Digital signal processing lab

Hugues Garnier
Mayank Shekar Jha

December 2019

Table des matières

Introduction

MATLAB, for MATrix LABoratory is a powerful software that can deal with complex problems in various scientific fields, exploit and analyse data in an interactive environment.

It is particularly widely used by scientists and engineers to perform various signal processing tasks.

The *Signal Processing* toolbox includes functions and graphic interfaces (Apps) for generating, viewing, processing and filtering signals. Algorithms are available to :

- approach the signal spectrum by discrete Fourier transform ;
- calculate the power spectra of random signals ;
- design and analyse filters ;
- estimate parametric models of random signals ;
- perform time-series forecasting.

This toolbox is used to analyse and compare the signals in the time, frequency and time-frequency domains, identify patterns and trends, infer characteristics, and develop and validate customized algorithms in order to extract relevant information on data coming from very diverse fields.

You will apply digital signal processing on audio data. Bring your headphones in order to hear the effects of your digital signal processing as well as your smartphone to record audio sounds.

Lab 1

Fourier analysis of digital signals

The work done during this lab will be noted in a report and marked. You are asked to follow the instructions given below to write your report.

Instructions for writing your laboratory report

A report is a scientific document. It must therefore respect a certain structure. Your report should be structured as follows :

- a general introduction specifying the objectives of the lab
- For each assignment or exercise :
  - ▷ a brief presentation of the expected outcomes
  - ▷ a description of the obtained results in graphical and or numerical form
  - ▷ a critical analysis of the results
  - ▷ a short conclusion
- a general conclusion explaining what has been understood during the lab and any difficulties encountered.

Sending your report to the tutor

Your report must be written in pairs and sent by email to the tutor in the form of a single pdf format file attached to the message. It must be sent before the deadline given during the lab session.
Please indicate the following "subject" for your email to send your report to the tutor : **TP1_TNS_Binom1name_Binom2name"**

Downloading of the data needed for the lab

1. Download the zipped file **Lab1_TNS.zip** from the course website and save it in your Matlab working directory.
2. Start Matlab and **go to your Lab1_TNS working directory** which includes the data files that will be analyzed during the lab.

Recording of sound files

You will do the recordings of sounds with your own hardware, for example, with the microphone of your smartphone.

Layout of the Lab

The lab is structured into two main parts :

1. A tutorial introduction of the graphical interface `signalAnalyzer` to determine the fundamental frequency of a tuning fork.
2. Several assignments where you have to analyze your data recordings and other real-life data with the methods and theory that you learn in Lectures 1 to 3.

1.1 Tutorial introduction of the SignalAnalyser App - Determination of the fundamental frequency of a tuning fork

A tuning fork is a small steel tool consisting of a U-shaped rod as shown in Figure 1.1 which can be used to tune certain musical instruments.

Figure 1.1 – Tuning fork

Load the tuning fork data into Matlab by typing the following command :
```
load tuning_fork;
```
This file contains a `y` column vector with the sound samples produced by the tuning fork and a `fs` scalar for the sampling frequency.
Listen to the sound made by the tuning fork by entering the following command :
```
sound(y,fs);
```

Convert the signal to a Matlab timetable object. This will facilitate the automatic labelling of the time and frequency axis in the SignalAnalyzer tool.
```
tuning_fork = timetable(seconds((0:length(y)-1)'/fs),y);
```

Matlab now has a `signalAnalyzer` graphical interface used to display the features of a signal in the time and frequency domains.
At any time, you can consult the technical documentation of this graphical tool by entering, in the command window of Matlab :
```
doc signalAnalyzer
```

The graphical interface can be open by clicking on its icon in the Apps menu or by typing in the command window :
```
signalAnalyzer;
```

A graphical window should then open. Select and drag the timetable to a main display windows on the right frame. The time-domain plot of the tuning fork data should be displayed as shown in Figure 1.2.

