Institut Mines-Télécom

Analysis and synthesis of bell sounds

Roland Badeau

The various files related to this practical work can be downloaded on the eCampus website of TSIA202b. You can load a sound file with Matlab by using the command `[x,Fs] = wavread('clocheB.wav')`. In order to listen to it, you can type up `soundsc(x,Fs)`. In Python, you can use the provided notebook template `template-TP-HR.ipynb`.

1 Introduction

Bells are among the oldest music instruments and the sound they produce is often evocative because it has soothed the everyday life of generations for about 3000 years, accompanying minor and major events. This evocation is partly due to the structure of the sound spectrum: the eigenmodes of vibration are generally tuned by the bell makers so that their frequencies follow a particular series, which includes the minor third (E flat if the bell is tuned in C). This series is not harmonic, but the ratios between the eigenfrequencies are such that one can perceive a well-defined pitch. In particular, the presence of the series 2-3-4, strong at the beginning of the sound, reinforces the feeling of pitch in the neighborhood of the fundamental frequency. This feeling is related to a psychoacoustic effect (processing of the signal received by the brain).

Let f_p be the frequency corresponding to the perceived pitch. The analysis of the eigenfrequencies series leads to a table of about 15 ratios $\alpha_n = f_n/f_p$. Their orders of magnitude are as follows: 0.5 (hum / "*bourdon*"), 1 (prime / fundamental), 1.2 (minor third), 1.5 (fifth), 2 (nominal / octave), 2.5, 2.6, 2.7, 3, 3.3, 3.7, 4.2 (wrong double octave), 4.5, 5, 5.9. The timbre of the corresponding sound depends on the amplitude and on the decrease of each of these partials.

This practical work aims to develop a high resolution spectral estimation method in order to perform the analysis / synthesis of bell sounds. As can be noticed in figure 1, this type of sounds presents a strong temporal decrease.

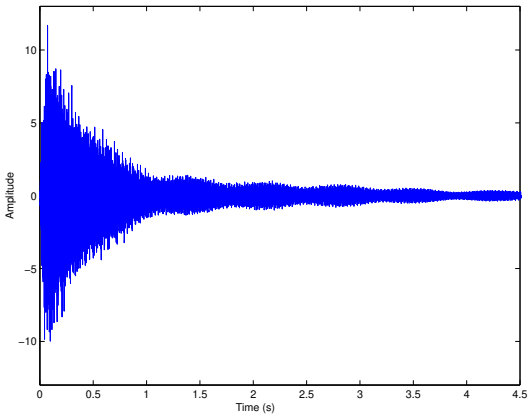

Figure 1: Bell sound

In order to take this damping into account, we consider the *Exponential Sinusoidal Model* (ESM):

$$s[t] = \sum_{k=0}^{K-1} a_k \, e^{\delta_k t} e^{i(2\pi f_k t + \phi_k)},$$

which to each frequency $f_k \in \left]-\frac{1}{2}, \frac{1}{2}\right]$ associates a real amplitude $a_k > 0$, a phase $\phi_k \in]-\pi, \pi]$, and

a damping factor $\delta_k \in \mathbb{R}$. By defining the complex amplitudes $\alpha_k = a_k e^{i\phi_k}$ and the complex poles $z_k = e^{\delta_k + i2\pi f_k}$, this model can be rewritten in the form

$$s[t] = \sum_{k=0}^{K-1} \alpha_k \, {z_k}^t.$$

The case $\delta_k < 0$ corresponds to exponentially decreasing sinusoids, which are solutions of physical propagation equations. The model parameters are then $\{\delta_k, f_k, a_k, \phi_k\}_{k\in\{0\ldots K-1\}}$. In order to estimate them, we will use the ESPRIT method presented in the course. Firstly, we will apply it to a synthetic signal in order to highlight the superiority of high resolution methods over Fourier analysis in terms of spectral resolution. Then this method will be applied to bell sounds.

2 Reminder about Matlab/Python

In Matlab:

- `A'`: Hermitian transpose (conjugate transpose) of matrix $\boldsymbol{A}$;
- `A.'`: transpose of matrix $\boldsymbol{A}$ (without complex conjugation);
- `A(l1:l2,c1:c2)`: matrix extracted from $\boldsymbol{A}$ between rows l_1 and l_2 (inclusive) and columns c_1 and c_2 (inclusive).

