Fast ESPRIT algorithms based on partial singular value decompositions

Daniel Potts

Technische Universität Chemnitz, Department of Mathematics, D–09107 Chemnitz, Germany

Manfred Tasche

University of Rostock, Institute of Mathematics, D–18051 Rostock, Germany

Abstract

Let $h(x)$ be a nonincreasing exponential sum of order M. For N given noisy sampled values $h_n = h(n) + e_n$ $(n = 0, \dots, N-1)$ with error terms e_n, all parameters of $h(x)$ can be estimated by the known ESPRIT (Estimation of Signal Parameters via Rotational Invariance Techniques) method. The ESPRIT method is based on singular value decomposition (SVD) of the L-trajectory matrix $(h_{\ell+m})_{\ell,m=0}^{L-1,N-L}$, where the window length L fulfills $M \le L \le N - M + 1$. The computational cost of the ESPRIT algorithm is dominated by the cost of SVD. In the case $L \approx \frac{N}{2}$, the ESPRIT algorithm based on complete SVD costs about $\frac{21}{8}\, N^3 + M^2(21\, N + \frac{91}{3}\, M)$ operations. Here we show that the ESPRIT algorithm based on partial SVD and fast Hankel matrix-vector multiplications has much lower cost. Especially for $L \approx \frac{N}{2}$, the ESPRIT algorithm based on partial Lanczos bidiagonalization with S steps requires only about $18\, SN \log_2 N + S^2(20N + 30S) + M^2(N + \frac{1}{3}M)$ operations, where $M \le S \le N - L + 1$. Numerical experiments demonstrate the high performance of these fast ESPRIT algorithms for noisy sampled data with relatively large error terms.

Keywords: ESPRIT algorithm, exponential sum, rectangular Hankel matrix, partial singular value decomposition, partial Lanczos bidiagonalization, computational cost.

AMS *Subject Classifications*: 65F15, 65F20, 94A12.

1. Introduction

Let $M \geq 1$ be an integer, $\alpha > 0$, $f_j \in [-\alpha, 0] + \mathrm{i}\,[-\pi, \pi)$ $(j = 1, \ldots, M)$ be distinct complex numbers and $c_j \in \mathbb{C} \setminus \{0\}$ $(j = 1, \ldots, M)$. We consider the nonincreasing exponential sum of order M

$$h(x) := \sum_{j=1}^{M} c_j \, \mathrm{e}^{f_j x} \quad (x \geq 0). \tag{1.1}$$

The recovery of the complex numbers f_j, the complex coefficients c_j, and the order M in the exponential sum (1.1) from noisy sampled values $h_k := h(k) + e_k$ $(k = 0, \ldots, N-1)$, where e_k are small error terms, is known as parameter identification problem, see e.g. [21, 22, 11]. Recently, G. Plonka and T. Peter [23] generalized the Prony method to reconstruct M-sparse expansions of generalized eigenfunctions of a linear operator from $\mathcal{O}(M)$ suitable values in a deterministic way. This method includes the parameter identification problem [27, 3, 25], the multivariate polynomial interpolation problem [2], and sparse polynomial interpolation problems in various polynomial bases [7, 26]. The main drawback of the Prony method is that it may be unstable in some cases. A simple but powerful method is to use more sampled values in combination with stable numerical methods. From the numerical point of view, this problem can be solved by the matrix pencil method [17, 9, 25]. However the main disadvantage is the high computational cost of the matrix pencil method.

In this paper we assume that a convenient upper bound for the unknown order M is available. In order to improve the numerical stability, we sample the function (1.1) on N equidistant nodes with $N \geq 2M$. Then we introduce the L-trajectory matrix

$$\boldsymbol{H}_{L,K} := \big(h_{\ell+m}\big)_{\ell,m=0}^{L-1,K-1} \in \mathbb{C}^{L \times K} \tag{1.2}$$

with the *window length* $L \in \{M, \ldots, N - M + 1\}$ and $K := N + 1 - L$. Note that L and K are upper bounds of M. An essential problem of the established ESPRIT method is the high computational cost of $\mathcal{O}(N^3)$ operations for the complete SVD of (1.2), see [8]. Based on the unified approach in [25], we show that iterative methods, such as the partial SVD or partial Lanczos bidiagonalization (LBD) of (1.2), lead to much faster algorithms.

Email addresses: potts@mathematik.tu-chemnitz.de (Daniel Potts), manfred.tasche@uni-rostock.de (Manfred Tasche)

More precisely, we develop an algorithm, such that the oversampling of the function (1.1) with $N > 2M$, which is very important for a good numerical stability, does not destroy the low computational cost. The aim in this paper is an improvement of our results in [25], such that we can solve the parameter identification problem from a splitting, see Section 5, which is computed by partial LBD. To this end, we apply the LBD algorithm developed by J. Baglama and L. Reichel [1] in combination with a fast Hankel matrix-vector multiplication. The idea of fast Hankel matrix-vector multiplication via fast Fourier transform (FFT) is known, see [5, 19]. But in [5, 19], it is assumed that there exist an FFT of length $N = L + K - 1$. Here we describe the fast Hankel matrix-vector multiplication without restrictions. This approach reduces the computational cost of the SVD of Hankel matrix and is known as fast Hankel SVD, see [5, 19]. The main result in this paper is the combination of the fast Hankel SVD with the matrix pencil method. We simplify the matrix pencil method in [25] by the use of partial SVD resp. partial LBD and fast Hankel matrix-vector multiplications.