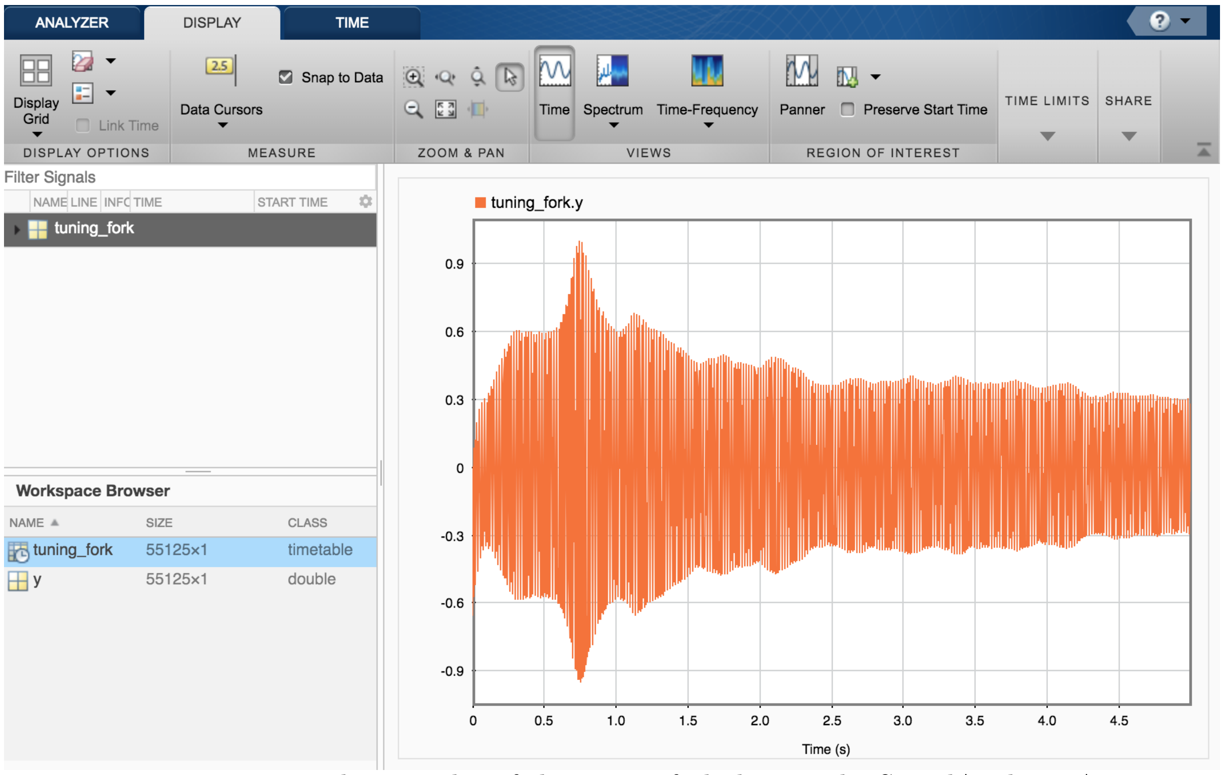


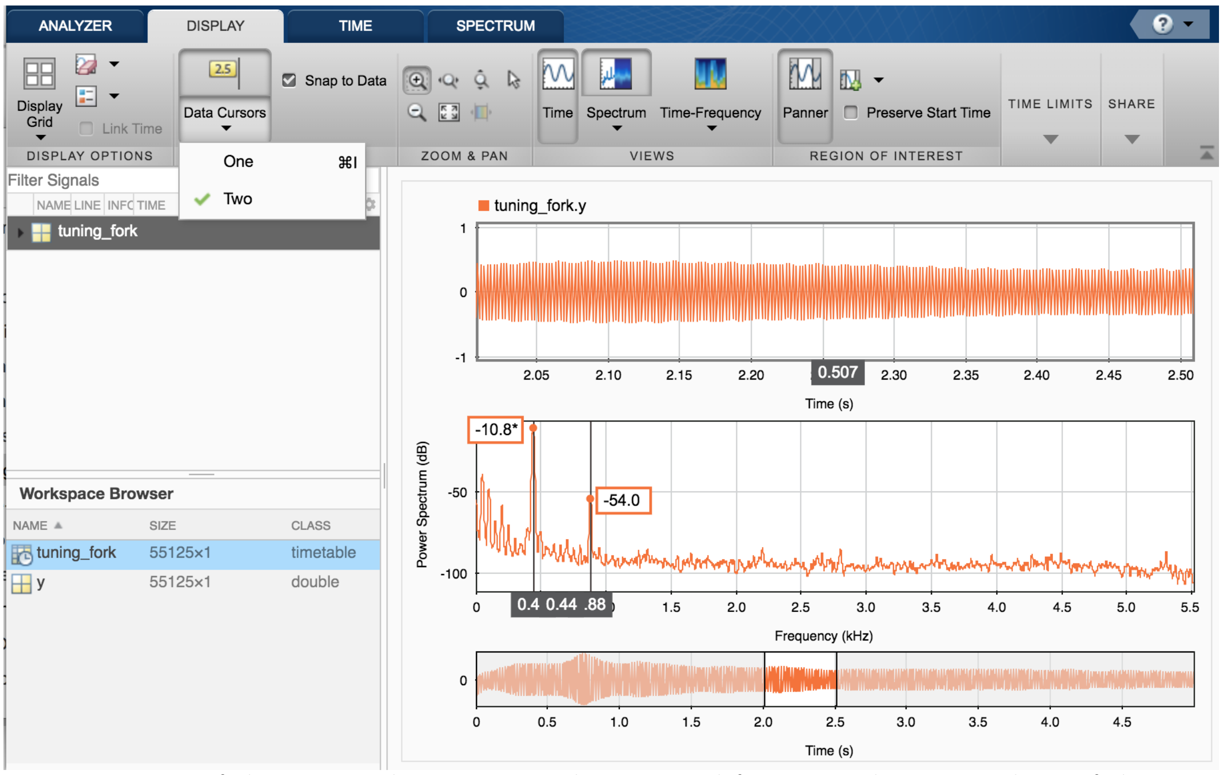
Figure 1.2 – Time-domain plot of the tuning fork data in the SignalAnalyser App

Check that the signal is plotted over a time range of about 5s.

