Signal Analysis: Homework 3

This is the third out of three homework sheets for the 2019 Signal Analysis problem class. Each homework sheet includes three different examples resulting in a total number of nine homework examples over progress of the course. The presented homework examples are mostly MATLAB examples with a strong real world focus.

The grading of the problem class is based on student presentations (Power Point or Latex presentation) and exercise interviews for five out of the nine examples. For each of the three homework sheets, students have to present one example, which will be selected by the lecturers ("must show"). Students will be informed three days prior to the deadline of the problem sheet, about the number of the example, which has to be presented. The remaining two examples can be selected by the students itself ("will show"). Hereby only one of the remaining two examples per problem sheet can be selected. Note, that the grading is an individual grading.

All presentations and necessary MATLAB-files have to be provided by the deadline to sa@emt.tugraz.at. This holds for all presentations ("must show" and selected "will show"). Please follow the instructions for the upload, which are provided by an additional sheet on the server.

The presentation slides have to consider all necessary derivations, algorithmic considerations, diagrams, conclusions, etc.. A pure presentation of the final results will lead to reduction of your points. Also take care to the representation of necessary diagrams. Slides can be in German of English. Note, that your slides have to include all graphics. A reference to a generating MATLAB-script is not accepted.

Deadline: 6.3.2020

Example 1: PSD Estimation

This example investigates parametric and non-parametric methods to estimate the power spectral density (PSD) of stochastic processes. For your work, you can use standard MATLAB commands for PSD estimation, as well as reference implementations provided by the authors of the book 'Spectral Analysis of Signals'. An electronic version of this book is available on the server.
You are provided by the MATLAB function

```
xn=signalgenerator1(id,Task)
```

which generates the test signals for your analysis tasks. The function is used for the subtasks 1 and 2. It requires the following inputs:

- `id`: your student ID (integer number). You can find your student ID on the server.

- `Task`: (1 to 2)

Depending on the variable `Task` you are provided with different signals, where you have to apply parametric or non-parametric PSD analysis.

The following subsections provide you with details about the individual tasks. Please provide answers for the listed tasks.

Task 1

Run the signal generator with `Task=1`. You are provided with 10^6 samples of a stationary SP. The sampling frequency of the signal is 1 MHz.

- Estimate the PSD by means of a nonparametric approach. Provide information about the frequencies/bandwidths of the signal components.
- Evaluate the total signal power by means of the second moment, as well as by means of integration of the PSD.
- Estimate the PSD by means of a parametric approach. You can use MATLABs `aryule`. Compute the total signal power of the AR-process (Hint: use `freqz` to evaluate the frequency response of the AR process). Discuss the result. What modifications are required to model the PSD by an AR-process?

Task 2

In this task we consider the estimation of the frequency of a sinusoidal signal using an AR(2) process. Run the signal generator with `Task=2`. The signal is a sinusoidal signal with 10000 samples. The sampling frequency of the signal is 1 MHz.

The transfer function of an AR(2) process is given by

$$H(z) = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}}. \tag{1}$$

- Create an AR(2) process with complex conjugate poles and simulate the signal. E.g. use the poles $z_{1,2} = re^{\pm j\omega_0}$, where $|r|$ is close to one (but within the unit circle) and $0 < \omega_0 < \pi$ holds. What kind of signal is generated from the AR process?
- Generate a sinusoidal signal with a frequency of some 10 kHz. Use a sampling frequency of 1 MHz, the number of samples should be 10000. Estimate the coefficients of an AR(2) for the signal. You can then determine the frequency of the signal from the angular position of the roots of the denominator polynomial. What can you say about the quality of the frequency estimation of this approach? Compare the result to a non-parametric approach, e.g. the DFT-based measurement system.

- Run the signal generator with `Task=2`. Estimate the signal frequency of the signal provided by the signal generator.

Task 3

Consider an AR-process with the transfer function

$$H(z) = \frac{1}{(z - 0.55)(z + 0.25)(z - j0.85)(z + j0.85)} \tag{2}$$

the input signal is i.i.d. Gaussian noise with a variance of $\sigma^2 = 1$. In this example we want to compare the parametric PSD estimator using the least squares approach, against the estimator based on the Yule-Walker equations. For the analysis we study the estimation error $e_i = \hat{a}_i - a_i$ for the coefficients a_i of the transfer function $H(z)$. $\hat{a}_i$ denotes the estimated coefficient. Use a Monte Carlo analysis and study the mean μ_{e_i}, the variance σ_{e_i} and the means square error $\mathcal{E}(e_i^2)$ for the estimation of the coefficients. You can set the system order to be the correct order of $H(z)$. Perform the analysis for 100, 500 and 1000 data points in the input vector for the PSD estimator. Provide the results by means of a table or suitable plots.