In order to present a relatively self-contained paper, we structure our work as follows: In the introductory Section 2 we summarize some basic results on partial SVD. We point out the main ingredients of the partial LBD and describe a fast Hankel matrix-vector multiplication without restrictive assumptions. In Section 3 we determine the rank of the L-trajectory matrix (1.2) with exact sampled data (see Lemma 3.1) and the numerical rank of the L-trajectory matrix (1.2) with noisy sampled data (see Lemma 3.4). Thus the order of the exponential sum (1.1) can be determined by the (numerical) rank of the L-trajectory matrix (1.2), if the window length L fulfills $M \leq L \leq N - M + 1$. In Sections 4 and 5, we present the main results, namely the SVD-based ESPRIT Algorithm 4.2 and the LBD-based ESPRIT Algorithm 5.1. In both algorithms we pass on the computation of a Moore-Penrose pseudoinverse (cf. [25]). The computational costs of Algorithms 4.2 and 5.1 are analyzed in detail. In Section 6, we apply our methods to various parameter identification problems and show that the new algorithms behave similar to the algorithms in [5, 19]. In the numerical tests with Algorithm 5.1 we prefer the LBD algorithm of J. Baglama and L. Reichel [1], since this method is very robust for noisy sampled data too. Furthermore we apply the new algorithms to problems with large N.

In this paper we use standard notations. The linear space of all column vectors with L complex components is denoted by $\mathbb{C}^L$, where $\mathbf{0}$ is the corresponding zero vector. The standard basis of $\mathbb{C}^L$ is denoted by $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_L\}$ in which $\boldsymbol{e}_\ell \in \mathbb{C}^L$ $(\ell = 1, \ldots, L)$ has one as its ℓth component and zeros elsewhere.

The linear space of all complex $L \times K$ matrices is denoted by $\mathbb{C}^{L\times K}$. Analogously, $\mathbb{R}^{L\times K}$ is the set of all real $L \times K$ matrices. For a matrix $\boldsymbol{A}_{L,K} \in \mathbb{C}^{L\times K}$, its transpose is denoted by $\boldsymbol{A}_{L,K}^{\mathrm{T}}$, its conjugate transpose is $\boldsymbol{A}_{L,K}^{*}$, and its Moore–Penrose pseudoinverse is $\boldsymbol{A}_{L,K}^{\dagger}$. A square matrix of size $L \times L$ is abbreviated by $\boldsymbol{A}_L$. By $\boldsymbol{I}_L$ we denote the $L \times L$ identity matrix. The null space and the range of a matrix $\boldsymbol{A}_{L,K}$ are denoted by $\mathcal{N}(\boldsymbol{A}_{L,K})$ and $\mathcal{R}(\boldsymbol{A}_{L,K})$, respectively. The spectral norm of $\boldsymbol{A}_{L,K}$ is $\|\boldsymbol{A}_{L,K}\|_2$ and its Frobenius norm is $\|\boldsymbol{A}_{L,K}\|_F$. Further we use the known submatrix notation. Thus $\boldsymbol{A}_{L,K}(1:L,\, 2:K)$ is the submatrix of $\boldsymbol{A}_{L,K}$ obtained by extracting rows 1 though L and columns 2 though K. Note that the first row or column of $\boldsymbol{A}_{L,K}$ can be indexed by zero.
The computational cost of an algorithm is measured in the number of arithmetical operations (such as $+$, $-$, $\times$, $\backslash$, $\sqrt{\cdot}$, etc.), where all operations are counted equally (see [12, p. 27]). When complex arithmetic is involved, one has to count the operations on complex numbers. Often the computational cost of an algorithm is reduced to the leading term, i.e., all lower order terms are omitted. Definitions are indicated by the symbol $:=$. Other notations are introduced when needed.

2. Partial singular value decompositions

Let L, $K \in \mathbb{N}$ with $L \geq K > 1$ be given. We consider a matrix $\boldsymbol{A}_{L,K} \in \mathbb{C}^{L\times K}$ with $\operatorname{rank} \boldsymbol{A}_{L,K} = R \leq N$. Then the *singular value decomposition* (SVD) of $\boldsymbol{A}_{L,K}$ (see [10, p. 70] or [16, pp. 449 – 450]) reads as follows

$$\boldsymbol{A}_{L,K} = \boldsymbol{U}_L \, \boldsymbol{D}_{L,K} \, \boldsymbol{W}_K^{*}\,, \tag{2.1}$$

where

$$\begin{aligned} \boldsymbol{U}_L &= (\boldsymbol{u}_1\,,\, \boldsymbol{u}_2\,,\, \ldots\,,\, \boldsymbol{u}_L) \in \mathbb{C}^{L\times L}\,, \\ \boldsymbol{W}_K &= (\boldsymbol{w}_1\,,\, \boldsymbol{w}_2\,,\, \ldots\,,\, \boldsymbol{w}_K) \in \mathbb{C}^{K\times K} \end{aligned}$$

are both unitary matrices and where $\boldsymbol{D}_{L,K} = \operatorname{diag}(\sigma_j)_{j=1}^{K} \in \mathbb{R}^{L\times K}$ is a diagonal matrix with ordered diagonal entries:

$$\sigma_1 \geq \ldots \geq \sigma_R > \sigma_{R+1} = \ldots = \sigma_K = 0\,.$$

From (2.1) it follows immediately that the rank-R matrix $\boldsymbol{A}_{L,K}$ can be decomposed into a sum of R rank-1 matrices:

$$\boldsymbol{A}_{L,K} = \sum_{j=1}^{R} \sigma_j \, \boldsymbol{u}_j \, \boldsymbol{w}_j^{*}\,.$$

The nonnegative numbers σ_j $(j = 1, \ldots, K)$ are called the *singular values* of $\boldsymbol{A}_{L,K}$. The columns of $\boldsymbol{U}_L$ and $\boldsymbol{W}_K$ are called *left* and *right singular vectors* of $\boldsymbol{A}_{L,K}$ which satisfy

$$\boldsymbol{A}_{L,K}\,\boldsymbol{w}_j = \sigma_j\,\boldsymbol{u}_j\,, \quad \boldsymbol{A}_{L,K}^*\,\boldsymbol{u}_j = \sigma_j\,\boldsymbol{w}_j \quad (j = 1, \ldots, R)$$

and

$$\begin{aligned} \boldsymbol{A}_{L,K}\,\boldsymbol{w}_j &= \boldsymbol{0} \quad (j = R+1, \ldots, K)\,, \\ \boldsymbol{A}_{L,K}^*\,\boldsymbol{u}_j &= \boldsymbol{0} \quad (j = R+1, \ldots, L)\,. \end{aligned}$$

