# MAP555 - Signal processing- Practical Session 3
# Signal Representation

The practical session will be done in Python 3 and it is strongly recommended to have a working Anaconda environnement. The different sections of the session should be implemented in Jupyter notebooks and it is strongly recommended to take notes in the notebook during the session.

The individual report for the session will be uploaded on moodle in python notebook format. It is expected to have a working code (reproducible figures) and a short discussion in markdown format for all the results obtained in the practicals session. The end of the report must contain a personal discussion about the session (what was hard to understand and implement, how you would do it newt time, what was new, discussion of relation with the course, personal discussion about how to use theses tools in a professional setting, ...).

#### Importing libraries

In this section we will use Numpy/Scipy Python libraries for handling numerical data and Matplotlib for plotting them. We will also need to have access to some function in the `scipy.signal` and `scipy.io.wavfile` submodule that have to be imported also.

```
import numpy as np
import pylab as pl
import scipy as sp
import scipy.signal
import scipy.io.wavfile
```

## 1 Short Time Fourier Transform

### 1.1 Signal interpretation

For all the signals listed below

1. Load the signal in memory (`np.load,sp.io.wavfile.read`).
2. Plot the signal as a function of time and thge absolute value of the the Fourier domain (`pl.plot,np.fft.fft,np.fft.fftfreq`).
3. Compute the short time Fourier transform (STFT) of the signal (`sp.signal.stft`).
4. Visualize the spectrogram (`pl.pcolormesh`) and interpret it by asking yourself the following questions When applicable:
    - What frequencies are present in the signal and what is their evolution?
    - What are the optimal parameters of the STFT (`nperseg,window`) and visualization (`norm=None` or log scaling with `norm=LogNorm()` from `from matplotlib.colors` in `pcolormesh`) for better-interpretability .
    - Can you see different sources in the signal? Is there visible noise in the spectrogram?

- What are the physical constants and information you can extract from the signal (period, fundamental frequency)? To what physical processes and events are they related to?

Here is the list of signal and their corresponding filename. Some have been saved in previous section of the practical session and the others can be downloaded from moodle (`data_TP3.zip`):

- `seq.wav` contains the sequence of note generated in section 1.3 of the first practical session.
- `drum.wav` is a recording of a drum playing containing both bass drums and cymbal corresponding to a low frequency and high frequency signal. Can you separate the different instrument and recognize them?
- `stairway.wav` and `stairwayb.wav` contains 10 seconds of the start of a well known song where the second file has been corrupted by noise.
- `conso.npz` is the recording of of the usage in Watt of the Drahi-X Novation Center building with a sampling period of 1 min for about 4 weeks.
- `205.npz` contains measure of the vibration of the engine of a venerable Peugeot 205 car while your professor plays with the throttle. The engine is a 4-stroke engine with 4 cylinders. What is the maximum RPM reached by the engine in the sequence if it's value in neutral (at the beginning of the sequence) is 750RPM.

### 1.2 Denoising with STFT

In this subsection we will use the spectral subtraction method to filter a recording from the 2010 Soccer World Cup.

1. Load in memory the file `world_cup_2010.wav` and listen to it. Can you hear the vuvuzela?
2. Compute the STFT of the signal and plot it. Can you distinguish the spectrum from the vuvuzela from the voice in the spectrogram?
3. Estimate an average of the PSD of the Vuvuzela from the columns of the spectrogram corresponding to the samples between seconds 3 and 8 of the sequence when only the vuvuzela is heard.
4. Use the Estimation of the PSD to perform spectral subtraction, reconstruct the signal as described in the course (part 4 slide 29) and save it as a wav file.
5. Listen to your reconstruction, can you still hear the vuvuzela? Are there audio artifacts? What is the effect of the size of the window?

## 2 Audio unmixing with Non-negative Matrix factorization

In this section we will use NMF to perform source separation of an audio signal. You will need to have scikit-learn installed so that you can access the module `sklearn`.

1. Load in memory the file `uku_chords.wav` and listen to it. The sequence contain several notes (and chords) played on a ukulele.
2. Compute the STFT of the signal and its spectrogram $\mathbf{S}$ and plot its spectrogram (`pl.pcolormesh` ). Can you distinguish different notes? Use a large window of `nperseg=2048` to see the different harmonics.
3. Compute a Non-negative Matrix factorization with $p = 4$ sources of the spectrogram using the `NMF` from `sklearn.decomposition` to estimate. This can be done with the following steps:
   - Instantiate a model of the `NMF` class : `nmf=NMF(n_components=p)`.

- Fit the model on the data (spectrogram) and get the representation $\mathbf{A}$ with the function `nmf.fit_transform`. Note that the spectrogram has to be given to the function transposed and the returned matrix $\mathbf{A}$ is also transposed (the samples are accessed as line of the matrix in sklearn).
- Get the dictionary $\mathbf{D}$ from the model by accessing it from `nmf.components_`. Keep in mind that it is also transposed.

4. Plot on the same figure (`pl.subplot`) the temporal signal, the spectrogram and the evolution of the representation coefficients $\mathbf{A}$ along time. Do the different sources correspond to individual notes, part of notes?

5. Reconstruct the low rank spectrogram with $\hat{\mathbf{S}} = \mathbf{DA} = \sum_k \mathbf{d}_k \bar{\mathbf{a}}_k$ where $\mathbf{d}_k$ are column matrices and $\bar{\mathbf{a}}_k$ are line matrices for source $k$. Compare the model to the true spectrogram (`pl.pcolormesh`).

6. The reconstruction of the STFT source can be done from the corresponding column $\mathbf{d}_k$ of $\mathbf{D}$ and line $\bar{\mathbf{a}}_k$ of $\mathbf{A}$ for source $k$:
$$\hat{\mathbf{S}}_k = \frac{\mathbf{S}}{|\mathbf{S}|} \mathbf{d}_k \bar{\mathbf{a}}_k$$
Reconstruct the individual sources with the inverse STFT and save them in different wav files.

7. What happens when you change the number of sources $p$? Can one note be represented by a unique source? Can you reconstruct better individual notes by grouping some sources?

8. Bonus question : Perform source separation on files `drum.wav`.