DMUSIC Algorithm

Pedro Pons-Villalonga, 2021

1 Singular Value Decomposition

A matrix $M \in \mathbb{C}^{m\times n}$ can be factorized as:

$$M = USV^* \tag{1}$$

Where:

- $U \in \mathbb{C}^{m\times m}$ is unitary.
- $S \in \mathbb{R}^{m\times n}$ is a rectangular diagonal matrix of positive real numbers.
- $V \in \mathbb{C}^{n\times n}$ is unitary.
- V^* denotes the complex conjugate of V.

The elements of the diagonal of S are called *singular values*, and are usually sorted in descending order (important). The columns of U and V are orthonormal bases, and we can say that the matrix M maps the vectors $\mathbf{v}_i$ to the stretched unit vector $\sigma_i \mathbf{u}_i$, where σ_i is the i-th singular value (i-th element of the diagonal of S), as:

$$MV = USV^*V = US \tag{2}$$

2 DMUSIC

Consider a signal $y(t)$, that we assume can be written as a sum of K harmonics and white noise $W(t)$:

$$y(t) = \sum_{k=1}^{\mathrm{K}} A_k e^{-a_k t + i\omega_k t} + W(t) \equiv \sum_{k=1}^{\mathrm{K}} A_k e^{s_k t} + W(t) \tag{3}$$

Or, for a digitized signal:

$$y(n) = \sum_{k=1}^{\mathrm{K}} A_k e^{s_k \tau n} + W(n) \tag{4}$$

Where $\tau = 1/F_{\mathrm{s}}$, the inverse of the sampling frequency, is the time interval between digitizations.

We want to recover from the signal the frequencies ω_k and damping rates a_k. For that, if the signal is digitized, we make the assumption that we can predict the signal in an interval of length N using the preceding J (> K) values:

$$y(n) = \sum_{i}^{\mathrm{J}} c_{(\mathrm{J}-i)}\; y(n-i) \tag{5}$$

We can build a *prediction matrix* $A \in \mathbb{R}^{(N-J)\times J}$, so that:

$$\begin{pmatrix} y(1) & y(2) & \cdots & y(J) \\ y(2) & y(3) & \cdots & \\ \vdots & & \ddots & \\ y(N-J) & \cdots & & y(N) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_J \end{pmatrix} = \begin{pmatrix} y(J+1) \\ y(J+2) \\ \vdots \\ y(N) \end{pmatrix} \tag{6}$$

The matrix A is a Hankel matrix. If the noise was zero, the matrix A can be expressed as a Vandermonde decomposition:

$$A = S_L C S_R^* \tag{7}$$

Where $C \in \mathbb{C}^{K\times K}$ is a diagonal matrix, and the k-th columns of the matrices $S_L, \in \mathbb{C}^{(N-J)\times K}$ and $S_R \in \mathbb{C}^{J\times K}$ are given by the so-called *left* and *right signal vectors*:

$$\vec{r_L}(s_k) = \begin{pmatrix} 1 & e^{s_k\tau} & e^{2s_k\tau} & \cdots & e^{(N-J-1)s_k\tau} \end{pmatrix}^T \tag{8}$$

$$\vec{r_R}(s_k) = \begin{pmatrix} 1 & e^{s_k\tau} & e^{2s_k\tau} & \cdots & e^{(J-1)s_k\tau} \end{pmatrix}^T \tag{9}$$

From the right signal vectors we can define the *signal space:*

$$\Omega^S \equiv \operatorname{span}\left(\{\vec{r}_R(s_k)\}\right) \tag{10}$$

and the *null space:*

$$\Omega^N \equiv \ker(A) \tag{11}$$

We can perform a singular value decomposition on matrix A, obtaining:

$$A = U S V^* \tag{12}$$

In this case, $U \in \mathbb{C}^{(N-J)\times(N-J)}$, $V \in \mathbb{C}^{J\times J}$ are square matrices and $S \in \mathbb{R}^{(N-J)\times J}$ is a diagonal rectangular matrix of real, positive numbers.

In the absence of noise, the rank of S is K, and only the first K singular values (elements of the diagonal of S) are non-zero. The first K columns of U and V, then, contain all the information from the signal, and the first K columns of V span the signal space Ω^S. Consequently, as the columns $\{K+1, \cdots, J\}$ of V correspond with null singular values, they belong to the null space.

If there is noise, the separation between signal space and null space (or, more appropriately, *noise space*) becomes less clear. However, if the signal-to-noise ratio is sufficiently high, we can safely assume that the information of the signal remains in the first K columns of U and V.