With the submatrices

$$\begin{aligned} \boldsymbol{U}_{L,R} &:= \boldsymbol{U}_L(1:L, 1:R) = (\boldsymbol{u}_1\,,\, \boldsymbol{u}_2\,,\, \ldots\,,\, \boldsymbol{u}_R) \in \mathbb{C}^{L\times R}\,, \\ \boldsymbol{W}_{K,R} &:= \boldsymbol{W}_K(1:K, 1:R) = (\boldsymbol{w}_1\,,\, \boldsymbol{w}_2\,,\, \ldots\,,\, \boldsymbol{w}_R) \in \mathbb{C}^{K\times R}\,, \\ \boldsymbol{D}_R &:= \boldsymbol{D}_{L,K}(1:R, 1:R) = \operatorname{diag}(\sigma_j)_{j=1}^R \in \mathbb{R}^{R\times R}\,, \end{aligned}$$

we obtain the *reduced* SVD of $\boldsymbol{A}_{L,K} = \boldsymbol{U}_{L,R}\,\boldsymbol{D}_R\,\boldsymbol{W}_{K,R}^*$. Note that both matrices $\boldsymbol{U}_{L,R}$ and $\boldsymbol{W}_{K,R}$ possess orthonormal columns.

The SVD plays an important role in the study of matrix approximation problems. Here we consider a low-rank approximation of $\boldsymbol{A}_{L,K}$, i.e., we approximate the given matrix $\boldsymbol{A}_{L,K}$ by another matrix $\boldsymbol{X}_{L,K} \in \mathbb{C}^{L\times K}$ with lower rank S $(S \le R = \operatorname{rank}\boldsymbol{A}_{L,K})$, i.e.

$$\text{minimize } \|\boldsymbol{A}_{L,K} - \boldsymbol{X}_{L,K}\| \quad \text{subject to } \operatorname{rank}\boldsymbol{X}_{L,K} = S$$

with respect to a unitarily invariant norm $\|\cdot\|$. By the Theorem of Eckart–Young–Mirsky (see [10, pp. 72 – 74]), a best rank-S approximation to the given matrix $\boldsymbol{A}_{L,K} \in \mathbb{C}^{L\times K}$ with $\operatorname{rank}\boldsymbol{A}_{L,K} = R$ is

$$\boldsymbol{X}_{L,K} = \boldsymbol{A}_{L,K}^{(S)} := \sum_{j=1}^{S} \sigma_j\,\boldsymbol{u}_j\,\boldsymbol{w}_j^*\,. \tag{2.2}$$

The matrix decomposition (2.2) is called a rank-S *partial* SVD of $\boldsymbol{A}_{L,K}$.

Remark 2.1 When $\boldsymbol{A}_{L,K}$ is a matrix of large size, the computation of the complete SVD (2.1) is very costly. Thus the Golub–Reinsch SVD algorithm requires $4\,L^2\,K + 8\,L\,K^2 + 9\,K^3$ operations (see [10, pp. 253 – 254]). In the case $L \approx K$, this SVD algorithm costs $21\,L^3$ operations. If $\boldsymbol{A}_L$ is Hermitian, then its spectral decomposition reads $\boldsymbol{A}_L = \boldsymbol{U}_L\,\boldsymbol{D}_L\,\boldsymbol{U}_L^*$ and costs $9\,L^3$ operations for the computation of the unitary matrix $\boldsymbol{U}_L$ and the diagonal matrix $\boldsymbol{D}_L$ (see [15, p. 337]). □

But in some applications, it is not necessary to compute the complete SVD of $\boldsymbol{A}_{L,K}$. It can be sufficient to compute a partial SVD of $\boldsymbol{A}_{L,K}$. Further it is not necessary to calculate a best rank-S approximation $\boldsymbol{A}_{L,K}^{(S)}$ to high accuracy, since $\boldsymbol{A}_{L,K}$ may contain certain errors. Therefore it is desirable to develop less expensive methods for computing good approximations of $\boldsymbol{A}_{L,K}^{(S)}$.

Good low-rank approximations of $\boldsymbol{A}_{L,K}$ can be obtained from a *Lanczos bidiagonalization* (LBD) of $\boldsymbol{A}_{L,K}$ without computing any SVD. First we recall the LBD of a matrix $\boldsymbol{A}_{L,K} \in \mathbb{C}^{L\times K}$ with $L \geq K$. We assume that $\operatorname{rank} \boldsymbol{A}_{L,K} = K$ such that $\mathcal{N}(\boldsymbol{A}_{L,K}) = \{\boldsymbol{0}\}$. If the entries of $\boldsymbol{A}_{L,K}$ contain certain errors, then this assumption is quite natural. By [10, pp. 495 – 496] or [4, pp. 303 – 306], one can find a matrix $\boldsymbol{P}_{L,K} \in \mathbb{C}^{L\times K}$ with orthonormal columns and a unitary matrix $\boldsymbol{Q}_K \in \mathbb{C}^{K\times K}$ such that

$$\boldsymbol{A}_{L,K} = \boldsymbol{P}_{L,K}\, \boldsymbol{B}_K\, \boldsymbol{Q}_K^* \tag{2.3}$$

with a real, upper bidiagonal matrix

$$\boldsymbol{B}_K := \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \alpha_{K-1} & \beta_{K-1} \\ & & & & \alpha_K \end{pmatrix} \in \mathbb{R}^{K\times K}.$$