From the time-domain plot, it can be noticed that the amplitude of the signal decreases much slowly after 2s. We will concentrate on this time range for the signal analysis.

Click on the Spectrum tab to display the spectrum of the signal.
Click then on the paner tab and select the time window corresponding to 2 to 5 s as shown in Figure 1.3.

From the displayed spectrum, we can clearly observed two main peaks : the first having the largest amplitude for $f = 440$ Hz which represents the fundamental frequency of the tuning fork. The second appears at $f = 880$Hz and is therefore a harmonic of the periodic signal with much lower amplitude than the fundamental, which can be neglected here.
It is recalled that the frequency band audible by the human ear ranges from 20 Hz to 20 kHz. When the frequency ranges from 20 to 200 Hz, the sound is low-pitched (*son grave*). The sound is said medium for a frequency between 200 and 1000 Hz and we get a high-pitched sound (*son aigu*) when the frequency is between 1000 and 15000 Hz. The tone produces by the tuning fork is therefore medium.



NAME	SIZE	CLASS
tuning_fork	55125x1	timetable
y	55125x1	double

Figure 1.3 – Use of the paner tab to improve the time and frequency-domain analysis of the tuning fork data in the SignalAnalyser App

Reduce further the time window of the paner around 2.5s for example to make appear the sine wave form of the tuning fork sound as shown in Figure 1.4.

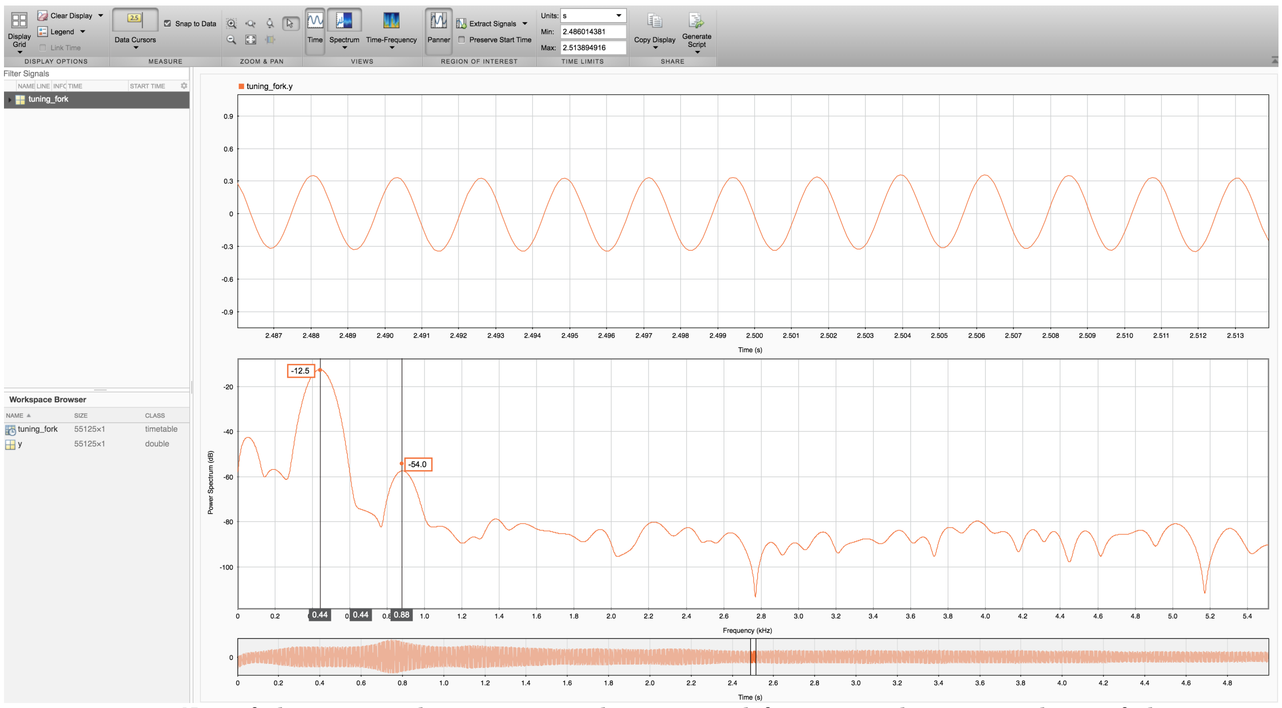


Figure 1.4 – Use of the zoom tab to improve the time and frequency-domain analysis of the tuning fork data in the SignalAnalyser App

1.2 Assignments - Application of Fourier analysis in different fields

The assignments aims at analyzing real-life signals by using Fourier analysis coming from different fields :

1. Music instruments : trumpet tone analysis
2. Telecom : decoding a phone number from DTMF data
3. Human sounds : can you produce a high-pitched whistle ?
4. Animal sounds : blue whale vocalization analysis
5. Solar physics : sunspot activity and space weather

1.2.1 Trumpet sounds

This assignment aims at determining the fundamental frequency of trumpet sounds by Fourier analysis.

