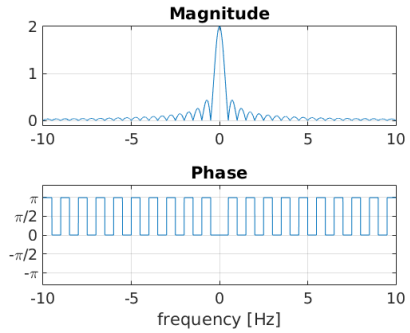
9. **(S)**: Do the same for a rectangle centered around 0.2. What difference do you notice?

A: The magnitude of the Fourier transform of `rect(x-0.2)` does not change, only the phase is affected. The phase increases linearly with the frequency due to the time-shifting property of Fourier transforms $x(t - t_0) \leftrightarrow e^{-j\omega t_0}X(\omega)$*. The discontinuities are due to both the magnitude being always positive and the phase being 2π periodic.*



10. **(B)**: Do the same for a rectangle with width 2, centered in 0. What happens? Try with other sizes.

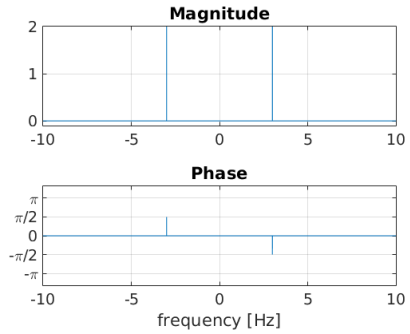
A: The function to implement is `rect(0.5*x)`. The Fourier transform is a sinc. The sinc function is two times tighter and higher. The changes in phase are also narrower.



11. **(S)**: Plot the Fourier Transform of a sine using the Matlab function `sin(a*x)` with pulsation of `a = 6*pi rad/s`. Describe the obtained function.

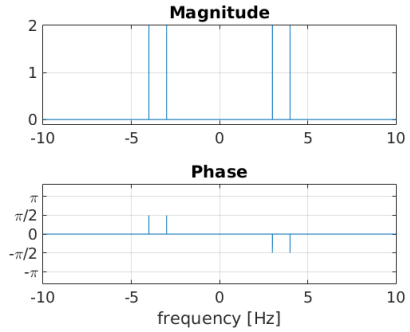
A: `plot_fourier(fourier(sin(6*pi*x)));`

Two diracs symmetric around 0.



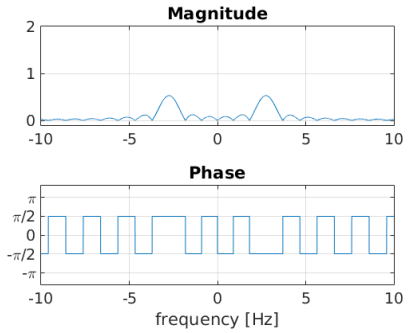
12. **(S)**: Plot the Fourier Transform of the sum of two sines at pulsations of f_1 = `6*pi rad/s` and f_2 = `8*pi rad/s`. What is the result?

A: The Fourier transform of `sin(6*pi*x)+sin(8*pi*x)` *is four diracs at the two corresponding frequencies: 3 Hz and 4 Hz. This is because Fourier transforms is a linear transformation.*



13. **(B)**: Plot the Fourier Transform of `rect(x)*sin(6*pi*x)`. Do you recognize the obtained magnitude plot in previous questions?

A: Two sinc functions 8**(S)** *at the near frequency as the diracs in questions* 12**(S)**. *Note that a multiplication in the time domain is a convolution in the frequency domain.*

