Systems and Techniques for Digital Signal Processing
A.Y. 2020-21

Project list and instructions

Every student has to choose **one** of the project reported in the list below (**if you have already selected and have a project chosen in one of the past A.Y. you may keep your old project unless you want to change it**). Some assignments can be solved in groups, if this is explicitly stated in the text.

- Read the project objectives and select one of them;
- Check that **no more than 4 students have already chosen the same project in this shared spreadsheet** that I will keep constantly up-to-date for the whole academic year.
- Immediately after choosing the project, send me an email (as a single or as a group) to inform me about the selected project (e.g. Mario Rossi – project 1). If more than 4 students select the same assignment, the latest ones will be redirected to the remaining projects. In case of no response from my side within one week, the selected project will be considered as approved automatically.

After completing the project, you are supposed to:

1. Write a report of about 10 page per person max. - excluding the title page and annexes, if any. The report has to be strictly focused on the results of the project with a clear explanation of the design choices and the related motivations.
2. Send me by email the zip file containing both the Matlab source files and the report before the written test (**see winter session exceptions below**).
3. Upload ONLY the report in Compilatio at this URL https://www.compilatio.net/submit/hl789 for plagiarism checking.

The project grade will be given before the oral exam and it will be published in the test result pdf file along with the written test results. Further clarifications on the project will be asked during the oral exam.

NOTES

- **The report on the project will be evaluated only if the written test has a sufficient grade.** Students that fail the written test or that withdraw from the exam can reuse the same project/report in a future examination session. **Nonetheless, to avoid confusion, all students that repeat the exam are invited to remind me by email when the report was submitted the very first time and/or to send it to me again**.
- The report must highlight your work and should not include a mere repetition of theoretical aspects already covered in the course. **Excessively verbose, redundant or poorly written reports will negatively affect your grade, regardless of the correctness of the technical content of the assignment.**
- **Copying verbatim large shares of source code, pictures of text from the web or from other students' works is strictly forbidden.** Students that are discovered to submit plagiarized material will be excluded from the exam and will be sanctioned officially as prescribed in the UNITN Students' honor code.

- **WINTER SESSION EXCEPTIONS.** In consideration of the limited time between the end of the classes and the examination dates, in the winter session students may decide to postpone the submission of the assignment as well as the oral exam to the session immediately after the one when they pass the written test. For instance, students passing the written test in January can submit the report before the February written test. Students that instead pass the written test in February can submit the report by the end of February to take the oral exam at the beginning of March (the exact date of this extra oral is to be determined).

Exercise 1 – Data acquisition and filtering

An inverter powering an electric motor generates a 100-Hz ±24 V sine-wave affected by a wideband noise of 0.1 Vrms (root mean square), by a 2nd-order harmonic of amplitude equal to 10% of the fundamental and by 25 higher-order harmonics of amplitude decreasing as 1/*f* with respect to the second.

1. Build the signal with the features described above as if it were continuous-time (namely with a sampling period much smaller than 25th harmonic).
2. Design the data acquisition stage (including an analog anti-aliasing filter, Sample & Hold if needed and ADC with an optimal number of bits and suitable sampling rate) along with an attenuator able to fully exploit the full range of the ADC (assuming that FS=±2.5 V).
3. Design a bandpass IIR filter with a technique of your choice able to attenuate harmonics by at least 40 dB (transition bandwidth and in-band ripple amplitude can be arbitrary, but the filter gain at 100 Hz must be 1).
4. Show that the filter performs as expected both in the time domain and in the frequency domain.