Load the file named `trumpet.mat` that contains audio data stored in the variable `y` from an actual trumpet playing note B.

The trumpet sound was sampled at the standard CD sampling frequency stored in the variable `fs=44.1` kHz.
Load the trumpet data into MATLAB and listen to the sound made by the trumpet by typing the following command :

```
load trumpet;
sound(y,fs);
```

Convert the signal to a MATLAB timetable object.

```
trumpet = timetable(seconds((0:length(y)-1)'/fs),y);
```

By using the signal analyzer App, determine :

1. the fundamental frequency of this trumpet sound
2. the number of non-negligible harmonics and their frequencies. A -40 dB attenuation of a harmonic in comparison with the fundamental or with the harmonic having the largest power will be considered here as the threshold of audibility. Indeed, the sounds below this value are of little interest because they are often masked by louder sounds, and therefore uninformative sounds.

Repeat the analysis above to the data stored in a file named `spurious_trumpet.mat`. It contains audio data recorded from an actual trumpet still playing note B but where an interfering or spurious signal is added to it.

Listen to the sound made by the spurious trumpet.

By using the signal analyzer App :

1. determine the fundamental frequency of the spurious sound.
2. suggest a way to eliminate the spurious sound from the recorded data. *You will test your proposed solution during the second lab.*

1.2.2 Decoding DTMF tone

This assignment aims at decoding a phone number from dual-tone multi-frequency (DTMF) signaling tone data by Fourier analysis.

A dual-tone multi-frequency (DTMF) or VF (Voice Frequency) is a telecommunication signaling system using the voice-frequency band over telephone lines between telephone equipment and other communication devices and switching centers. These codes are emitted when pressing a key on the telephone keypad, and are used for dialling phone.
Technically, each key on a telephone corresponds to a pair of two audible frequencies that are transmitted simultaneously. Here we look at the DTMF code used in the United States which uses seven distinct frequencies that can encode the twelve telephone keypad shown in Figure 1.5. These seven frequencies are shown on the left edge and on the bottom of figure 1.5.
They are also listed in the following table.

697 Hz	770 Hz	852 Hz	941 Hz	1209 Hz	1336 Hz	1477 Hz

The sound generated by pressing a key result in the sum of two sine waves with associated frequencies. Pressing button 1 will therefore produce the following signal :

$$y_1 = \frac{1}{2}\big(\sin(2\pi \times 697 \times t) + \sin(2\pi \times 1209 \times t)\big);$$

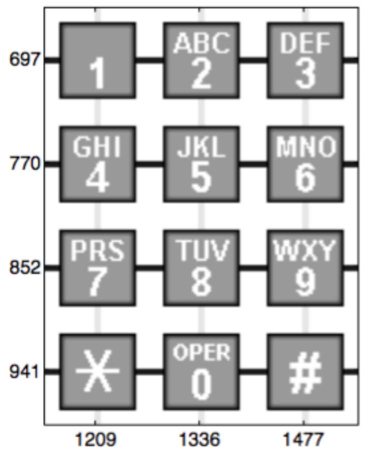


Figure 1.5 – telephone keypad

Figure 1.6 represents the signal and its amplitude spectrum when pressing the "1" key of the dialpad. The two frequencies can be clearly identified from the spectrum.

A `hidden_number.mat` file containing an American telephone number is available in the downloaded file.
This file contains the column vectors `y` et `t` containing the number samples and recording instants as well as sampling frequency `fs`.

Load the DTFM data in MATLAB by typing the command :

```
load hidden_number;
```

Plot the time evolution of the telephone number over the 9 seconds of recording using the following command :