Monte Carlo Analysis: a Monte Carlo Analysis is a computer simulation, which maintains random experiments to derive a statistical characterization of a process. In a single Monte Carlo simulation you have to perform the following steps:

- Simulate the AR process to generate the data. Make sure that your data is free from filter transients. With this step you introduce the randomness in the Monte carlo analysis
- Estimate the parameters with the least squares estimator.
- Estimate the parameters with the estimator, which is based on the Yule-Walker equations. Note, that estimation of the ACF in the estimator, can be done by means of the biased or the unbiased estimator. Consider both variants in your analysis.

Make sure that the number of Monte-Carlo simulations is sufficiently high. These simulations might take longer, e.g. in our tests we used > 10000 Monte Carlo simulations!

Example 2: STFT Analysis

This example deals with the coding and decoding of encrypted text messages. The coding scheme is based on a sequence of different tones. Hence, STFT analysis is a proper tool to analyze the signal.

You are provided by the MATLAB function

```
xn=signalgenerator2(id)
```

which provides you with a personalized signal to analyze. Note, that the sampling frequency in this example is set to $f_S = 8192\,\text{Hz}$.

The coding of a symbol is based on the following scheme. A symbol contains a block of three characters out of the character set 'a' to 'z' and ' ' (blank). Each symbol is coded in the following manner:

- First character: coded by means of the frequency of the tone in Hertz.
- Second character: coded by means of the duration of the tone in samples.
- Third character: coded by means of the duration of the succeeding signal pause in samples.

After a pause representing the third character of the symbol, the next symbol starts with the first character coded by means of a sine tone with a certain frequency. The coding scheme for the symbol 'abc' is illustrated in Figure 1. Note, that the signal provided by the function `signalgenerator2` starts and ends with 1000 samples, which have the value of zero. Also the signal is corrupted by additive white Gaussian noise.
For the coding of the first character, the chosen frequencies equal the tone pitches of the scale of C major ('C-Dur Tonleiter'). The lowest frequency (character 'a') corresponds to the tone A3. The complete coding table is depicted in Figure 2.

Again note, that the sampling frequency of the coded signals is set to $f_S = 8192\,\text{Hz}$.

Please provide answers to the following points:

1. Given the coding table in Figure 2 you should decode the encrypted messages provided by

   ```
   xn=signalgenerator2(id)
   ```

   to obtain a message based on your student ID `id`. Decode the message using the STFT (MATLAB function spectrogram).

2. Which bandwidth is occupied by the signal? Determine the upper and lower frequency, for which the spectrogram is smaller than $-20\,\text{dB}$ of the main signal. Provide a proper depiction of the spectrogram to verify your result.

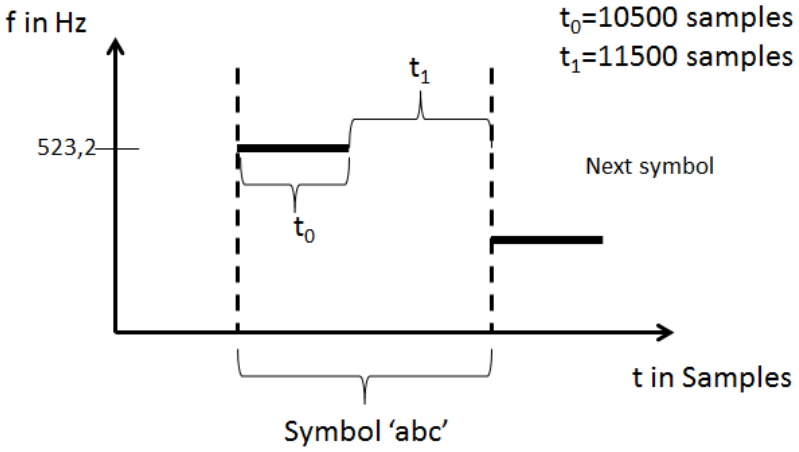

Figure 1: Coding scheme for symbol 'abc'. The character 'a' is coded as tone with a frequency of 523.3 Hz, the tone length of 105000 samples codes character 'b' and the following pause of 11500 samples codes character 'c'. The delays are a result of the sampling frequency, which is set to $f_S = 8192$ Hz.

Letter	Encryption 1st Letter: Frequency [Hz]	Encryption 2nd / 3rd Letter: Length / Pause [Samples]	Letter	Encryption 1st Letter: Frequency [Hz]	Encryption 2nd / 3rd Letter: Length / Pause [Samples]
'a'	523.3	5000	'o'		
'b'	1568.0	10500	'p'		
'c'	1975.5	11500	'q'		
'd'	2793.8	13500	'r'	2093.0	12000
'e'	1174.7	9000	's'	1760.0	11000
'f'	2349.3	12500	't'	261.6	1500
'g'			'u'	2637.0	13000
'h'	329.6	2500	'v'	1318.5	9500
'i'	246.9	1000	'w'	1396.9	10000
'j'			'x'		
'k'			'y'		
''l	880.0	7500	'z'	392.0	3500
'm'			' '	493.9	4500
'n'	220.0	500			

Figure 2: Coding table for characters 'a' to 'z' and ' ' (blank). The delays are specified for a sampling frequency of $f_S = 8192$ Hz.