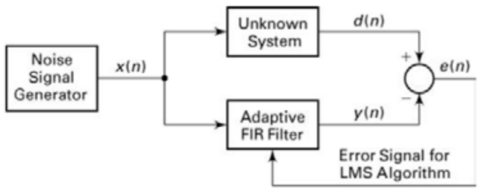
Exercise 2 – Acquisition and processing of nonstationary data

Record your voice while saying “a”, “o” and “u”. You can create your recordings using a computer or other digital recording device. For example, `Audacity' is a free recording and audio editing software system for both Mac and Windows (see http://audacity.sourceforge.net/)

1. Using the recording software, you can create a wav file. You can then read the wav file with MATLAB. Your recordings should be sampled at least at 8 kSa/s. Using MATLAB, collect three audio vectors of some thousands of samples each. Plot and check that if the collected data are correct. You can use the sound function to this purpose.
2. Compute and display the spectrum of the vowels. The vowel sounds are roughly periodic (i.e. mainly deterministic). Based on the spectra that you compute, what is the pitch frequency of your speech? (The pitch frequency is the fundamental frequency of the quasi-periodic voiced speech signal). Can you recognize the harmonics in the spectrum?
3. Select one of the prominent equally-spaced peaks in the spectrum. Determine the frequency of the peak from the spectrum you computed. Design a second-order notch filter (look for more information on notch filters in Matlab) that eliminates the chosen peak without affecting the nearby peaks.
4. Plot the frequency response, pole-zero diagram and impulse response of the digital filter.
5. Compare the signal before and after filtering.

Exercise 3 – System identification through adaptive filtering (max. 2 people)

System identification refers to the ability to emulate the behavior of an unknown system by tuning the parameters of a known model in such a way that the output of the unknown system and the output of the parametric model are reasonably close to each other when they are stimulated by the same signal. A classic approach for the identification of a linear system (see picture below) relies on adaptive FIR filtering. The basic idea is to stimulate both the unknown system and a FIR filter with the same signal (e.g. noise) and to compute the filter coefficients in order to minimize the difference between the outputs of both systems in a least mean squares (LMS) sense.



Assuming that the unknown system is described by the following difference equation:

$$d(n) = a_1 d(n-1) + a_2 d(n-2) + x(n) + b_1 x(n-1) + b_2 x(n-2)$$

where the values of coefficients a_1, a_2, b_1 and b_2 can be chosen arbitrarily (provided that the system is stable), you have to

1. Test the “unknown” system and plot its frequency response;
2. Implement a noise generator to stimulate the system and plot the output of the system;
3. Implement a Wiener filter or an LMS function to adjust adaptively the coefficients of an FIR filter minimizing $E\{e^2(n)\}$ (refer to some literature for a better understanding of LMS adaptive filter design). In the latter case, the function is supposed to have as inputs:
    - the input stimulus sequence;
    - the corresponding desired output *d*;
    - the length *N* of the adaptive FIR filter;
    - a step size parameter Δ controlling the rate of convergence of the algorithm to the optimal solution.
    - (compulsory for 2 people only) The routine above could be also used iteratively in order to increase the value of *N* till when the mean square error $E\{e^2(n)\}$ does not change significantly.
4. Implement the adaptive FIR filter and analyze its behavior with different signals (e.g., noise and a sinewave). Plot the histograms, mean values and variances of the output errors in both cases (noise and sinusoidal stimulation).

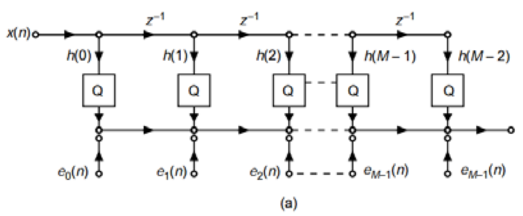
Exercise 4 – Low-pass digital differentiator design (Max. 2 people)

1. Read the paper "Maximally flat Low-pass Digital Differentiators" by Selesnick and design a maximally-flat low-pass differentiator as described in the document.
2. Plot the frequency responses of the designed filters and try to find numerically (e.g. through simulations or interpolation) the relationship between filter parameters *K* (or *L*) and *N* and the bandwidth of the filter.
3. For a wanted pair of *N* and K values, generate two input test signals of your choice (one within the passband and one within the stopband) and show the correct operation of the filter. Also quantify the maximum error in the passband and in the stopband.
4. (compulsory for 2 people only) Read the paper "Linear Phase Low-Pass IIR Digital Differentiators" by Al-Alaoui and implement the low-pass digital differentiator with the cascaded approach described in Section IV of that paper.
5. (compulsory for 2 people only) Plot the frequency responses of the designed filters (denoted as Lp differentiator I and II in the paper) and compare them with the Selesnick ones in terms of: passband maximum error, stopband maximum error and group delay.