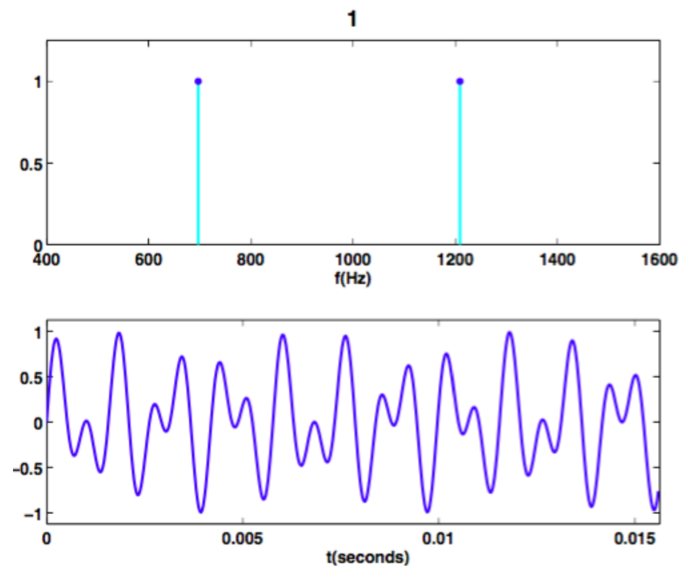


Figure 1.6 – Time evolution and spectrum of the signal resulting from pressing the "1" key on the dialpad

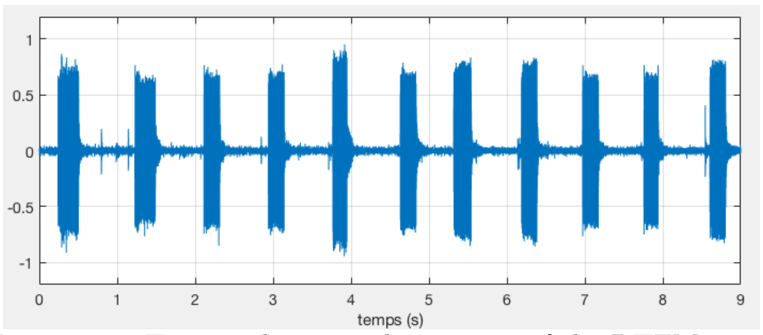


Figure 1.7 – Time evolution and spectrum of the DTFM signal

```
plot(t,y),shg
```

Make sure you get a similar plot to the one shown in Figure 1.7.
Address the following questions :

1. How many digits are in the hidden phone number ?
2. Is it possible to determine the phone number from the time evolution of the composite signal ?

Listen to the recorded number by typing the following command :

```
sound(y,fs);
```

Use the `SignalAnalyzer` App to address the following questions :

1. Display the time-domain evolution and the spectrum of the recorded best two second signal.
2. Plot the spectrogram view (time-frequency representation) of the hidden number by clicking the Time-Frequency tab. On the Spectrogram tab, under Time Resolution, select Specify. Enter a time resolution of 0.8 second and zero overlap between adjoining segments. From the spectrogram view, it can be observed that the first tone has its frequency content concentrated around 697 Hz and 1209 Hz, corresponding to the digit '1' in the DTMF standard. Continue the analysis for determining the rest of the phone number.

3. Another way to determine the phone number is to use the panner and to define time-windows of about 0.8s. The spectrum obtained for the time-domain windowed signal should allow you to determine the phone number.
4. Using the Internet, find the name of the company which owns the number.
5. Find the address in the United States and use Google Maps to view the main entrance of the US company. A clue is provided in Figure 1.8.

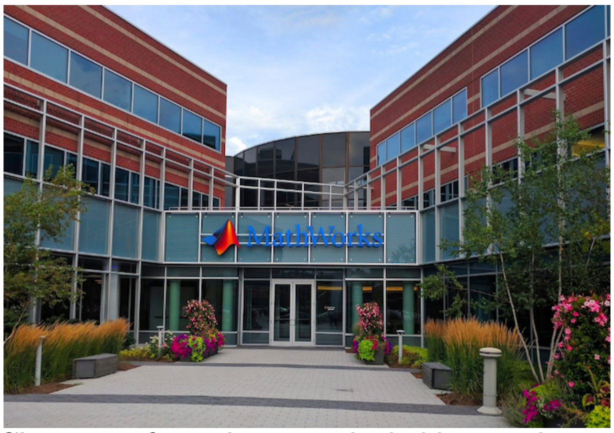


Figure 1.8 – Clue to confirm who owns the hidden number to be identified

In practice, the detection of DTMF tones is performed by using digital processing techniques such as the Goertzel algorithm. To learn more, browse :
`https://en.wikipedia.org/wiki/Goertzel_algorithm`

1.2.3 Whistles

This assignment aims at analyzing the time and frequency content of your whistle.

Use your mobile phone to record you whistling for about 10s. Try to whistle in the medium band as possible. Do not blow straight into the microphone. Hold the microphone from the side, so the sound is recorded from the side. This will reduce the incidence of disturbing blowing sounds.

Send the recorded file by e-mail to your address and save it to your working directory. A whistle data recorded from an iPhone 7 is available in the downloaded file for those who forgot to bring their own smartphone.

```
 [y,fs] = audioread('WhistleiPhone7.mp3');
whistle = timetable(seconds((0:length(y)-1)'/fs),y);
```

Open the signal analyzer App and select the best two seconds.