3. Use the coding table and the explained scheme to generate a signal for the following

sequences:

a) 'ghtqhtljtlhtxtnyaayhtxtnxht ht ht htaaa'

b) 'xmmjmmymmktnkkm mmatnokmjmmymmommztnzkmkmm tnakm'

The sequences result in the melody of two 'famous' songs. Create the signals and use MATLABs `soundsc` command to play the generated signal. Which are the 'famous' songs and their interpreters?

To complete the coding table for missing characters you can use the command

```
xn=codingtest('x')
```

for a single character `x`, which provides you with the coded signal for the sequence `xxx`. Again, `codingtest` adds 1000 samples to the beginning and 1000 samples to the end of the signal.

4. Again determine the bandwidth which is occupied by the signal? Provide the upper and lower frequency, for which the spectrogram is smaller than $-20\,\mathrm{dB}$ of the main signal.

Example 3: Wavelet Denoising

Almost 200 years ago Jean Baptiste Joseph Fourier introduced the great idea of Fourier analysis of a signal. The idea of the Fourier transform is based on the decomposition of a signal by means of sinusoidal components of infinite length. While this concept is useful for stationary stochastic processes, it offers limits for nonstationary stochastic processes. Although the STFT appears to be a natural extension of the Fouriers concept, this transform offers limits, e.g. by means of the time frequency resolution. The wavelet transform was founded in the late 80s of the last century. In contrast to Fouriers approach, the wavelet transform maintains a band pass decomposition of signals. This allows a greater degree of freedom, as the basic function are not specifically defined. A major advantage of the (discrete) wavelet transform is given by its efficient implementation using filter banks. While the wavelet transform might cause difficulties by means of its understanding due the different basis functions, this transform has revealed excellent properties for applications like signal compression and denoising applications.

In this example you are dealing with wavelet denoising.

The MATLAB function (see the corresponding MATLAB command, e.g. type `doc wavedec` in MATLABs command window)

```
[C,L] = wavedec(X,N,'wname')
```

performs a discrete wavelet transform of the signal `X` of depth `N` using the wavelet defined by `wname`. Thus, this command implements a wavelet analysis filter bank. For signal synthesis the command

```
X = waverec(C,L,'wname')
```

provides the opposite operation.

The input argument `'wname'` refers to the used wavelet, e.g. you can use terms like `db1` or `haar` to apply different wavelets. The MATLAB command

```
doc wfilters
```

gives you an overview about the wavelets provided by your MATLAB version. Also the command `wfilters` provides the FIR filters used in the analysis filter bank and the synthesis filter bank.

Please provide answers to the following points:

(A) Draw the structure of the filter bank (signal flow graph) of the analysis filter generated be command

```
[C,L] = wavedec(X,5,'wname')
```

Take care about the notation of the signals with respect to the MATLAB implementation. Hint: you can use SIMULINK to draw a proper signal flow graph as it provides all required components (FIR-filters, up-samplers and down-samplers).

(B) Develop a numerical test concept, e.g. by means of MATLABs convolution command or transfer function object, to test, that the filters provided by

```
[Lo_D,Hi_D,Lo_R,Hi_R] = wfilters('wname')
```

can be used in a perfect reconstructing (PR) filter bank.

Compute the delay of a quadrature mirror filter bank for the following wavelets: `'haar'`, `'db10'`, `'sym8'` `'bior2.2'`, and `'rbio2.2'`.

(C) For wavelet denoising so called hard and soft thresholding techniques are known in literature. They are defined by

$$\hat{y}_{\text{hard}} = \begin{cases} y[n] & \text{for} \quad y[n] > \varepsilon \\ y[n] & \text{for} \quad y[n] < -\varepsilon \\ 0 & \text{for} \quad |y[n]| \leq \varepsilon \end{cases}$$

and

$$\hat{y}_{\text{soft}} = \begin{cases} y[n] - \varepsilon & \text{for} \quad y[n] > \varepsilon \\ y[n] + \varepsilon & \text{for} \quad y[n] < -\varepsilon \\ 0 & \text{for} \quad |y[n]| \leq \varepsilon \end{cases}$$

The idea of thresholding is the rejection of signal components with small amplitudes. The `mat`-file `Data Ex3.mat` contains two signals. For each signal a noise-free and a noisy variant are provided. Apply Wavelet denoising to both signals, e.g. find a proper wavelet basis, threshold values and denoising strategy. For each signal provide a plot where you depict the noise free, the noisy and the denoised version.

Evaluate the performance of the denoising approach by computing the SNR before and after the denoising. Use the command `wavedec` for the decomposition and `waverec` for the reconstruction.