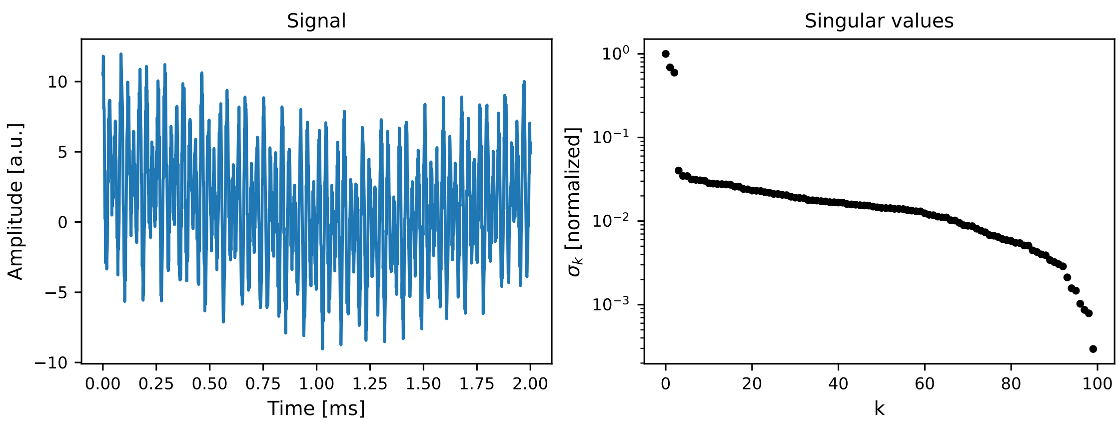

Figure 1: Synthetic (noisy) signal and singular values of the SVD decomposition of its A matrix. N=200, K=3.

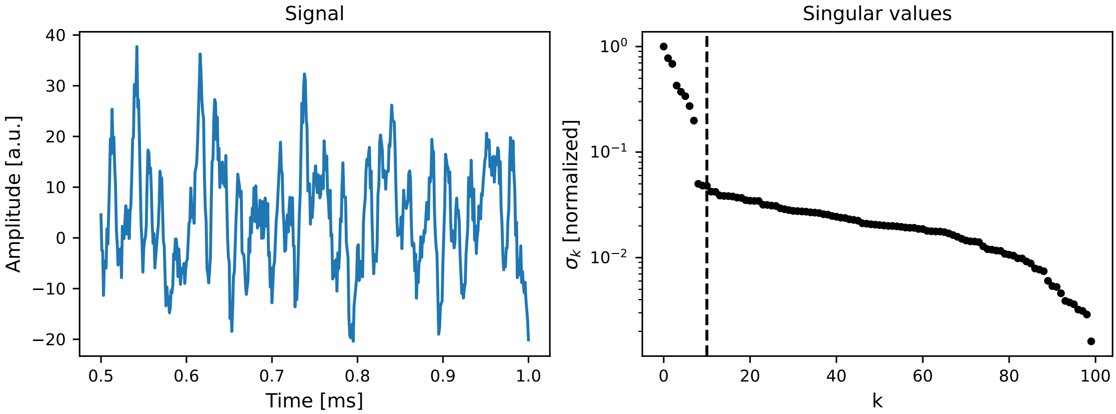

Figure 2: Synthetic (noisy) signal and singular values of its SVD decomposition. N=200, K=10. The separation between signal and noise space is not that clear, maybe in part because the amplitudes of some harmonics are small.

We want, given a real signal (of which we do not know the number of harmonics K) to estimate its dominant frequencies and corresponding damping rates. For that, we need a good estimate of K, which can be obtained by looking at the SVD decomposition of the signal. Lower values of K can misidentify signal components as noise, and viceversa for higher values of K.