Address the following questions/tasks :

1. Display the spectrum of the recorded best two second signal.
2. What is the sampling frequency by default of the mobile phone ?

3. Does your mobile phone include an anti-aliasing filter applied to the analogue signal before digitizing it ? If so, estimate the low-pass filter cut-off frequency and check its consistency with the sampling frequency.
4. Determine the dominating frequency of your whistle from the spectrum.
5. Can you observe any significant harmonics from the spectrum ?
6. Deduce the type of sound (low-pitched, medium or high-pitched) produced by your whistle.

Use again your mobile phone to record you and your partner whistling simultaneously for about 10s. One of you should whistle in the medium band and the other in the high-pitched band. Address the following questions/tasks :

1. Display the spectrum of the recorded best two seconds signal.
2. Determine the dominating frequency of each of the two whistles.
3. Deduce the type of sound (low-pitched, medium or high-pitched) produced by the two whistles.

1.2.4 Blue whale vocalization

This assignment aims at analyzing a blue whale vocalization in the time and frequency domains.
A Pacific blue whale recording is available in Matlab. The file is from the library of animal vocalizations maintained by the Cornell University Bioacoustics Research Program. Load the blue whale vocalization data :

```
whaleFile = fullfile(matlabroot,'examples','matlab','bluewhale.au');
```

Convert the signal to a timetable.

```
[y,fs] = audioread(whaleFile);
whale = timetable(seconds((0:length(y)-1)'/fs),y);
```

The time scale in the data has been compressed by a factor of 10 to raise the pitch and make the sound more audible. To hear the sound made by the blue whale, type :

```
soundsc(y,fs);
```

Open the Signal Analyzer App and drag the timetable to the main frame on the right. Four features stand out from the noise. The first is known as a trill, and the other three are known as moans.

By using the signal analyzer App, analyze the time and frequency contents of :

1. the trill ;
2. each of the three moans ;
3. deduce the type of sound (low-pitched, medium or high-pitched) produced by a blue whale vocalization.

1.2.5 Sunspot activity

This assignment aims at analyzing sunspot data which is available from NASA for the years 1749-2012.

Sunspots are temporary phenomena on the Sun's photosphere that appear as spots darker than the surrounding areas on the side of the Sun visible to an Earth observer.
The Sun is the source of the solar wind ; a flow of gases from the Sun that streams past the Earth at speeds of more than 500 km/s. Disturbances in the solar wind shake the Earth's magnetic field and pump energy into the radiation belts. Regions on the surface of the Sun often flare and give off ultraviolet light and x-rays that heat up the Earth's upper atmosphere. This **"Space Weather"** can change the orbits of satellites and shorten mission lifetimes. The excess radiation can physically damage satellites and pose a threat to astronauts. Shaking the Earth's magnetic field can also cause current surges in power lines that destroy equipment and knock out power over large areas on the planet. As we become more dependent upon satellites in space and electricity power on Earth, we will increasingly feel the effects of space weather and therefore we need to better understand and predict it.
To learn more about space weather, browse :
`https://en.wikipedia.org/wiki/Space_weather`

The nature and causes of the sunspot cycle constitute one of the great mysteries of solar astronomy. While we now know many details about the sunspot cycle, we are still unable to produce a model that will allow us to reliably predict future sunspot numbers using basic physical principles. This problem is a little like trying to predict the severity of next year's winter or summer weather.
For almost 300 years, astronomers have tabulated the number and size of sunspots every year. Load the sunspot data available in the file that you have downloaded :

```
load sunspot;
```

Convert the signal to a timetable.

```
sunspot = timetable(years(year),y);
```

Use the signal analyzer App to address the following questions/tasks :

1. Analyze the time and frequency contents of the sunspots since 1749.
2. Determine the main cycle/period in years of the sunspot activity.
3. Predict in which year we should reach a new maximum sunspot number.

Lab 2

Digital filtering and modelling of stochastic signals

The work done during this lab will be noted in a report and marked. You are asked to follow the instructions given below to write your report.

Instructions for writing your lab report

A lab report is scientific document. It should be self-contained : that is, someone who has never seen the lab instructions should be able to understand the problems that you are solving and how you are solving them. All the choices you have made should be clearly motivated. Comment and explain all your plots and your Matlab-code, so as to make it easy to follow your way of thinking.
Laboratory reports that do not fulfill the required standards will not be marked by the teaching staff. See instructions given in Lab 1 for more information about the structure of your report.

Sending your report to the tutor

The reports should be written in groups of two students and sent by email to the tutor in the form of a single pdf format file attached to the message. It must be sent before the deadline given during the lab session.
Please indicate the following "subject" for your email to send your report to the tutor : **Lab2_DSP_Binom1name_Binom2name"**

Downloading of the data needed for the lab

1. Download the zipped file **Lab2_DSP.zip** from the course website and save it in your Matlab working directory.
2. Start Matlab and **go to your Lab2_DSP working directory** which includes the data files that will be exploited during the lab.

