Time Series TSIA202a – Practical works

The goal of this practical work is to experiment the estimation of second order moments for random processes, and use them to estimate the spectral density. Moreover, these tools will be applied to the analysis and synthesis of speech signal, modeled as an AR process of suitable order.

The exercises are to be developed in Python. Some scripts are provided, with partially implemented methods and examples, that you have to complete. **The provided scripts are guaranteed to work with Python 3.7**. Moreover, some hints are given for the "Spyder" developping environment.

You may use your own PC and / or different environments (such as jupyter) or versions but in this case some adjustment may be needed to make them run.

The simplest is to use spyder, as described in the following.

Start by:

- Create a working directory
- Go to the "eCampus" website
- Download the zip file with the Python Scripts and unzip it in your working directory
- Download the `wav` file with speech sample
- From the terminal, launch `spyder`
- In the editor, open the scripts `covariance.py`, `estimation.py` and `speech_proc.py`
- To open up separate windows for interactive figures in Spyder, go to Spyder menu and set: Tools → Preferences → Ipython Console → Graphics → Graphics Backend → Backend: "automatic"

Exercise 1: Computing averages and covariances

Run the script `covariance.py` such that the averages and the covariances are shown. They have to be measured using a large number of realizations of the following processes:

1. White noise Z_t with variance σ^2 (use `np.random.normal`).
2. A causal AR(1) process with a coefficient $\phi \in]-1, 1[$
3. A sinusoidal process $X_t = A_0 \cos(\lambda_0 t + \Phi_0) + Z_t$, where $\lambda \in [0, \pi[$, and Φ_0 is a uniform random variable in$[0, 2\pi]$, independent from Z_t.

The experimental result must be compared with the theoretical ones: what are the ACF functions of these three models?

Use the function `drawZ_DTFT_AR` from the `randproc` module to draw the poles of the transfer function of the AR model, its squared absolute value, and the sample PSD.

Exercise 2: Spectral density estimation and periodogram

We try to estimate the power spectral density (PSD) of a real, zero-mean, weakly stationary process $\{X_t, t \in \mathbb{Z}\}$. We have access to observations in $t \in \{0, 1, \ldots, n-1\}$, and we want to use the FFT algorithm that implements the DFT, using the `numpy.fft` module.

We recall that the DSP $f_X(\lambda)$ is the discrete time Fourier transform (DTFT) of the autocovariance function $\gamma_X(h)$:

$$f_X(\lambda) = \frac{1}{2\pi} \sum_{h \in \mathbb{Z}} \gamma(h) e^{-i\lambda h}$$

.

For this exercise, you will have to complete the script `estimation.py`

1. Let $\widehat{\gamma}_n(h)$ be the empirical estimator of the acf of X_t given the observations for $t \in \{0, 1, \ldots, n-1\}$:

$$\widehat{\gamma}_n(h) = \begin{cases} \frac{1}{n} \sum_{t=0}^{n-1-h} (X_t - \hat{\mu}_n)(X_{t+h} - \hat{\mu}_n) & \text{if } h < n \\ 0 & \text{if } h \geq n \end{cases}$$

where $\hat{\mu}_n = \frac{1}{n} \sum_{t=0}^{n-1} X_t$, while $\forall h < 0, \widehat{\gamma}_n(h) = \widehat{\gamma}_n(|h|)$, since the process is real.
We define the *periodogram* of $X_0, X_1, \ldots, X_{n-1}$ as $I_n(\lambda)$:

$$I_n(\lambda) = \frac{1}{2\pi} \sum_{h=-n+1}^{n-1} \widehat{\gamma}_n(h) e^{-i\lambda h}.$$

Express $I_n(\lambda)$ as a function of the DTFT of $X_0 - \hat{\mu}_n, X_1 - \hat{\mu}_n, \ldots, X_{n-1} - \hat{\mu}_n$.