The columns $\boldsymbol{p}_j \in \mathbb{C}^L$ $(j = 1, \ldots, K)$ of $\boldsymbol{P}_{L,K}$ are called *left Lanczos vectors*. The columns $\boldsymbol{q}_j \in \mathbb{C}^K$ $(j = 1, \ldots, K)$ of $\boldsymbol{Q}_K$ are called *right Lanczos vectors*. From (2.3) it follows that

$$\boldsymbol{A}_{L,K}\, \boldsymbol{Q}_K = \boldsymbol{P}_{L,K}\, \boldsymbol{B}_K\,, \quad \boldsymbol{A}_{L,K}^*\, \boldsymbol{P}_{L,K} = \boldsymbol{Q}_K\, \boldsymbol{B}_K^{\mathrm{T}}\,. \tag{2.4}$$

Comparing the j-th columns of both sides of the equations (2.4), we obtain the Lanczos recursions

$$\alpha_1\, \boldsymbol{p}_1 = \boldsymbol{A}_{L,K}\, \boldsymbol{q}_1\,, \quad \alpha_{j+1}\, \boldsymbol{p}_{j+1} = \boldsymbol{A}_{L,K}\, \boldsymbol{q}_{j+1} - \beta_j\, \boldsymbol{p}_j\,, \tag{2.5}$$

$$\beta_j\, \boldsymbol{q}_{j+1} = \boldsymbol{A}_{L,K}^*\, \boldsymbol{p}_j - \alpha_j\, \boldsymbol{q}_j \quad (j = 1, \ldots, K-1)\,. \tag{2.6}$$

The orthonormality of the Lanczos vectors requires that $\alpha_1 = \|\boldsymbol{A}_{L,K}\, \boldsymbol{q}_1\|_2$, $\alpha_{j+1} = \|\boldsymbol{A}_{L,K}\, \boldsymbol{q}_{j+1} - \beta_j\, \boldsymbol{p}_j\|_2$ and $\beta_j = \|\boldsymbol{A}_{L,K}^*\, \boldsymbol{p}_j - \alpha_j\, \boldsymbol{q}_j\|_2$ $(j = 1, \ldots, K-1)$. If $\alpha_j > 0$ $(j = 1, \ldots, K)$ and $\beta_k > 0$ $(k = 0, \ldots, N-1)$, then the normed left/right Lanczos vectors are orthogonal by construction.

Now we describe the *partial Lanczos bidiagonalization* of $\boldsymbol{A}_{L,K}$ by S bidiagonalization steps, where $S \leq K$.

Algorithm 2.2 (Partial LBD with full reorthogonalization)

Input: S, L, $K \in \mathbb{N}$ with $L \geq K \geq S > 1$, $\boldsymbol{A}_{L,K} \in \mathbb{C}^{L \times K}$ with $\operatorname{rank} \boldsymbol{A}_{L,K} = K$, $\boldsymbol{q}_1 \in \mathbb{C}^K$ initial unit vector, S number of bidiagonalization steps.

1. Form $\boldsymbol{Q}_{K,1} := \boldsymbol{q}_1$ and compute $\boldsymbol{p}_1 := \boldsymbol{A}_{L,K}\, \boldsymbol{q}_1$.
2. Calculate $\alpha_1 := \|\boldsymbol{p}_1\|_2$ and $\boldsymbol{p}_1 := \boldsymbol{p}_1/\alpha_1$. Form $\boldsymbol{P}_{L,1} := \boldsymbol{p}_1$.
3. For $j = 1$ to $S - 1$ do

   - $\boldsymbol{q}_{j+1} := \boldsymbol{A}_{L,K}^*\, \boldsymbol{p}_j - \alpha_j\, \boldsymbol{q}_j$.
   - Reorthogonalization: $\boldsymbol{q}_{j+1} := \boldsymbol{q}_{j+1} - \boldsymbol{Q}_{K,j}\, \boldsymbol{Q}_{K,j}^*\, \boldsymbol{q}_{j+1}$.
   - $\beta_j := \|\boldsymbol{q}_{j+1}\|_2$, $\boldsymbol{q}_{j+1} := \boldsymbol{q}_{j+1}/\beta_j$, $\boldsymbol{Q}_{K,j+1} := (\boldsymbol{Q}_{K,j}\, ,\, \boldsymbol{q}_{j+1})$.
   - $\boldsymbol{p}_{j+1} := \boldsymbol{A}_{L,K}\, \boldsymbol{q}_{j+1} - \beta_j\, \boldsymbol{p}_j$.
   - Reorthogonalization: $\boldsymbol{p}_{j+1} := \boldsymbol{p}_{j+1} - \boldsymbol{P}_{L,j}\, \boldsymbol{P}_{L,j}^*\, \boldsymbol{p}_{j+1}$.
   - $\alpha_{j+1} := \|\boldsymbol{p}_{j+1}\|_2$, $\boldsymbol{p}_{j+1} := \boldsymbol{p}_{j+1}/\alpha_{j+1}$, $\boldsymbol{P}_{L,j+1} := (\boldsymbol{P}_{L,j}\, ,\, \boldsymbol{p}_{j+1})$.

4. Compute $\boldsymbol{r}_S := \boldsymbol{A}_{L,K}^*\, \boldsymbol{p}_S - \alpha_S\, \boldsymbol{q}_S$.

Output: $\boldsymbol{Q}_{K,S} = (\boldsymbol{q}_1\, ,\, \boldsymbol{q}_2\, ,\, \ldots\, ,\, \boldsymbol{q}_S) \in \mathbb{C}^{K \times S}$ and $\boldsymbol{P}_{L,S} = (\boldsymbol{p}_1\, ,\, \boldsymbol{p}_2\, ,\, \ldots\, ,\, \boldsymbol{p}_S) \in \mathbb{C}^{L \times S}$ matrices with orthonormal columns, $\boldsymbol{B}_S \in \mathbb{R}^{S \times S}$ upper bidiagonal matrix with diagonal entries $\alpha_j > 0$ $(j = 1, \ldots, S)$ and superdiagonal entries $\beta_j > 0$ $(j = 1, \ldots, S-1)$, $\boldsymbol{r}_S \in \mathbb{C}^N$ residual vector.