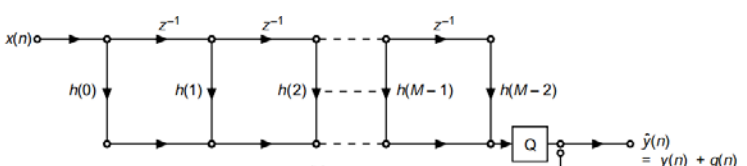
Explain your design choices and comment the results in the report.

Exercise 5 – Quantization noise and numerical issues (Max. 2 people)

1. Write a Matlab function y = quant_fix(x,B, 'Qmode','Omode') performing a fixed-point 2's complement quantization using (B+1) bit in Q_B notation so that the resulting number y lies in $-1 \le y < 1$. The quantization mode option, "Qmode", is either a rounding or a truncation operation. The overflow option "Omode" should perform either saturation or 2's-complement overflow.
2. Digital filters are linear systems, but when quantization is incorporated in their implementation, they become nonlinear. For nonlinear systems it is possible to have an output sequence even when there is no input. A zero-input **limit cycle** is a nonzero periodic output sequence produced by nonlinear elements or quantizers in the feedback loop of a digital filter. Consider the 1st-order recursive system
$$y(n)=0.75\, y(n-1) + 0.125\delta(n)$$
with nonzero initial conditions. The filter is implemented in 4-bit (including sign) using fixed-point 2's-complement fractional arithmetic. Products are rounded to 3-bits.
   - Determine and plot the first 20 samples of the output using a saturation limiter for the addition. Does the filter enter into a limit cycle?
   - Determine and plot the first 20 samples of the output using two's-complement overflow for the addition. Does the filter enter into a limit cycle?
3. Consider a 5th-order FIR system given by
$$H(z)=0.1+0.2z^{-1} + 0.3z^{-2} + 0.3z^{-3} + 0.2z^{-4} + 0.1z^{-5}$$
which is implemented in a direct form using B = 10 bits. Input to the filter is a random sequence whose samples are independent and identically distributed over $[-1, 1]$.
   - Investigate the output quantization errors when all 6 multipliers are used in the implementation (rounding with 2's complement overflow). In this case each multiplier output is quantized (case a). Plot the normalized histogram of the output error, as well as its mean value and variance.
   - Investigate the output quantization errors when quantization is performed only after the final sum (rounding with saturation-based overflow). In this case, typical of fixed-point DSP architectures with a multiply-and-accumulate data-path, (case b) the numerical error is introduced mainly at the end of computation. Plot the normalized histogram of the output error, as well as its mean value and variance.



(a)

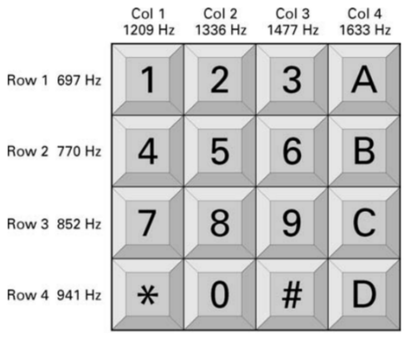


(b)

4. (compulsory for 2 people only) Assume to stimulate the filter with a sinusoidal signal with amplitude equal to 2. Avoid overflow using the most suitable scaling technique you know. Compute numerically the SNR values in both case a and case b, with and without scaling.

Exercise 6 – DTMF tone generation and detection (Max. 2 people)

Dual Tone Multifrequency (DTMF) generation and decoding is commonly used in electronic mail systems and banking systems in which the user can select options from a menu by sending DTMF signals from the phone keyboard. In a DTMF signaling system, a combination of a high-frequency tone and a low-frequency tone represent a specific digit or the characters * and #. The eight frequencies are arranged as shown in the figure to accommodate a total of 16 characters, 12 of which are assigned as shown, while the other four are reserved for future use.