2. Deduce an algorithm for computing $I_n\left(2\pi \frac{k}{m}\right)$ for $k \in \{0, 1, \ldots, m-1\}$, where $m \geq n$ is the order of the DFT used. Test this with the processes defined in Exercise 1.
3. How to obtain $\widehat{\gamma}_n(h)$ from $I_n\left(2\pi \frac{k}{m}\right)$? Which value of m should be used? Verify that the function `acovb(X)` in the module `randproc` correctly computes the estimate of the autocovariance function from a set of samples.
4. In the case of white noise, estimate the variance of the periodogram for several values of n. Draw conclusions.

Exercise 3: Filtering random processes

Part I — Yule-Walker equations

Let us consider a causal, zero-mean, AR(p) process defined by the following recurrent equation:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + Z_t$$

where $\{Z_t, t \in \mathbb{Z}\}$ is a weak white noise with variance σ^2.

1. Show that, $\forall h \geq 1, \mathbb{E}\left[X_{t-h} Z_t\right] = 0$.
2. Deduce a recurrent relation between $\gamma(h)$ and $\gamma(h-1), \gamma(h-2), \ldots, \gamma(h-p)$ for $h \geq 1$.
3. We consider separately the case $h = 0$: find a new relationship between $\gamma(0)$ and $\gamma(-1), \gamma(-2), \ldots, \gamma(-p)$

4. Put these relationship in matricial form:

$$\Gamma_{p+1} \begin{bmatrix} 1 \\ -\phi_1 \\ \vdots \\ -\phi_p \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{1}$$

where Γ_{p+1} is a suitable Toeplitz matrix[1] that you have to determine.

Part II — Estimation

1. Complete the script `estimation.py`. Start by generating `n=1000` samples of an AR(4) process using the function `genAR` in the module `randproc.py`
2. Use the function `acovb` to produce an estimation $\hat{\Gamma}_{n,p+1}$ of Γ_{p+1}.
3. Solve the equation system (1). Start by computing the intermediate vector:

$$v = \hat{\Gamma}^{-1}_{n,p+1} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

then estimate the variance and the AR coefficients. Compute the relative error, show the estimated PSD and compare with the frequencies associated to the poles of $\frac{1}{\Phi(z^{-1})}$.

Part III — Application to speech signal

In this part, we explore the analysis/synthesis represention of a speech signal. In order to listen to the speech sample, use any player on the provided and the synthesized `.wav` files.

The main elements of the analysis-synthesis processing of a speech signal are implemented in the `speech_proc.py` script. You will have to complete the missing parts.

In the **analysis** part, the signal is divided into *frames*, with a given overlap. The typical frame duration is around 30 ms, while the overlap is often chosen at 50 %. Each frame is considered as an AR(12) process, whose coefficients can be estimated as shown before. If the frame presents an harmonic spectrum (because of the vocal cords vibration), it is said to be a *voiced* frame, and the foundamental period `T` can be estimated by using the function `detectPitch`. If the frame is unvoiced, the estimated period is zero. The estimated parameters $\hat{\phi}, \hat{T}, \hat{\gamma}(0)$ are then sent to the synthesis part.

In the **synthesis** part, the signal is reconstructed as AR filtering of a pulse train of period T (for voiced frames) or of white noise (for unvoiced frames). Moreover, the synthesis must take into account the overlap, using a Hann window.

1. The function `detectPitch(frame, minT, maxT)` detects the existence of a fundamental period in a frame, by looking for a period $T \in [\mathrm{minT}, \mathrm{maxT}]$. It returns zero if the trame is unvoiced. Knowing that the fundaments frequency $f_0 = \frac{1}{T}$ of the human voice is typically in the interval $80 - 400$ Hz, and knowing the sampling frequency F_s of the signal, how should we choose minT and maxT?

[1] A Toeplitz matrix is characterized by constant diagonals.

2. Complete the estimation of the pitch and of the signal parameters (AR coefficients, variance). Compare graphically the behavior of the AR model and the PSD of the signal.
3. Perform the analysis and synthesis of the provided speech sample. The synthesized signal is saved into a `.wav` file. Listen to the original and synthesized signal by using any audio player. Try to modify the pitch of the signal and listen to the result.