When the computations of Algorithm 2.2 are carried out in floating point arithmetic without the both reorthogonalization steps, the computed Lanczos vectors $\boldsymbol{q}_j$ and $\boldsymbol{p}_j$ might be far from orthogonality. Therefore one has to reorthogonalize these vectors. Several reorthogonalization strategies for the Lanczos vectors are discussed in the literature (see [20, 29, 31, 5]).
We are mainly interested in finding a low-rank approximation of $\boldsymbol{A}_{L,K}$. From the convergence theory of the LBD (see [4, pp. 309 – 310]), one knows that $\boldsymbol{P}_{L,S}$ and $\boldsymbol{Q}_{K,S}$ contains good approximations of the singular vectors related to the dominant singular values of $\boldsymbol{A}_{L,K}$. Thus it is quite natural (see [5]) to use the matrix $\boldsymbol{P}_{L,S}\, \boldsymbol{B}_S\, \boldsymbol{Q}_{K,S}^*$ as a low-rank approximation of $\boldsymbol{A}_{L,K}$. The *approximation error* of $\boldsymbol{P}_{L,S}\, \boldsymbol{B}_S\, \boldsymbol{Q}_{K,S}^*$ with respect to $\boldsymbol{A}_{L,K}$ is defined in the Frobenius norm by

$$\varepsilon_S := \|\boldsymbol{A}_{L,K} - \boldsymbol{P}_{L,S}\, \boldsymbol{B}_S\, \boldsymbol{Q}_{K,S}^*\|_F \quad (S = 1, \ldots, K)\,. \tag{2.7}$$

In the following lemma, we improve a result of [5] by the relation (2.8) and the computation of ε_1^2.

Lemma 2.3 *The approximation error* (2.7) *can be recursively determined by*

$$\varepsilon_{S+1}^2 = \varepsilon_S^2 - \alpha_{S+1}^2 - \beta_S^2 \quad (S = 1, \ldots, K-1)$$

with

$$\varepsilon_1^2 = \|\boldsymbol{A}_{L,K}\|_F^2 - \alpha_1^2\,, \quad \varepsilon_K = 0\,.$$

Further it holds for $S = 1, \ldots, K$

$$\|\boldsymbol{P}_{L,S}\,\boldsymbol{B}_S\,\boldsymbol{Q}_{K,S}^*\|_F^2 = \|\boldsymbol{B}_S\|_F^2 = \sum_{j=1}^{S-1}(\alpha_j^2 + \beta_j^2) + \alpha_S^2\,. \tag{2.8}$$

Since the proof follows similar lines as that in [5], we omit the proof.

Finally, we recollect the fast Hankel matrix-vector multiplication. As known, a cyclic convolution coincides with a circulant matrix-vector product (see [30, pp. 205 – 207]). Using the cyclic convolution property of the discrete Fourier transform (see [30, pp. 207 – 208]) and applying fast Fourier transform (FFT), one obtains immediately a fast evaluation of a circulant matrix-vector product. A similar technique can be applied to a fast computation for a product of a $L \times K$ Hankel matrix and a vector. We describe this method without any further assumption, i.e., we don't assume that the Hankel matrix is square (see [5]) or that there exists an FFT of length $L + K - 1$ (see [5, 19]).
Let an arbitrary Hankel matrix $\boldsymbol{H}_{L,K} := (h_{\ell+k})_{\ell,k=0}^{L-1,K-1} \in \mathbb{C}^{L\times K}$ be given. Then

$$\boldsymbol{T}_{L,K} := \boldsymbol{H}_{L,K}\,\boldsymbol{J}_K = \big(h_{K-1+\ell-k}\big)_{\ell,k=0}^{L-1,K-1} \in \mathbb{C}^{L\times K}$$

is a Toeplitz matrix. Here $\boldsymbol{J}_K := (\boldsymbol{e}_K, \boldsymbol{e}_{K-1}, \ldots, \boldsymbol{e}_1)$ denote the $K \times K$ counteridentity matrix, where $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_K\}$ is the standard basis of $\mathbb{C}^K$. Note that a fast computation for a product of a square Toeplitz matrix and a vector can be found in [30, pp. 208 – 209]. The same technique can be used for a rectangular Hankel matrix too. Thus we can rapidly compute a vector $\boldsymbol{y} = \boldsymbol{H}_{L,K}\,\boldsymbol{x} \in \mathbb{C}^L$ for an arbitrary vector $\boldsymbol{x} \in \mathbb{C}^K$ as follows:

Algorithm 2.4 (Fast Hankel matrix-vector multiplication)

Input: L, $K \in \mathbb{N}$, $\boldsymbol{x} = (x_k)_{k=0}^{K-1} \in \mathbb{C}^K$, $h_\ell \in \mathbb{C}$ $(\ell = 0, \ldots, L+K-2)$, $P \in \mathbb{N}$ is the smallest power of 2 with $P \geq L + K - 1$, i.e. $P := 2^{\lceil \log_2(L+K-1)\rceil}$.