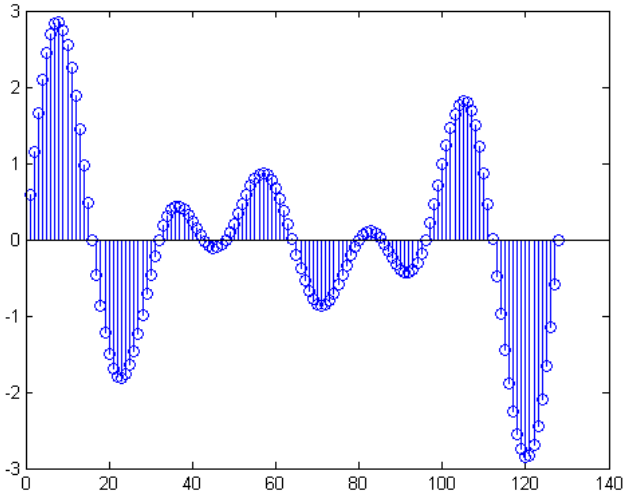


Part 3: Fourier Series, Discrete Fourier Transform and FFT

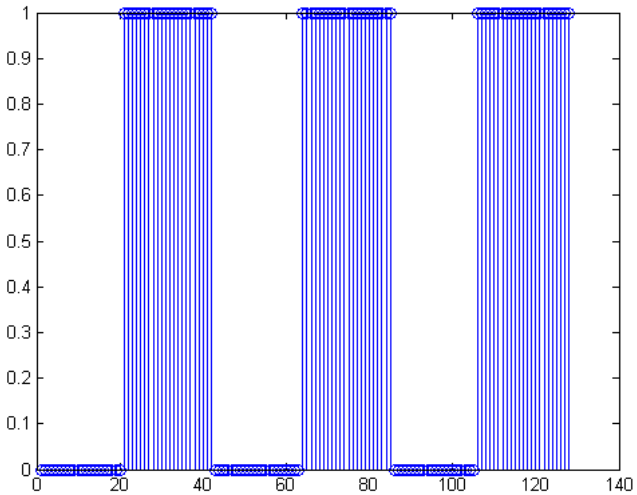
14. **(S)**: Go to the folder `part3` and load the 3 signals `f2[n]`, `g2[n]` and `l[n]` using `load('Lab01Signals.mat')`. Plot `f2[n]`, `g2[n]`, using `stem()`.

A:

`stem(f2)`



`stem(g2)`

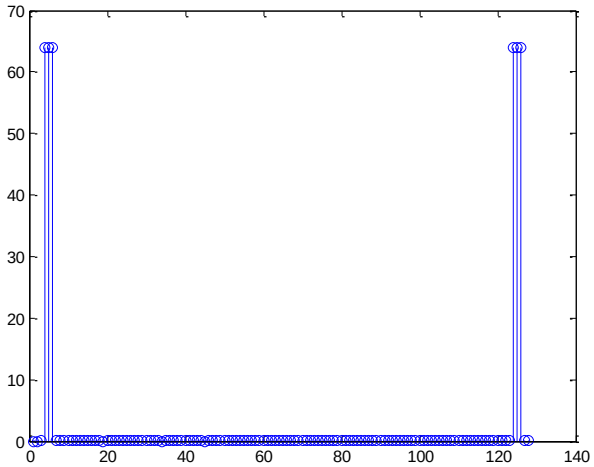


15. **(S)**: The function `fourier_compute.m` implements the Fourier analysis. Compute the Fourier coefficients `X[k]` for `f2[n]` using `fourier_compute.m` and plot their

magnitude using `stem()`. Do you see symmetry? *Hint: Remember that the Fourier coefficients are complex. You can plot the magnitude using the function* `abs()`.

A:

```
stem(abs(fourier_compute(f2)));
```



16. **(S)**: Now synthesize the signal `f2[n]` using `fourier_revert()` and plot it. Do you get the original signal `f2[n]` back? *Hint: a good way to quickly check how similar two signals are is to simply plot their difference.*

A:

```
F2 = fourier_compute(f2);
stem(abs(f2-fourier_revert(F2)));
```

Yes, except for very small truncating differences due to the finite accuracy of number representation in a computer.

17. **(S)**: Now set all Fourier coefficients `X[k]` to 0 except for `k= 4,5,6,124,125,126`. Resynthesize `f2[n]` and plot it. Is the signal still the same as the one you got in 16 **(S)**?

A: Yes, except for very small truncating differences due to the finite accuracy of number representation in a computer.

