Generalization of ESPRIT algorithm

Lei Zhang

October 2, 2025

The generalization of ESPRIT to matrix-valued systems follows a similar idea presented in Ref. [1] as a matrix-valued generalization of the scalar-valued Prony's method of Ref. [2]. The current notes provide the detailed mathematical derivation of this algorithm.

The objective is to approximate a matrix-valued function using a linear combination of shared exponentials, based on its samples on a uniform grid. For simplicity, we flatten the $n_{\mathrm{orb}} \times n_{\mathrm{orb}}$ matrices into column vectors of length n_{orb}^2 (denoted by $\rightarrow$) and express the approximation as:

$$\vec{y}_k = \vec{x}_k + \vec{n}_k \approx \sum_{i=1}^{M} \vec{R}_i z_i^k + \vec{n}_k \text{ for } k = 0, 1, \cdots, N-1 \ , \tag{1}$$

where $\vec{y}$ is the sampled signal, $\vec{x}$ is the exact signal, $\vec{n}$ is the noise, z_i is the i-th exponential shared by all matrix elements, and $\vec{R}_i$ are the corresponding weights.

1 Noise-free case

For noise-free case, i.e., $\vec{n} \equiv 0$, define the matrix

$$Y = \begin{bmatrix} \vec{y}_0 & \vec{y}_1 & \cdots & \vec{y}_L \\ \vec{y}_1 & \vec{y}_2 & \cdots & \vec{y}_{L+1} \\ \vdots & \vdots & \ddots & \vdots \\ \vec{y}_{N-L-1} & \vec{y}_{N-L} & \cdots & \vec{y}_{N-1} \end{bmatrix}_{n_{\mathrm{orb}}^2 (N-L)\times(L+1)} . \tag{2}$$

If we extract two submatrices from Y by deleting the last and the first column, respectively (note that in the following formula, the Python convention is used; for example, $A[:, a:b]$ represents the submatrix of A consisting of columns with indices $a, a+1, \ldots, b-1$):

$$Y_1 = Y[:, 0:L] = \begin{bmatrix} \vec{y}_0 & \vec{y}_1 & \cdots & \vec{y}_{L-1} \\ \vec{y}_1 & \vec{y}_2 & \cdots & \vec{y}_L \\ \vdots & \vdots & \ddots & \vdots \\ \vec{y}_{N-L-1} & \vec{y}_{N-L} & \cdots & \vec{y}_{N-2} \end{bmatrix}_{n_{\mathrm{orb}}^2 (N-L)\times L} , \tag{3}$$

$$Y_2 = Y[:, 1:(L+1)] = \begin{bmatrix} \vec{y}_1 & \vec{y}_2 & \cdots & \vec{y}_L \\ \vec{y}_2 & \vec{y}_3 & \cdots & \vec{y}_{L+1} \\ \vdots & \vdots & \ddots & \vdots \\ \vec{y}_{N-L} & \vec{y}_{N-L+1} & \cdots & \vec{y}_{N-1} \end{bmatrix}_{n_{\text{orb}}^2(N-L)\times L}, \tag{4}$$

then it can be verified that

$$Y_1 = Z_1 R Z_2 \tag{5}$$

$$Y_2 = Z_1 R Z_0 Z_2 \tag{6}$$

with the following definition:

$$Z_1 = \begin{bmatrix} \mathbb{I} & \mathbb{I} & \cdots & \mathbb{I} \\ z_1\mathbb{I} & z_2\mathbb{I} & \cdots & z_M\mathbb{I} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{N-L-1}\mathbb{I} & z_2^{N-L-1}\mathbb{I} & \cdots & z_M^{N-L-1}\mathbb{I} \end{bmatrix}_{n_{\text{orb}}^2(N-L)\times n_{\text{orb}}^2 M}, \tag{7}$$

$$R = \begin{bmatrix} \vec{R}_1 & & & \\ & \vec{R}_2 & & \\ & & \ddots & \\ & & & \vec{R}_M \end{bmatrix}_{n_{\text{orb}}^2 M\times M}, \tag{8}$$

$$Z_0 = \begin{bmatrix} z_1 & & & \\ & z_2 & & \\ & & \ddots & \\ & & & z_M \end{bmatrix}_{M\times M}, \tag{9}$$

$$Z_2 = \begin{bmatrix} 1 & z_1 & \cdots & z_1^{L-1} \\ 1 & z_2 & \cdots & z_2^{L-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & z_M & \cdots & z_M^{L-1} \end{bmatrix}_{M\times L}, \tag{10}$$

where $\mathbb{I}$ in Eq. (7) denotes an $n_{\text{orb}}^2 \times n_{\text{orb}}^2$ identity matrix.