In Python:

- `A.conj().T`: Hermitian transpose (conjugate transpose) of matrix $\boldsymbol{A}$;
- `A.T`: transpose of matrix $\boldsymbol{A}$ (without complex conjugation);
- `A[l1:l2,c1:c2]`: matrix extracted from $\boldsymbol{A}$ between rows l_1 and l_2 (excluded) and columns c_1 and c_2 (excluded).

3 Synthetic signal

Here we consider a synthetic signal of length N, consisting of a sum of two complex exponentials, whose frequencies are separated by an interval $\Delta f = \frac{1}{N}$ (which corresponds to the resolution limit of Fourier analysis). The phases are drawn randomly, according to a uniform probability distribution on $(-\pi, \pi)$. We do not add noise to this signal, so that that the observed signal $x[t]$ is equal to $s[t]$. You can use the following parameters: $N = 63$, $f_0 = \frac{1}{4}$, $f_1 = f_0 + \frac{1}{N}$, $a_0 = 1$, $a_1 = 10$, $\delta_0 = 0$, $\delta_1 = -0.05$. In order to synthesize it, you can call the provided function

```
x = Synthesis(N,delta,f,a,phi);
```

whose arguments are N, the vector `delta` of damping factors δ_k, the vector `f` of frequencies f_k, the vector `a` of amplitudes a_k, and the vector `phi` of phases ϕ_k.

3.1 Spectral analysis by Fourier transform

Observe the periodogram of this signal. Briefly study the separability of the two spectral lines, without zero-padding ($N_{\text{fft}} = N$) and with zero-padding ($N_{\text{fft}} = 1024 > N$).

3.2 High resolution methods

Our goal is to write functions

`MUSIC(x,n,K)` and `[delta,f] = ESPRIT(x,n,K)`

which analyze the signal x of length N by using methods MUSIC and ESPRIT, with a signal subspace of dimension K and a noise subspace of dimension $n-K$, and the data vectors of length n ranging from $K+1$ to $N-K+1$. In order to process the synthetic signals, you can choose $n = 32$ and $K = 2$. The two methods share the following steps:

1. **Computation of the empirical covariance matrix**

   The empirical covariance matrix of the observed signal is defined by the equation

   $$\widehat{\boldsymbol{R}}_{xx} = \frac{1}{l}\boldsymbol{X}\,\boldsymbol{X}^H$$

   where $\boldsymbol{X}$ is an $n \times l$ Hankel matrix containing the $N = n + l - 1$ samples of the signal:

   $$\boldsymbol{X} = \begin{bmatrix} x[0] & x[1] & \dots & x[l-1] \\ x[1] & x[2] & \dots & x[l] \\ \vdots & \vdots & \ddots & \vdots \\ x[n-1] & x[n] & \dots & x[N-1] \end{bmatrix}$$

   Matrix $\boldsymbol{X}$ can be constructed with function `hankel`.

2. **Estimation of the signal subspace**

   Matrix $\widehat{\boldsymbol{R}}_{xx}$ can be diagonalized with the command `[U1,Lambda,U2] = svd(Rxx)`. Matrix $\widehat{\boldsymbol{R}}_{xx}$ being positive semidefinite, the column vectors of the $n \times n$ matrices $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ are the eigenvectors of $\widehat{\boldsymbol{R}}_{xx}$, corresponding to the n eigenvalues sorted in the diagonal matrix $\boldsymbol{\Lambda}$ in decreasing order (we thus have $\widehat{\boldsymbol{R}}_{xx} = \boldsymbol{U}_1\,\boldsymbol{\Lambda}\,\boldsymbol{U}_1^H = \boldsymbol{U}_2\,\boldsymbol{\Lambda}\,\boldsymbol{U}_2^H$). Therefore you can extract from $\boldsymbol{U}_1$ (or from $\boldsymbol{U}_2$) an $n \times K$ basis of the signal subspace $\boldsymbol{W}$.

