# Lab 11 : Spectral Methods

Jeroen Olieslagers
Richard-John Lin

Center for Data Science

12/06/2022

## Sampling function

- Make sure you are not returning zeros!
- Double check your matrix shapes, it's easy to mess up a transpose!
- Make sure you are using the Cholensky decomposition and $\mathbf{x} = \mu + L\mathbf{z}$
- Overall very high grades for this lab

## Principle

- A periodic signal can be decomposed as a superposition of periodic functions
- More generally, the Fourier basis $\{\cos(\omega t), \sin(\omega t)\}_{\omega \in \mathbb{R}_+}$ / $\{\exp(i\omega t)\}_{\omega \in \mathbb{R}}$ is an orthogonal basis of the $L^1$ functions (can be extended to $L^2$)

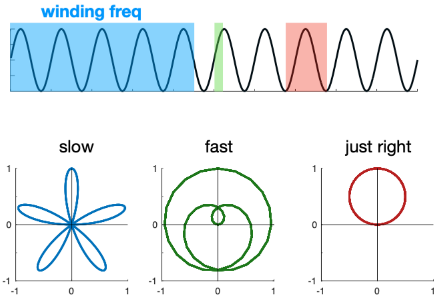



Figure 1: Fourier transform illustration (3Blue1Brown)

## Formulation

- **Discrete Fourier Transform**
  Let a series of points $x_{0:N-1}$. Then,

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-\frac{i2\pi}{N}kn}.$$

- **Continuous Fourier Transform**
  Let $f(t) \in L^2$. Then,

$$\hat{f}(\omega) = \int_{\mathbb{R}} f(t) e^{-i\omega t}\,\mathrm{d}t$$

## A note on conventions

- The conventions for the Fourier Transform sometimes differ according to some fields.
- This usually changes the multiplicative term. In particular, we need to check the multiplicative constant in convolution property.
- For example, sometimes in Physics :

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} f(t) e^{-i\omega t}.$$

This is practical to inverse the transformation but not for the convolution product.

## A few properties

| | "Temporal" | Frequential |
|---|---|---|
| Linearity | $af(x) + bg(x)$ | $a\hat{f}(\omega) + b\hat{g}(\omega)$ |
| Domain contraction | $f(ax)$ | $\frac{1}{\lvert a\rvert}\hat{f}(\frac{\omega}{a})$ |
| Convolution | $(f * g)(x)$ | $\hat{f}(\omega)\hat{g}(\omega)$ |
| Product | $(fg)(x)$ | $\frac{1}{2\pi}(\hat{f} * \hat{g})(\omega)$ |
| Temporal derivative | $f'(x)$ | $i\omega\hat{f}(\omega)$ |
| Frequential derivative | $xf(x)$ | $if'(\omega)$ |

- Also the FT of a Gaussian is Gaussian

## Nyquist / Shannon frequency

- As we are dealing with discrete measurements in real life, not all the frequencies can be observed.
- Play here

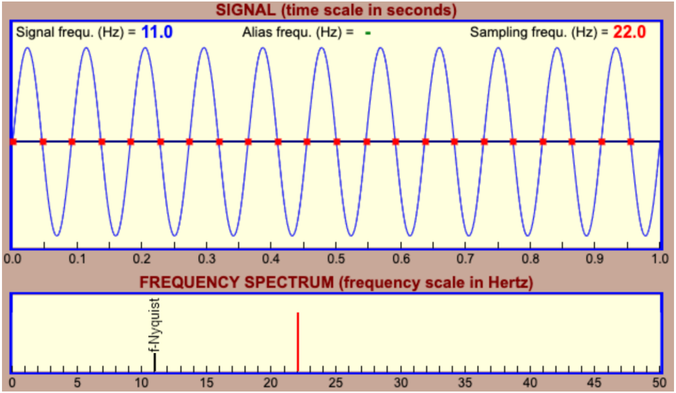



Figure 2: At Nyquist frequency

## Nyquist / Shannon frequency

- As we are dealing with discrete measurements in real life, not all the frequencies can be observed.
- Play here

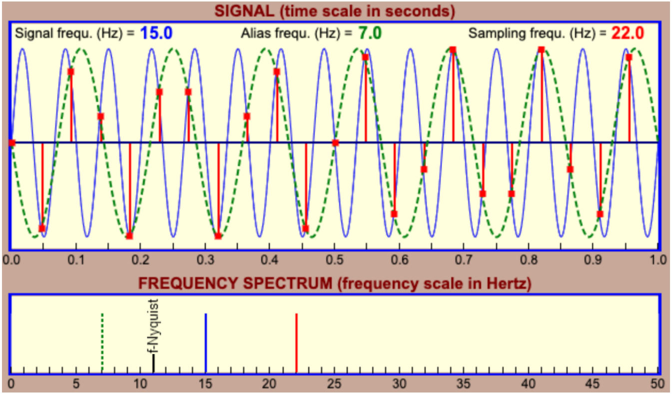



Figure 2: Higher than Nyquist frequency

## Border effects

- Although the Fourier basis is a basis, as the real series are discrete, the fit is not necessary perfect