As a result, we have

$$Y_2 - \lambda Y_1 = Z_1 R (Z_0 - \lambda \mathbb{I}_{M\times M}) Z_2, \tag{11}$$

so that solving for z_i is equivalent to solving the ordinary eigenvalue problem:

$$Y_1^{\dagger} Y_2 - \lambda \mathbb{I}\,, \tag{12}$$

where $\dagger$ represents the pseudo-inverse.

2 Noisy case

In the noisy case, we apply Singular Value Decomposition (SVD) to the matrix Y as follows:

$$Y = U\Sigma V^H \ , \tag{13}$$

where H denotes the Hermitian transpose, U is an $n_{\text{orb}}^2(N-L) \times n_{\text{orb}}^2(N-L)$ matrix, $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{L+1})$ is an $n_{\text{orb}}^2(N-L) \times (L+1)$ matrix, and $V = [v_1, \ldots, v_{L+1}]$ is an $(L+1) \times (L+1)$ matrix. To filter the noise, we do the truncation at

$$\sigma_{M+1}/\sigma_1 \approx \text{noise level} \ . \tag{14}$$

Thus, we have:

$$Y_1 = U\Sigma' V_1'^H \tag{15}$$

$$Y_2 = U\Sigma' V_2'^H \ , \tag{16}$$

where $\Sigma' = \text{diag}(\sigma_1, \sigma_2, \cdots, \sigma_M)$ is an $n_{\text{orb}}^2(N-L) \times M$ matrix, and V_1' and V_2' are obtained from $V' = [v_1, v_2, \cdots, v_M]$ by deleting the last and first row, respectively. Specifically, $V_1' = V'[0:L, :]$ and $V_2' = V'[1:(L{+}1), :]$. As a result, solving the eigenvalues of $Y_2 - \lambda Y_1$ is equivalent to solving the eigenvalues of $V_2'^H - \lambda V_1'^H$, which ultimately reduces to solving the eigenvalues of:

$$V_2'^H (V_1'^H)^\dagger - \lambda \mathbb{I}_{M\times M} \ . \tag{17}$$

After extracting the nodes $z_1, \ldots, z_M$ from Eq. (17), the corresponding weights can be computed by:

$$\begin{bmatrix} \vec{y}_0^T \\ \vec{y}_1^T \\ \vdots \\ \vec{y}_{N-1}^T \end{bmatrix}_{N\times n_{\text{orb}}^2} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ z_1 & z_2 & \cdots & z_M \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{N-1} & z_2^{N-1} & \cdots & z_M^{N-1} \end{bmatrix}_{N\times M} \begin{bmatrix} \vec{R}_1^T \\ \vec{R}_2^T \\ \vdots \\ \vec{R}_M^T \end{bmatrix}_{M\times n_{\text{orb}}^2} . \tag{18}$$

3 Algorithm

To summarize, the algorithm for ESPRIT works as follows:

1. Construct the matrix as shown in Eq. (2).
2. Perform SVD as described in Eq. (13).
3. Determine the truncation as in Eq. (14).
4. Construct $V_2'^H (V_1'^H)^\dagger$ and find its eigenvalues $\{z_1, \ldots, z_M\}$ according to Eq. (17).
5. Solve Eq. (18) to obtain the matrix-valued weights.

References

[1] Lexing Ying. Pole recovery from noisy data on imaginary axis. Journal of Scientific Computing, 92(3):107, 2022.

[2] Lexing Ying. Analytic continuation from limited noisy matsubara data. Journal of Computational Physics, 469:111549, 2022.