DTMF digit = Row tone + Column tone

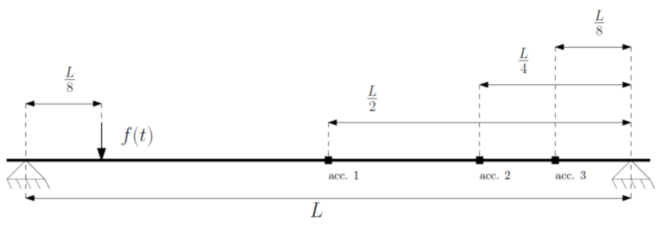
The detection algorithm can rely on the FFT algorithm or on a filter bank implementation. Design the following MATLAB modules:

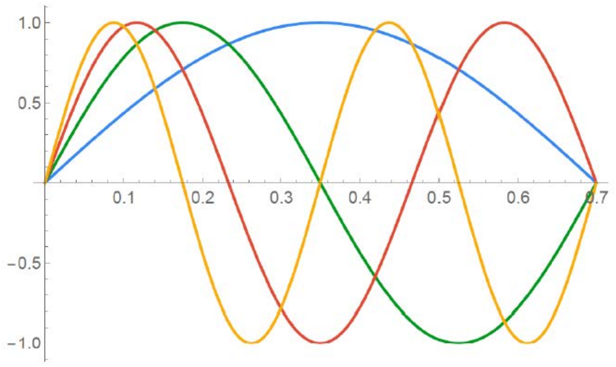
1. A tone generation function that accepts an array containing dialing digits and produces a signal with the tones as shown in the figure. Optionally, you can try to collect the data directly from the keyboard and hear the corresponding sound. The DTMF tone associated with each digit should last 0.5 s and it should be sampled at 8 kHz;
2. A decoding function of your choice that accepts a DTMF signal and produces an array containing the corresponding digits.
3. Test the decoding function and show the result in the frequency domain.

Generate several arrays containing a mix of digits. Experiment with the tone generation and detection modules and plot the result.

Exercise 7 – Vibration mode analysis

A hammer located on the left side of a beam (picture on the left) is used to stimulate its modes of vibration. A load cell next to the hammer is used to measure the force applied by the hammer itself. Three accelerometers in three different positions of the beam are used to measure its vibration at different distances, namely where the amplitude of different modes of vibration of a beam is supposed to be maximum (see picture on the right). Download the "hammer.zip" file from the course repository. This file contains 4 records of force and acceleration data measured over a 10-s time interval at a sampling rate of 10240 Hz (after being filtered by a Butterworth low-pass filter with a cut-off frequency of 5 kHz).





1. After plotting the impulse responses of the system based on the accelerometer data of all records, determine the average frequency response of the vibrating beam in the three points where the accelerometers are located using different records.
2. Write a routine to detect the peaks of the spectrum and to estimate the frequency and the maximum amplitude of the modes of vibration in the band [0,2] kHz. Estimate mean values and standard deviations of such parameters.
3. Design a set of notch filters of your choice to remove the peaks. Show the correct operation of the filter designed. Justify your design choices in the report.

Exercise 8 – Synchrophasor estimation (Max. 2 people)

As known, in Europe the electric voltage waveforms in low-voltage distribution grids have a nominal frequency of 50 Hz and a root mean square (RMS) amplitude of 220 V. A synchrophasor is defined as the phasor (namely the magnitude and the phase) of a voltage waveform at a given reference time.

1. Assume to collect windows of N samples (N being a generic odd number) with a random initial phase at the reference time t_r chosen in the center of each observation interval (the sampling rate can be set equal to f_s= 2.4 kHz). Using both the plain DFT (not the FFT) and the coherent sampling (with a sampling period of your choice) estimate the synchrophasor at t_r. Evaluate (and try to minimize) the magnitude and phase error at 50 Hz.
2. Assume that the nominal waveform is corrupted by a second- and third-order harmonic equal to 10% of the nominal amplitude and by a random noise with a SNR= 50 dB. Estimate mean value and variance of the estimation errors in such conditions.
3. Check what happens in the case of noncoherent sampling and if you change the number of observed cycles. Compare and comment the results.
4. (Compulsory for 2 people). Repeat steps 1-3 using the Kaiser window with different values of parameter β. Compare the results.