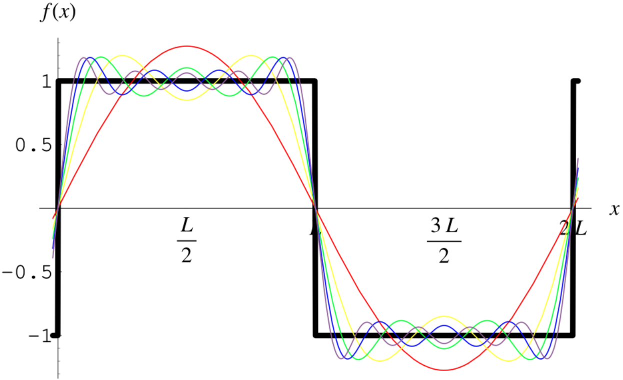



Figure 3: Border effects

# Application of Fourier Transform : Heisenberg Uncertainty principle

## Periodogram

- The periodogram is an estimate of the energy spectral density of a signal

$$\mathcal{F}\left(x(t) * x^*(-t)\right) = \hat{x}(\omega)\hat{x^*}() = |\hat{x}(\omega)|^2$$

- Proportional to the norm of the FT coefficients

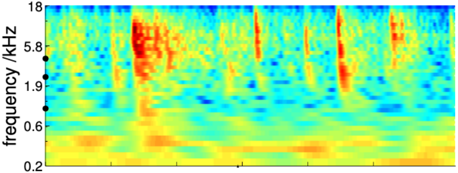



Figure 4: Spectrogram

# Stationary and non-stationary data

Below is the function for generating the datasets. The first dataset is stationary, and is given by:

$y_s(t) = \sum_{\{\omega\}} \sin(\omega t) + \eta(t),$

where $\eta(t) \sim \mathcal{N}(0, \sigma^2)$.

The second dataset is nonstationary, and is given by:

$y_{ns}(t) = m_1(t) \sum_{\{\omega_1\}} \sin(\omega_1 t) + m_2(t) \sum_{\{\omega_2\}} \sin(\omega_2 t) + \eta(t),$

where again $\eta(t) \sim \mathcal{N}(0, \sigma^2)$, $m_1(t)$ and $m_2(t)$ are positive nonstationary processes, and the sets $\{\omega_1\}$ and $\{\omega_2\}$ define two separate sets of frequencies, each associated with a separate modulator.

We will be exploring the relationship between each process and its power spectrum.

## Part I: Power Spectrum

### Computing the Power Spectrum

The power spectrum of a signal is given by:

$$S(f) = |a(f)|^2,$$

where $a(f)$ is the Fourier coefficient for frequency $f$, computed using the Fourier transform.

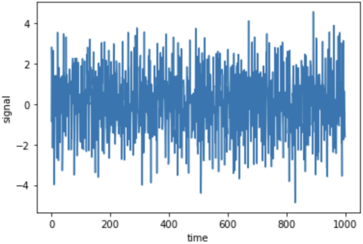
In the following section, you will compute the power spectrum using the Fast Fourier Transform (try numpy.fft.fft).

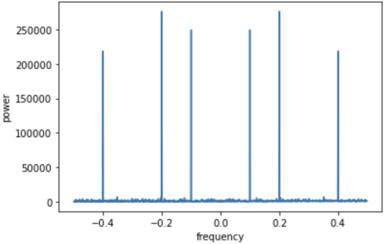
## Part I: Power Spectrum

Don't forget about second TODO!

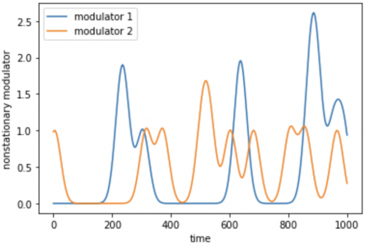
```
#plot the stationary process
sigma = 1
freq = np.array([0.1, 0.2, 0.4])
tstep = 1000
data_stationary = gen_stationary_dataset(sigma, freq, tstep)

plt.figure()
plt.plot(np.arange(0, tstep), data_stationary)
plt.xlabel('time')
plt.ylabel('signal')

###TODO: use the numpy fft function to plot the power spectrum of the data###


###TODO: re-plot the data for increased values of sigma.
```

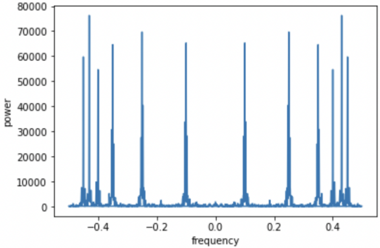
## Part I: Power Spectrum for stationary data



## Part I: Power Spectrum for non-stationary data



# Part II: Spectrogram

## Compute Power Spectrum in each time window

### Computing a spectrogram

Here, we will adapt the power spectrum to gather spectral information over shorter periods of time. This will allow us to analyze nonstationarities in our signal.

```
def power_spectrum_window(data, window_size):
    ### TODO: write a function that returns a (len(data)-window_size) x window_size matrix
    ### where the ith column gives the power spectrum of the data for the data from i-window_size/2:i+window_size

    pspectrum = np.zeros((len(data)-window_size, window_size))

    return pspectrum
```

# Part II: Spectrogram