1. Form the P-dimensional vectors

$$\begin{aligned}\tilde{\boldsymbol{h}} &:= (h_{K-1},\, h_K,\, \ldots,\, h_{L+K-2},\, \underbrace{0,\, \ldots,\, 0}_{P-L-K+1},\, h_0,\, h_1,\, \ldots,\, h_{K-2})^{\mathrm{T}}\,,\\ \tilde{\boldsymbol{x}} &:= (x_{K-1},\, x_{K-2}\, \ldots,\, x_0, \underbrace{0,\, \ldots,\, 0}_{P-K})^{\mathrm{T}}\,.\end{aligned}$$

2. For the $P \times P$ Fourier matrix $\boldsymbol{F}_P := \big(\exp(-2\pi \mathrm{i} jk/P)\big)_{j,k=0}^{P-1}$, compute the vectors $\boldsymbol{F}_P\,\tilde{\boldsymbol{h}}$ and $\boldsymbol{F}_P\,\tilde{\boldsymbol{x}}$ by FFT of length P.
3. Evaluate the componentwise vector product $\tilde{\boldsymbol{p}} := (\boldsymbol{F}_P\,\tilde{\boldsymbol{h}}) \circ (\boldsymbol{F}_P\,\tilde{\boldsymbol{x}})$.
4. Compute $\tilde{\boldsymbol{y}} := \frac{1}{P}\,\boldsymbol{J}'_P\,\boldsymbol{F}_P\,\tilde{\boldsymbol{p}}$ by FFT, where $\boldsymbol{J}'_P := (\boldsymbol{e}_1,\, \boldsymbol{e}_P,\, \ldots,\, \boldsymbol{e}_2)$ denotes the $P \times P$ flip matrix.
4. Form $\boldsymbol{y} := \tilde{\boldsymbol{y}}(0 : L-1)$.

Output: $\boldsymbol{y} := \boldsymbol{H}_{L,K}\,\boldsymbol{x} \in \mathbb{C}^L$ with $\boldsymbol{H}_{L,K} := (h_{\ell+k})_{\ell,k=0}^{L-1,K-1}$.

Remark 2.5 As known, an FFT of radix-2 length P requires $\frac{3\,P}{2}\,\log_2\,P$ operations. Thus in Algorithm 2.4, the three FFT operations need $\frac{9\,P}{2}\,\log_2\,P$ operations. Since the componentwise vector product involves only P multiplications, the Algorithm 2.4 requires about $\frac{9\,P}{2}\,\log_2\,P$ operations. By $\lceil \log_2(L+K-1) \rceil \le \log_2(L+K-1)+1$, we see that $P \le 2\,(L+K-1)$. Therefore the Algorithm 2.4 costs about $9\,(L+K-1)\,\log_2(L+K-1)$ operations. Since a classical matrix-vector multiplication involves $2\,LK$ operations, the Algorithm 2.4 is more efficient than a classical matrix-vector multiplication. Note that Algorithm 2.4 is valid for arbitrary $L,\, K \in \mathbb{N}$. □

Remark 2.6 Now we estimate the cost of the Algorithm 2.2 with S bidiagonalization steps, where $\boldsymbol{A}_{L,K}$ is a given Hankel matrix. The first step of Algorithm 2.2 requires the computation of $\boldsymbol{A}_{L,K}\,\boldsymbol{q}_1$. The other bidiagonalization steps need the calculations of $\boldsymbol{A}^*_{L,K}\,\boldsymbol{p}_j$ and $\boldsymbol{A}_{L,K}\,\boldsymbol{q}_{j+1}$ $(j = 1, \ldots, S-1)$. Applying Algorithm 2.4, we need about $18\,S(L+K-1)\,\log_2(L+K-1)$ operations for the fast computation of these $(2S-1)$ Hankel matrix-vector products. Since the Gram–Schmidt orthogonalization of S vectors of $\mathbb{C}^L$ costs $2\,S^2L$ operations (see [14, p. 369]), we can assume that the corresponding reorthogonalization steps in Algorithm 2.2 require $2\,S^2(L+K)$ operations. Since the other steps of Algorithm 2.2 have costs of linear order of $L+K$, the Algorithm 2.2 with S bidiagonalization steps needs about $18\,S\,(L+K-1)\,\log_2(L+K-1)$ operations. □

3. Exponential sums and trajectory matrices

In the following, we apply the described low-rank approximation to a Hankel matrix. We consider the following *parameter identification problem* of signal processing: Recover the positive integer M, the distinct numbers $f_j \in [-\alpha,\, 0] + \mathrm{i}\,[-\pi,\, \pi)$ with $\alpha > 0$, and the complex coefficients $c_j \neq 0$ $(j = 1, \ldots, M)$, in the nonincreasing *exponential sum of order* M

$$h(x) := \sum_{j=1}^{M} c_j\, \mathrm{e}^{f_j x} \quad (x \geq 0)\,, \tag{3.1}$$

if noisy sampled data $h_k = h(k) + e_k$ $(k = 0, \ldots, N-1)$ with sufficiently large integer N (with $2M \leq N$) are given, where e_k are small error terms. Often the sequence $\{h_0,\, h_1,\, \ldots,\, h_{N-1}\}$ of sampled data is called as a *time series of length* N. Then we form the *L-trajectory matrix* of this time series

$$\boldsymbol{H}_{L,N-L+1} := \big(h_{\ell+m}\big)_{\ell,m=0}^{L-1,N-L} \in \mathbb{C}^{L\times(N-L+1)} \tag{3.2}$$

with the *window length* $L \in \{1,\, \ldots,\, N\}$. Obviously, (3.2) is an $L\times(N-L+1)$ Hankel matrix. Further we introduce the exact L-trajectory matrix

$$\boldsymbol{H}^{(0)}_{L,N-L+1} := \big(h(\ell+m)\big)_{\ell,m=0}^{L-1,N-L} \in \mathbb{C}^{L\times(N-L+1)} \tag{3.3}$$

and the corresponding error matrix

$$\boldsymbol{E}_{L,N-L+1} := \big(e_{\ell+m}\big)_{\ell,m=0}^{L-1,N-L} \in \mathbb{C}^{L\times(N-L+1)}$$

such that

$$\boldsymbol{H}_{L,N-L+1} = \boldsymbol{H}^{(0)}_{L,N-L+1} + \boldsymbol{E}_{L,N-L+1}\,.$$

Note that the negative real part of f_j is the damping factor and the imaginary part of f_j is the angular frequency of the exponential $\mathrm{e}^{f_j x}$. The nodes $z_j := \mathrm{e}^{f_j}$ $(j = 1, \ldots, M)$ are distinct values in the annulus $\mathbb{D} := \{z \in \mathbb{C} : \mathrm{e}^{-\alpha} \leq |z| \leq 1\}$.