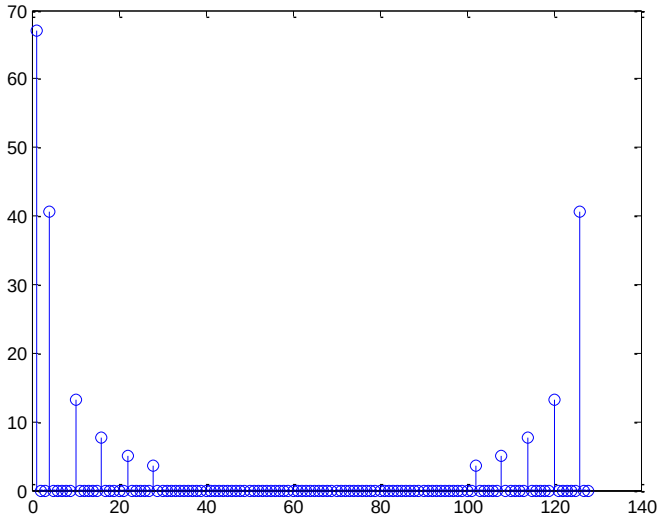
18. **(Q)**: You just set almost all Fourier coefficients `X[k]` to 0 yet the resynthesized signal still looks very much the same as the one resynthesized from only non-zero coefficients. Why?

A: All Fourier coefficients except for 4,5,6,124,125 and 126 are almost 0 and the signal f is the sum of only a finite number of sines/cosines represented by a finite number of Fourier coefficients.

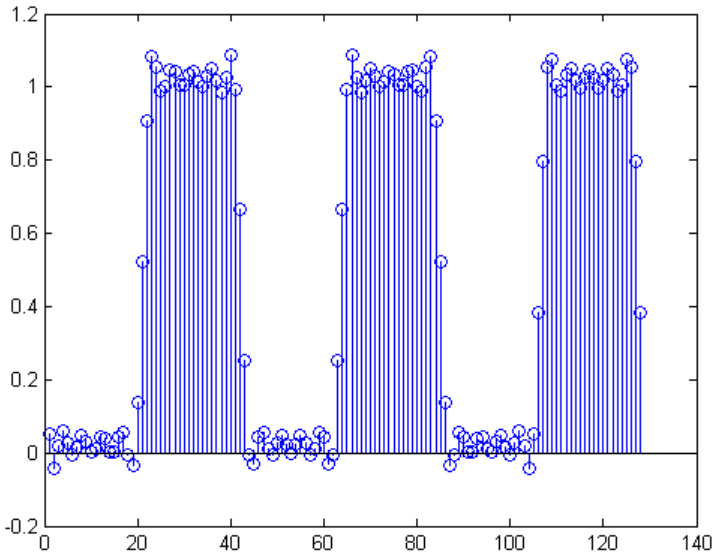
19. **(S)**: Determine the Fourier coefficients `X[k]` for `g2[n]`. Eliminate all `X[k]` for which `|X[k]|<3`, resynthesize `g2[n]` and plot it. Is it the same as the original signal?

A: Fourier coefficients with `|X[k]|>=3, other X[k] = 0.`

```
X = fourier_compute(g2);
X(abs(X)<3) = 0;
stem(fourier_revert(X))
```



The resynthesized signal which is similar but not quite the same as `g2[n]`.



20. **(Q)**: Using just 3 Fourier coefficients `k=4,5,6` (and their symmetric values `k=126,125,124`) was enough to synthesize the signal `f2[n]`. Why do we need many more coefficients (all?) to properly resynthesize g2[`n]`?

A: The Fourier transform of a rectangular signal such as `g2[n]` *has infinitely many non-zero coefficients (while* `f2[n]` *only has 6) which are all required when the original signal is to be properly resynthesized.*

21. **(S)**: `time_fourier_compute.m` computes the *Fourier* coefficients of `l[n]`, a random signal with 10000 values, using `fourier_compute.m`. The script `time_fft.m` does the same, but uses Matlab's built-in `fft()`. Both programs indicate the time it took them to complete. Execute both and compare the computation times.

A: Execution time depends on hardware. In laptop with Intel Core i5 processor `time_fourier_analyze` *takes roughly 11 seconds and* `time_fft` *takes 6 milliseconds.*

22. **(B)**: Using `fft()`, revisit 18 **(I)**. Compute the *Fourier* coefficients for `f[n]` using `fft()` and compare them to the ones computed using `fourier_analyze.m`

A: They are identical, except for very small truncating differences due to the finite accuracy of number representation in a computer.

Part 4: Continuous Convolution

23. **(Q):** In the picture above, a plot of function f(t) is shown. Find a mathematical expression for f(t) using only a combination of the rectangle shape functions you have already used in Part 2.