Exercise 9 – ECG total variation filtering (Max. 2 people)

Given a noisy collected *y* = [*y*(1),…,*y*(*N*)], the so-called total variation (TV) filtering is able to remove the noise by finding the signal ***x*** = [x(1),…,x(*N*)] that minimizes the cost function *F*(***x***) defined as follows:

$$F(x) = \sum_{n=1}^{N} |y(n) - x(n)|^2 + \lambda \sum_{n=2}^{N} |x(n) - x(n-1)|^2$$

The TV filter is particularly effective in removing the noise when the underlying signal exhibits sudden changes (e.g. piecewise constant signals). Filter performances strongly depend on parameter λ. The goal of this exercise is to design a TV filter able to remove the noise affecting electrocardiographic (ECG) signal, possibly comparing the result of this filter with the results obtained with other filters.

1. Download one or more ECG data record (sampled at 360 Hz per channel with 11-bit resolution over a 10 mV range) (e.g. from https://physionet.org/content/mitdb/1.0.0/)
   Plot the data in the time domain. Choose a noisy record. If you cannot find a noisy record, add artificially a zero-mean random noise large enough to hide significantly the original ECG signal.
2. Relying on the information available both in the literature and online, implement a TV filter for the problem at hand (see for instance https://www.aaai.org/ocs/index.php/SSS/SSS11/paper/viewFile/2451/2902). Explain your design choices in the report.
3. Plot the waveform before and after filtering and evaluate the filter performance by computing the respective SNR values.
4. (compulsory for 2 people only). Repeat steps 1-3 using a so-called median filter (check Matlab documentation) and a standard linear (e.g. Butterworth) filter.

Exercise 10– Digital filter implementation using the Fixed-Point Toolbox (Max. 2 people)

The Fixed-Point Toolbox™ provides fixed-point data types and arithmetic in MATLAB®. The toolbox lets you design fixed-point algorithms using MATLAB syntax and execute them at compiled C-code speed. The Fixed-Point Designer™ defines its own functions and Simulink models and it is able to handle critical numerical issues, such as scaling and numerical noise, e.g. due to rounding and quantization of coefficients. The goal of this exercise is to implement IIR and FIR filters using the Fixed-Point Designer toolbox.

1. Using the bilinear transformation method, design a digital Chebyshev-II lowpass filter operating at a sampling rate of 80 kHz with a passband edge frequency at 4 kHz, a passband ripple of 0.5 dB, and a minimum stopband attenuation of 45 dB at 20 kHz.
2. Implement the canonical form and the cascade form of the resulting filter using the functions and the data types of the Fixed-Point Toolbox™ with 8-bit, 12-bit and 16-bit precision.
3. Plot the zero-pole diagrams and the frequency responses of the different implementations. Show and compare their behavior.
4. (Compulsory for 2 people only) Repeat steps 1-3 assuming to design an equiripple FIR filter with the same specifications.

Exercise 11 – Digital signal processing algorithms ARVA signal detection (Max. 3 people)

An ARVA (Appareil de Recherche de Victimes d'Avalanches) is a safety device transmitting a 457 kHz signal that can be detected by rescuers at a distance > 100 m when the person wearing the device is buried by snow. Most modern ARVA devices are based on DSP algorithms and in the future they could be transported by UAVs to speed up the searching process. Download and read carefully the paper "Digital Signal Processing in Triple Antenna Arvas" by Salos, Lera and Villarroel, which presents a DSP technique for ARVA signal detection. The goal of the exercise is to simulate the magnetic field of a possible ARVA device in order to detect it in the presence of noise. In particular,

- Implement the data acquisition and FIR filtering stage in the range [3750 Hz, 4250 Hz]. If possible adopt the Adaptive Line Enhancer (ALE) approach described in Section 4 of the paper, or any other bandpass filter of your choice;
- Implement the periodogram-based signal detector described in Section 5.
- (Compulsory for 3 people only) Implement the signal Maximum Likehood (ML) parameter estimator of the magnetic field described in Section 6, under the influence of different levels of noise.

Explain your design choices and comment the results in the report.