The main step in the solution of the parameter identification problem is the determination of the order M and the computation of the "frequencies" f_j or alternatively of the nodes z_j $(j = 1, \ldots, M)$. Afterward one can calculate the coefficient vector $\boldsymbol{c} := (c_k)_{k=1}^{M}$ as least squares solution of overdetermined linear system

$$\boldsymbol{V}_{N,M}(\boldsymbol{z})\, \boldsymbol{c} = (h_j)_{j=0}^{N-1} \tag{3.4}$$

with the *rectangular Vandermonde matrix* $\boldsymbol{V}_{N,M}(\boldsymbol{z}) := (z_k^{j-1})_{j,k=1}^{N,M}$ and the vector $\boldsymbol{z} := (z_k)_{k=1}^{M}$. As known, the square Vandermonde matrix $\boldsymbol{V}_M(\boldsymbol{z})$ is invertible. Note that

$$\operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = \min\{L,\, M\} \quad (L = 1,\, \ldots,\, N)\,, \tag{3.5}$$

since $\operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) \le \min\{L,\, M\}$ and since the submatrix $(z_k^{j-1})_{j,k=1}^{\min\{L,M\}}$ is invertible.

By (3.1), the exact L-trajectory matrix (3.3) can be factorized in the following form

$$\boldsymbol{H}_{L,N-L+1}^{(0)} = \boldsymbol{V}_{L,M}(\boldsymbol{z})\, (\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z})^{\mathrm{T}}\,. \tag{3.6}$$

Lemma 3.1 *For each $L = 1,\, \ldots,\, N$ with $N \ge 2\,M$, the rank of the exact L-trajectory matrix* (3.3) *is given by*

$$\operatorname{rank} \boldsymbol{H}_{L,N-L+1}^{(0)} = \min\{L,\, N-L+1,\, M\}\,.$$

Proof. 1. Assume that $M \ge 2$. For $L \in \{1,\, \ldots,\, M-1\}$, we know by (3.5) that $\operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = L$ and $\operatorname{rank} \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z}) = M$. Thus the rank of the $M \times (N-L+1)$ matrix $(\operatorname{diag} \boldsymbol{z})\, \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z})^{\mathrm{T}}$ is equal to M. Hence we obtain that

$$\begin{aligned} \operatorname{rank} \boldsymbol{H}_{L,N-L+1}^{(0)} &= \operatorname{rank} \Big(\boldsymbol{V}_{L,M}(\boldsymbol{z}) \big((\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z})^{\mathrm{T}} \big) \Big) \\ &= \operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = L\,. \end{aligned}$$

Note that $\min\{L,\, N-L+1,\, M\} = L$ for $L \in \{1,\, \ldots,\, M-1\}$.
2. For $L \in \{M,\, \ldots,\, N-M\}$, we see by (3.5) that

$$\operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = \operatorname{rank} \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z}) = M\,.$$

Thus the rank of the $M \times (N-L+1)$ matrix $(\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z})^{\mathrm{T}}$ is equal to M. Hence we conclude that

$$\begin{aligned} \operatorname{rank} \boldsymbol{H}_{L,N-L+1}^{(0)} &= \operatorname{rank} \Big(\boldsymbol{V}_{L,M}(\boldsymbol{z}) \big((\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z})^{\mathrm{T}} \big) \Big) \\ &= \operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = M\,. \end{aligned}$$

Note that $\min\{L,\, N-L+1,\, M\} = M$ for $L \in \{M,\, \ldots,\, N-M\}$.
3. In the case $L \in \{N-M+1,\, \ldots,\, N\}$, we use the property

$$\operatorname{rank} \boldsymbol{H}_{L,N-L+1}^{(0)} = \operatorname{rank} \big(\boldsymbol{H}_{L,N-L+1}^{(0)} \big)^{\mathrm{T}}$$

and the transposed factorization (3.6). By (3.5), we see that $\operatorname{rank} \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z}) = N - L + 1$ and $\operatorname{rank} \boldsymbol{V}_{L,M}(\boldsymbol{z}) = M$. Thus the rank of the $M \times L$ matrix $(\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{L,M}(\boldsymbol{z})^{\mathrm{T}}$ is equal to M. Hence we obtain that

$$\begin{aligned} \operatorname{rank} \left(\boldsymbol{H}^{(0)}_{L,N-L+1}\right)^{\mathrm{T}} &= \operatorname{rank} \Big(\boldsymbol{V}_{N-L+1,M}(\boldsymbol{z}) \left((\operatorname{diag} \boldsymbol{c})\, \boldsymbol{V}_{L,M}(\boldsymbol{z})^{\mathrm{T}}\right)\Big) \\ &= \operatorname{rank} \boldsymbol{V}_{N-L+1,M}(\boldsymbol{z}) = N - L + 1\,. \end{aligned}$$

Note that $\min\{L,\, N-L+1,\, M\} = N-L+1$ for $L \in \{N-M+1,\, \ldots,\, N\}$. This completes the proof. □

Thus we have shown that for convenient window length L with $M \le L \le N - M + 1$, the rank of the exact L-trajectory matrix (3.3) coincides with the order of the exponential sum (3.1) (see Figure 3.1).