3.2.1 ESPRIT method

In a first stage, the ESPRIT method consists in estimating the frequencies and damping factors:

3. **Estimation of the frequencies and damping factors**

   In order to estimate the frequencies, you can proceed in the following way:

   - extract from $\boldsymbol{W}$ the matrices $\boldsymbol{W}_\downarrow$ (obtained by removing the last row of $\boldsymbol{W}$) and $\boldsymbol{W}_\uparrow$ (obtained by removing the first row of $\boldsymbol{W}$);
   - compute $\boldsymbol{\Phi} = \left(\left(\boldsymbol{W}_\downarrow^H \boldsymbol{W}_\downarrow\right)^{-1} \boldsymbol{W}_\downarrow^H\right) \boldsymbol{W}_\uparrow = \boldsymbol{W}_\downarrow^\dagger \boldsymbol{W}_\uparrow$, where the symbol † denotes the pseudo inverse operator (Matlab function `pinv` or Python function `numpy.linalg.pinv`).
   - compute the eigenvalues of $\boldsymbol{\Phi}$ by using the Matlab function `eig` or the Python function `numpy.linalg.eig` (we remind that the eigenvalues of $\boldsymbol{\Phi}$ are the poles $z_k = e^{\delta_k + i2\pi f_k}$). Then compute $\delta_k = \ln(|z_k|)$ and $f_k = \frac{1}{2\pi}\,\mathrm{angle}(z_k)$.

4. **Estimation of the amplitudes and phases**

   We now aim to write a function

   ```
   [a,phi] = LeastSquares(x,delta,f)
   ```

   which estimates the amplitudes a_k and phases ϕ_k by means of the least squares method, given the signal x, the damping factors δ_k and the frequencies f_k. The complex amplitudes are thus determined by the equation

   $$\boldsymbol{\alpha} = \left((\boldsymbol{V}^{N\,H} \boldsymbol{V}^N)^{-1} \boldsymbol{V}^{N\,H} \right) \boldsymbol{x} = \boldsymbol{V}^{N\,\dagger}\, \boldsymbol{x} \tag{1}$$

   where $\boldsymbol{x}$ is the vector $[x[0], \ldots, x[N-1]]^\top$ and $\boldsymbol{V}^N$ is the $N \times K$ Vandermonde matrix, whose entries are such that $\boldsymbol{V}^N_{(t,k)} = {z_k}^t$ for all $(t,k) \in \{0 \ldots N-1\} \times \{0 \ldots K-1\}$. In order to compute matrix $\boldsymbol{V}^N$, it is possible of avoid using a `for` loop by noting that $\ln\left(\boldsymbol{V}^N_{(t,k)}\right) = t(\delta_k + i2\pi f_k)$. Therefore, the matrix containing the coefficients $\ln\left(\boldsymbol{V}^N_{(t,k)}\right)$ can be expressed as the product of a column vector and a row vector. Then compute $a_k = |\alpha_k|$ and $\phi_k = \text{angle}(\alpha_k)$ for all $k \in \{0 \ldots K-1\}$.

5. **Application to synthetic signals**

   Apply functions `ESPRIT` and `LeastSquares` to the previously synthesized signal. Comment.

3.2.2 MUSIC method

We remind that the MUSIC pseudo-spectrum is defined as $P(z) = \dfrac{1}{\|\boldsymbol{W}_\perp^H \boldsymbol{v}^n(z)\|^2}$, where matrix $\boldsymbol{W}_\perp$ spans the noise subspace.

6. **MUSIC pseudo-spectrum**

   Write a function `MUSIC(x,n,K)` which plots the logarithm of the pseudo-spectrum as a function of the two variables $f \in [0,1]$ and $\delta \in [-0.1, 0.1]$ (you can use the Matlab function `surf` or the Python Matplotlib function `plot_surface`). Apply function `MUSIC` to the previously synthesized signal, and check that the pseudo-spectrum makes the two poles $z_k = e^{\delta_k + i2\pi f_k}$ clearly visible.

4 Audio signals

We now propose to apply the functions developed in the previous part to bell sounds.

4.1 Spectral analysis by Fourier transform

Look at the periodogram of the signal `ClocheA.wav` or `ClocheB.wav`, and compare the series of its eigenfrequencies with the values given in the introduction.

4.2 High resolution method

We now want to apply the ESPRIT method to this signal. Let $K = 54$, $n = 512$ and $l = 2n = 1024$ (hence $N = n + l - 1 = 1535$).

In order to guarantee that the signal model holds on the analysis window (exponential damping), we will extract a segment of length N whose beginning is posterior to the maximum of the waveform envelope. We may thus start at the 10000^{th} sample.

Apply function `ESPRIT` to the extracted signal in order to estimate the eigenfrequencies and the corresponding damping factors. Then estimate the amplitudes and phases by calling function `LeastSquares`. Finally, listen to the signal resynthesized with function `Synthesis` (on a duration longer than the extracted segment, in order to clearly highlight the sound resonances), and comment.