A:

$$f(t) = rect(0.5 + t) + 2 * rect(-0.5 + t)$$
Or
$$f(t) = rect(0.5 * t) + rect(-0.5 + t)$$

24. **(I):** Using the function `rect(a, b)` provided with the lab, generate function f(t) in Matlab. It is possible to add or subtract rectangle functions together. The function `rect(a, b)` provides a rectangle function with its middle at `a` and of width `b`. Use `plot_function(f)` to plot the function.

A:

```
f = rect2(-0.5, 1) + 2*rect2(0.5, 1)
```
Or
```
f = rect2(0, 2) + rect2(0.5, 1)
```

25. **(Q):** Convolution (French: "convolution", German: "Faltung") is an operation which is extensively used in signal processing, particularly in system analysis and filtering. Write an analytic expression for f(x) as a convolution of one rectangle function with a sum of dirac delta functions.

A:

$$f(t) = rect(t) * (\delta(t + 0.5) + 2\delta(t - 0.5))$$

26. **(S):** Do the convolution of Question **30** in Matlab using the function `h = convolution(f, g)` where f and g are the functions you want to convolve. The dirac function is provided in Matlab with `ddirac(t)` where t is the position of the dirac.

A:

```
h = convolution(rect2(0, 1), ddirac(-0.5)+2*ddirac(0.5))
```

27. **(S):** Do the convolution of h with itself several times: `h = convolution(h,h)`, and observe how the signal changes. Do you think the signal is converging to anything?

A:

It becomes a Gaussian after 5 to 8 times. If convolved more, the function degenerates as there are some border effects (the signal is finite in time).

Part 5: Discrete Convolution

28. **(S):** As you already learned in Part 1, in Matlab, a discrete time signal is the same as a vector. Thus, the signal w[n] can be represented as `w = [0 0 0 1 1 0 0]` for `n=[-3;3]`. Define signal v[n] in a similar way. Using `stem(n,w)` and `stem(n,v)` plot these signals to double check you defined them properly.

A:

```
n = -3:3;
w = [0 0 0 1 1 0 0];
v = [0 0 1 1 0 0 0];
stem(n,w);
axis ([min(n) max(n) 0 2]);

figure
stem(n,v);
axis ([min(n) max(n) 0 3]);
```

29. **(Q):** On a piece of paper, compute `l[n]=(w*v)[n]`. Hint: consider the formula for `n = [-2, 2]`.

A:

$$n = -2 \rightarrow \sum_m w[m]v[-2 - m] = \{for\ m = 0, 1\}\ 0 * 1 + 0 * 1 = 0$$

$$n = -1 \rightarrow \sum_m w[m]v[-1-m] = \{for\ m = 0,1\}\ 1*1+0*1 = 1$$
$$n = 0 \rightarrow \sum_m w[m]v[0-m] = \{for\ m = 0,1\}\ 1*1+1*1 = 2$$
$$n = 1 \rightarrow \sum_m w[m]v[1-m] = \{for\ m = 0,1\}\ 0*1+1*1 = 1$$
$$n = 2 \rightarrow \sum_m w[m]v[2-m] = \{for\ m = 0,1\}\ 0*1+0*1 = 0$$

30. **(S):** Use the Matlab function `l = conv(w, v)` that does a discrete convolution between w and v. Does it show the same result? Try also the function with the following parameter: `l = conv(w, v, 'same')`. Use the Matlab help to understand the differences.

A:
You will notice that the shape is the same but that it is shifted. This is because the vector size of the signal has increased. This is because of the way convolution is calculated in matlab (increasing both signals by adding zeros at the borders). It is possible to get a convolution of the same size by providing the option 'same' in the conv function, matlab will return a vector of same size as the first input.

31. **(I):** Write the code for performing the convolution by yourself (i.e, without using the matlab function `conv`) for `rect` function.

A: *The MATLAB code is shown below:*

```
for n = 1:13
    result(n) = 0;
    for m = 1:7
         if (n-m)>= 1 && n-m <=7
         result(n) =  result(n) + rect_1(m) * rect_2(n-m);
         end
    end
end
```

where `rect_1`, `rect_2` are two copies `rect` function.