Recording of sound files

You will do some recordings of sounds with your own hardware, for example, with the microphone of your smartphone.

Layout of the Lab

The lab is structured into two main parts :

1. A tutorial introduction of the interactive `Filter Designer` app to design digital FIR and IIR filters and to implement then using the command line functions, `filter`.
2. Several assignments where you have to analyze your data recordings and other real-life data with the methods and theory that you learn in Lectures 1 to 7.

2.1 Tutorial introduction of the Filter Designer App

We will create a signal to be used in this tutorial. The signal is a 100 Hz sine wave in additive white Gaussian noise of standard deviation 0.5.

```
Fs = 1000;
N = 10000;
t = linspace(0,1,N);
x0 =cos(2*pi*100*t);
x = x0+0.5*randn(size(t));
```

— Start the App by entering `filterDesigner` at the command line or by selecting it from the App Menu. The main window of the filterDesigner App should then open.

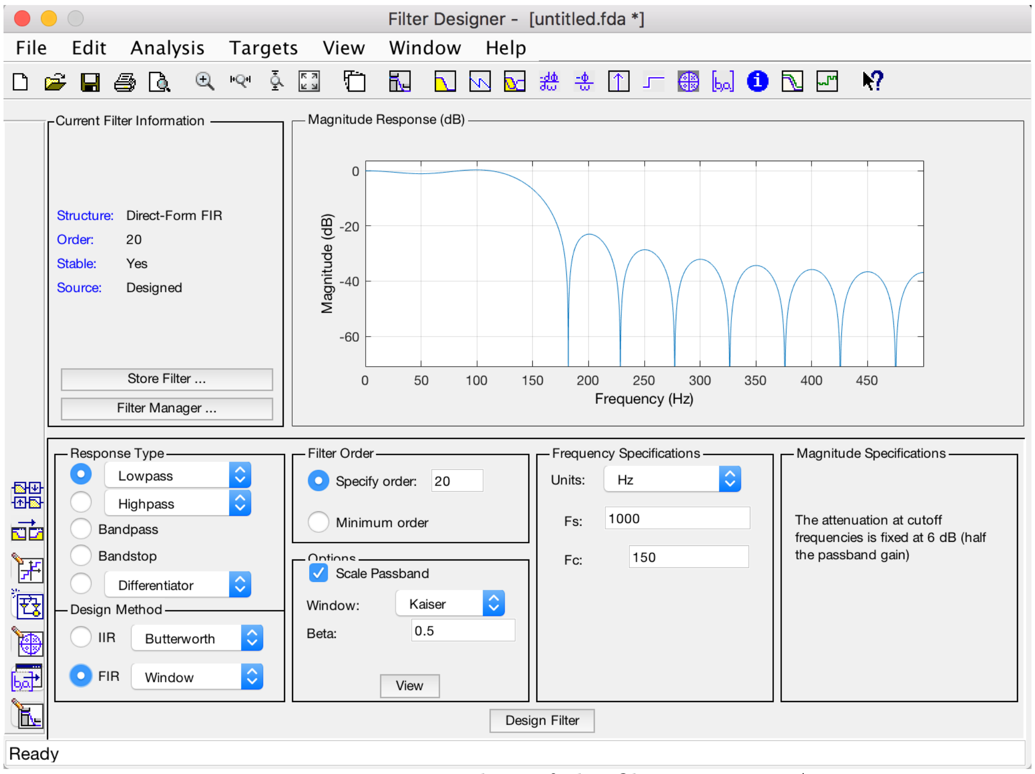
— Set the Response Type to Lowpass.

— Set the Design Method to FIR and select the Window method.

— Under Filter Order, select Specify order. Set the order to 20.

— Under Frequency Specifications, set Units to Hz, Fs to 1000, and Fc to 110.

— Click Design Filter.



Structure: Direct-Form FIR
Order: 20
Stable: Yes
Source: Designed

Response Type: [✓ Lowpass], Highpass, Bandpass, Bandstop, Differentiator

Design Method: IIR Butterworth, [✓ FIR Window]

Filter Order: [✓ Specify order: 20], Minimum order

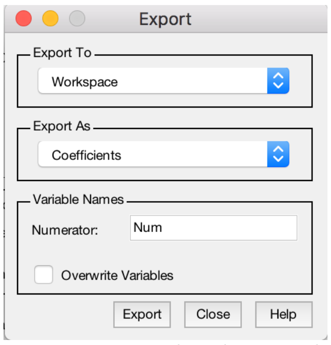
Options: [✓ Scale Passband]
Window: Kaiser
Beta: 0.5

Frequency Specifications
Units: Hz
Fs: 1000
Fc: 150

Magnitude Specifications
The attenuation at cutoff frequencies is fixed at 6 dB (half the passband gain)

Figure 6.1 – Main window of the filterDesigner App

- Select File > Export... to export your FIR filter to the Matlab workspace as coefficients or a filter object. In this example, export the filter as coefficients. Specify the variable name as `Num`
- Click Export.