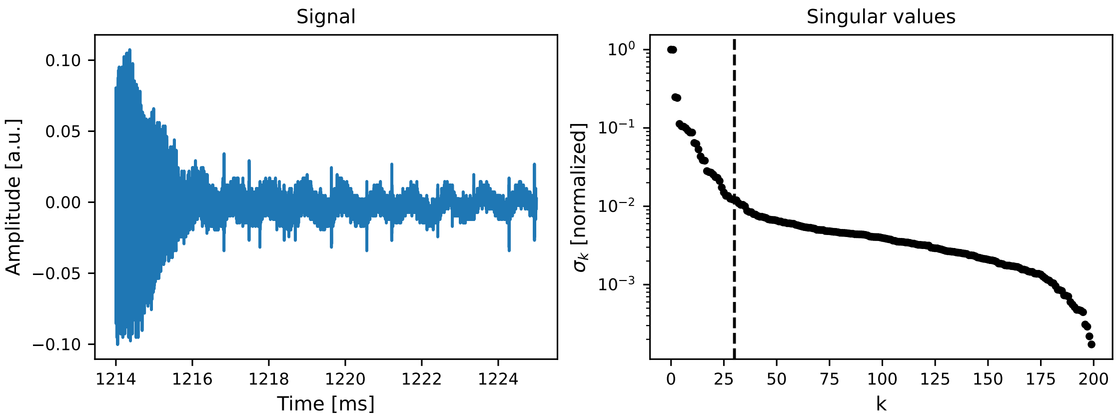

Figure 3: Real signal and singular values. Vertical line at K=30 shows the justification for a commonly used value of K. N=400.

We can define a new matrix, V_N, given by the columns $(K+1,\cdots,J)$ of V. Then, all vectors $\vec{r}(s) \in \mathbb{C}^J$:

$$\vec{r}(s) = \left(1,\ e^{s\tau},\ e^{2s\tau},\ \cdots,\ e^{(J-1)s\tau}\right) \tag{13}$$

that verify:

$$\left\| V_N \vec{r}(s) \right\| = 0 \tag{14}$$

Will be orthogonal to the noise space and thus belong to the signal space. In the presence of noise, the true equality is not necessarily satisfied. However, the function:

$$P(s) \equiv \frac{1}{\left\| V_N \vec{r}(s) \right\|} \tag{15}$$

will still present maxima when $s \simeq s_k$, so we can use it to estimate the spectrum of the signal. To do that, we must perform a scan over a suitable grid. As we are interested only in the frequencies of the maxima, we integrate over the real part of the grid to get the frequency estimator:

$$p(\omega) = \int_{a_0}^{a_1} P(-a + i\omega)\, da \tag{16}$$

3 Implementation

A simple, reasonably fast implementation has been made with Python and `MATLAB`. We must realize that the bulk of the computational effort will be spent in the matrix multiplication $V_N \vec{r}(s)$. Python is relatively slow with loops, but the matrix multiplication routines of the package `numpy` are optimized. Because of that, the optimal solution consists in the creation of a big matrix R of all the $\vec{r}(s)$, that will be multiplied by V_N only once. This multiplication is done in parallel automatically.

Now, the implementation (a bit convoluted):

Algorithm 1 DMUSIC spectrogram

procedure DMUSIC_SPEC(t, y, N, K, overlap) ▷ t: time array, y: raw signal
  J ← N//2 ▷ For performance
  f_{arr} ← linspace(0, F_{nyq}, n_f) ▷ nf: number of frequencies to scan (typ. ~ 500)
  a_{arr} ← linspace(0, -1, n_a) ▷ na: number of damping factors to scan (typ. ~ 50)
  R ← Empty matrix $\in \mathbb{C}^{J\times(n_f \cdot n_a)}$ ▷ This way it is only created once
  for f_j ← w_arr **do**
    for a_j ← a_arr **do**
      s ← a_i + i f_i
      Make $\vec{r}(s)$ (normalized)
      Append $\vec{r}(s)$ to R
    end for
  end for
  Find nsteps
  P_f ← Empty matrix $\in \mathbb{R}^{\text{nsteps}\times n_f}$
  i ← 0
  while i < len(y) **do**
    ym ← Hankel(`y[i:i+N]`)
    U, S, V_T ← svd(ym)
    V_N ← V_T`[K:(N-J)]` ▷ Last rows of V_T, starting at K
    P_s ← norm($V_N R$) ▷ $P_s \in \mathbb{R}^{n_f \cdot n_a}$
    i ← i + (1 − overlap)·N
    Append to P_f ← Integral of P_s over a_{arr} in intervals of length n_a
  end while
  P_f ← $10 \cdot \log_{10}(P_f/\max(P_f))$
end procedure

References

[1] Y. Li, J. Razavilar, and K. R. Liu, “A high-resolution technique for multidimensional NMR spectroscopy,” *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 1, pp. 78–86, 1998.

[2] R. Kleiber, M. Borchardt, A. Könies, and C. Slaby, “Modern methods of signal processing applied to gyrokinetic simulations,” *Plasma Physics and Controlled Fusion*, vol. 63, no. 3, p. 035017, 2021.

[3] W. Liao and A. Fannjiang, “MUSIC for single-snapshot spectral estimation: Stability and super-resolution,” *Applied and Computational Harmonic Analysis*, vol. 40, no. 1, pp. 33–67, 2016.