Exercise 12– Digital PWM modulator

Pulse Width Modulators (PWM) are widely adopted in mechatronics applications, e.g. for motor control. In digital PWM controllers, the pulse width is controlled using a digital counter driven by the internal clock signal. Therefore, for a given internal clock frequency, the resolution of the counter is inversely proportional to the switching frequency. Consequently, a higher switching frequency results in lower PWM resolution. The poor resolution of the PWM operation may cause undesirable lower order harmonic components in the output current. The goal of this paper is to implement a DSP technique to mitigate this problem.
Read carefully the paper "Quantization Noise Shaping in Digital PWM Converters" by Norris et al

- Implement and plot the frequency response of a first, second and third-order sigma-delta noise shaper followed by a digital PWM modulator as described in that paper (not considering of course the power electronics output stage based on MOS transistors).
- Simulate the behavior of the various sigma-delta digital PWM modulators, comparing the various solutions and showing differences and improvements in terms of signal-to-noise-and distortion ratio (SNDR) and total harmonic distortion (THD).

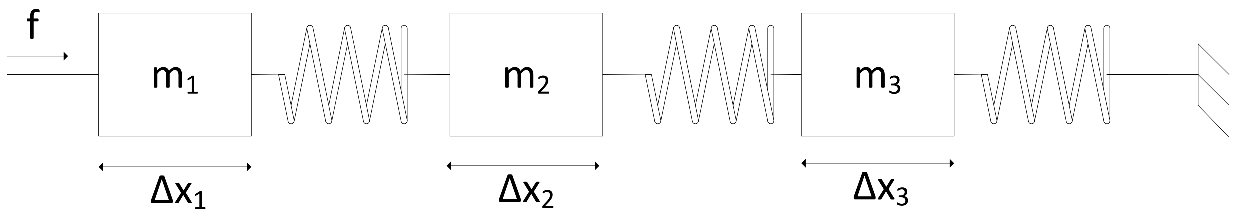
Explain your design choices and comment the results in the report.

Exercise 13 – System identification example (max. 2 people)

Consider the mass-spring model shown in the picture and download the "mass-spring.zip" file from the repository of the course. The .zip file contains a "data_impulses" file including the following data in different columns: time (with f_s = 200 Hz), linear displacements Δx_1, Δx_2 and Δx_3 (expressed in encoder counts) and voltage V used to power the motor generating a train of impulsive force f. In particular, $f = (k_a \cdot k_t \cdot k_{mp}) \cdot V$ where $k_a \approx 2$ A/V, $k_t \approx 0.1$ Nm/A and $k_{mp} \approx 1/26.25$ 1/m. The relationship between the linear displacements of the three masses and the encoder counts is given by the following expression, i.e.

$$\Delta x = 2\pi r_e \frac{\Delta \text{counts}}{16000}$$

where $2\pi r_e = 0.0706$ m is the circumference length of the encoder and 16000 is the number of counts per revolution.



The sequences $\Delta x_1(n)$ and $\Delta x_2(n)$ and $\Delta x_3(n)$ can be (approximately) regarded as the outputs of the same linear system. Assuming that the overall mass-spring model is a 6th order LTI system, and remembering that the applied impulses are not ideal, do the following, i.e.

- Compute the frequency responses $H_1(e^{j\omega})$, $H_2(e^{j\omega})$ and $H_3(e^{j\omega})$ of the subsystems generating $\Delta x_1(n)$ and $\Delta x_2(n)$ and $\Delta x_3(n)$ (Note: since you have multiple impulse responses in the data source file, compute the average of different frequency responses to obtain a more accurate estimate).
- Perform an optimal least-squares numerical fitting of the experimental frequency responses, in order to find the positions of poles and zeros of each subsystem and reconstruct the continuous-time frequency responses $H_1(\Omega)$, $H_2(\Omega)$ and $H_3(\Omega)$ (Note: the position of the poles should be the same for all transfer functions since the natural modes of the system are the same).
- Compute the step responses of the subsystems and compare them with those obtained experimentally and reported in the "data_steps" file. (Note: remember to normalize the experimental data to work with unit steps).

Explain your design choices and comment the results in the report.