Export

Export To: Workspace

Export As: Coefficients

Variable Names

Numerator: Num

Overwrite Variables: ☐

Figure 6.2 – Window to export the designed filter as coefficients

- Go to the command window and implement the filtering of the noisy sine signal with the `filter` function by using the exported filter. Plot the result for the five periods of the 100 Hz sinusoid.

```
xf = filter(Num,1,x);
plot(t,x,t,xf)
xlim([0.5 0.55])
xlabel('Time (s)')
ylabel('Amplitude')
legend('Noisy signal','Filtered signal')
```

- You can implement zero-phase filtering with the `filtfilt` routine :

```
xzp = filtfilt(Num,1,x);
plot(t,x0,t,x,t,xf,t,xzp)
xlim([0.5 0.55])
xlabel('Time (s)')
ylabel('Amplitude')
legend('Noise-free','Noisy signal','Filtered signal','Zero-phase filtered
signal')
grid
```

The `signalAnalyzer` graphical interface used in Lab 1 can be very useful in the following assignments to display the features of the original signal and its filtered version in the time and frequency domains.

2.2 Assignments - Digital filtering and modelling of stochastic signals

The assignments aim at designing digital filtering and modelling of stochastic signals coming from different fields :

1. Noisy artefact : How to filter out an artefact noisy harmonic in a musical signal ?
2. Spikes : How to cancel out unwanted spikes on an electrical signal ?
3. Sunspots : How to smooth monthly sunspot data ?
4. Whistles : How to separate two whistles simultaneously recorded ?
5. Vowels : How to model and artificially reproduced the sound of a vowel ?

2.2.1 Noisy artefact in a piece of music

The file `music_track.mat` contains a musical signal stored in `y` and sampled at `fs` that is contaminated by a noisy artefact.
Address the following questions/tasks :

1. Listen to the noisy signal.
2. Determine the frequency of the spurious harmonic.
3. Design a filter to restore the original piece of music.
4. Implement your filter and check its efficiency by listening to the filtered signal.

2.2.2 Spiky electrical voltage

The file `electrical_voltage.mat` contains a voltage signal stored in `y` and sampled at `fs` that is contaminated by unwanted spikes.
Address the following questions/tasks :

1. Plot the electrical signal and observe the spikes.
2. Develop and implement a processing method to filter out the unwanted spikes on the electrical signal.
3. Determine the fundamental frequency of the electrical signal.
4. Was the signal recorded in France ?

2.2.3 Daily sunspot data

The file `daily_sunspot_data.mat` contains the daily sunspot data stored in `sunspot` since 1800. The days of measurement are stored in `day`.
Address the following questions/tasks :

1. Plot the daily sunspot data.
2. Develop and implement a processing method to smooth the sunspot data.
3. Determine the main cycle/period in years of the sunspot activity from the daily data.

4. The file `yearly_sunspot_data.mat` contains the daily sunspot data stored in `sunspot` since 1700. The years of measurement are stored in `years`. Superpose the processed daily data and the yearly daily data.
5. If the two signals are different, figure out why. Suggest how the yearly sunspot data have been computed from the daily measurement. Implement your suggestion to validate your conjecture.

2.2.4 Separation of two whistles

Use your mobile phone to record you and your partner whistling simultaneously for about 10s. One of you should whistle in the medium band and the other in the high-pitched band.
Address the following questions/tasks :

1. Display the spectrum of the recorded best two second signal.
2. Determine the dominating frequency of each of the two whistles.
3. Design a digital processing approach to separate the two whistles.
4. Implement your approach and check its efficiency by listening to the two filtered signals.

2.2.5 AR modelling of a vowel

The goal is to try to model the sound of a vowel using an AR model of an appropriate order and then simulate the model and listen to the result.

1. Record you while pronouncing the vowel *a* for two seconds.
2. Remove the transients at the beginning and the end of the recording if necessary, and decimate the data so that the sampling frequency is around 8000Hz (use the Matlab `decimate` function) :
   ```
   ydec=decimate(y,R); % R is the decimating rate to be chosen according to
   the sampling frequency of your smartphone
   ```
3. Model the vowels using an AR model of order 8 by using the least squares method.
4. Is the estimated model any good ? Motivate your answer.
5. Compute the one-step ahead prediction of the vowel signal.
   ```
   yhat=predict(Mar,data,1);
   N=1000;
   stem([ydec(N:N+49) yhat(N:N+49)])
   ```
6. Listen to the predicted vowel :
   ```
   sound(yhat.y,fs/R)
   ```
   `fs/R` is the new sampling frequency after the possible decimation step.
7. Repeat the AR modelling for the vowel *o*.
8. Repeat the AR modelling for the consonant *m* or *n*.