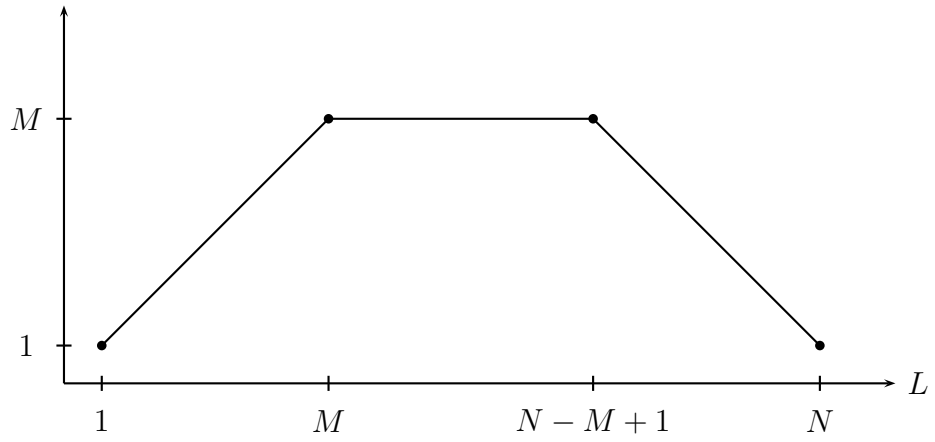

Figure 3.1: The rank of the L-trajectory matrix (3.3) for several window lengths $L = 1, \ldots, N$.

Remark 3.2 The rank of an arbitrary Hankel matrix has a similar behavior, see [13, pp. 80 – 81]. For an L-trajectory matrix (3.3) of an exponential sum (3.1), we can explicitly determine the so-called characteristic degrees M and $N - M + 1$ (see [13, p. 81]) as well as the maximum rank M of the Hankel matrix (3.3). Further the proof of Lemma 3.1 is different from the corresponding proof in [13, pp. 80 – 81]. □

The *nullity* of the exact L-trajectory matrix (3.3) is defined by

$$\operatorname{null} \boldsymbol{H}^{(0)}_{L,N-L+1} := \dim \mathcal{N}(\boldsymbol{H}^{(0)}_{L,N-L+1})\,.$$

Corollary 3.3 *For each $L = 1, \ldots, N$ with $N \geq 2M$, the nullity of the exact L-trajectory matrix* (3.3) *is given by*

$$\operatorname{null} \boldsymbol{H}^{(0)}_{L,N-L+1} = \max\{N - 2\,L + 1,\, N - L - M + 1,\, 0\}\,.$$

Proof. Splitting the column space $\mathbb{C}^{N-L+1}$ of (3.3) into the orthogonal sum of null space $\mathcal{N}(\boldsymbol{H}^{(0)}_{L,N-L+1})$ and the range $\mathcal{R}\big((\boldsymbol{H}^{(0)}_{L,N-L+1})^*\big)$, we obtain for the corresponding dimensions

$$\begin{aligned} N - L + 1 &= \dim \mathcal{N}(\boldsymbol{H}^{(0)}_{L,N-L+1}) + \dim \mathcal{R}\big((\boldsymbol{H}^{(0)}_{L,N-L+1})^*\big) \\ &= \operatorname{null} \boldsymbol{H}^{(0)}_{L,N-L+1} + \operatorname{rank}\big((\boldsymbol{H}^{(0)}_{L,N-L+1})^*\big) \\ &= \operatorname{null} \boldsymbol{H}^{(0)}_{L,N-L+1} + \operatorname{rank} \boldsymbol{H}^{(0)}_{L,N-L+1} \end{aligned}$$

and hence

$$\operatorname{null} \boldsymbol{H}^{(0)}_{L,N-L+1} = N - L + 1 - \operatorname{rank} \boldsymbol{H}^{(0)}_{L,N-L+1}\,.$$

Using Lemma 3.1, it follows the result. □

For a Hankel matrix (3.2) with noisy entries, we consider the numerical rank. Let σ_j ($j = 1, \ldots, \min\{L, N - L + 1\}$) be the singular values of (3.2) arranged in nonincreasing order

$$\sigma_1 = \|\boldsymbol{H}_{L,N-L+1}\|_2 \geq \sigma_2 \geq \ldots \geq \sigma_{\min\{L,N-L+1\}}\,.$$

Then the *numerical rank* of (3.2) is defined as the largest integer R such that $\sigma_R \geq \varepsilon\,\sigma_1$ for sufficiently small tolerance $\varepsilon > 0$. Using IEEE double precision arithmetic, one can choose $\varepsilon = 10^{-10}$ for given exact data. For noisy data, one has to choose a larger tolerance ε. Depending on the noise level, there is usually an obvious gap in the singular value distribution such that ε can be suitably chosen. Now we show that the numerical rank of the noisy L-trajectory matrix (3.2) has a similar behavior for a window length $L \in \{M, \ldots, N - M + 1\}$.

Lemma 3.4 *Assume that for a window length L with $M \leq L \leq N - M + 1$, the estimates $\|\boldsymbol{E}_{L,N-L+1}\|_2 < \varepsilon\,\|\boldsymbol{H}_{L,N-L+1}\|_2$ and $\sigma_M \geq \varepsilon\,\|\boldsymbol{H}_{L,N-L+1}\|_2$ are fulfilled.*
Then the numerical rank of (3.2) *is equal to M and coincides with the rank of* (3.3).

Proof. By assumption, we have

$$\sigma_M \geq \varepsilon \, \|\boldsymbol{H}_{L,N-L+1}\|_2 = \varepsilon \, \sigma_1 \, .$$

Now we show that $\sigma_{M+1} < \varepsilon \, \sigma_1$. Let $\sigma_j^{(0)}$ $(j = 1, \ldots, \min\{L, N-L+1\})$ be the singular values of the exact L-trajectory matrix (3.3) arranged in the form

$$\sigma_1^{(0)} \geq \ldots \geq \sigma_M^{(0)} > \sigma_{M+1}^{(0)} = \ldots = \sigma_{\min\{L,N-L+1\}}^{(0)} = 0 \, .$$

By the matrix perturbation theory (see [16], p. 451) we know that

$$|\sigma_{M+1} - \sigma_{M+1}^{(0)}| = \sigma_{M+1} \leq \|\boldsymbol{E}_{L,N-L+1}\|_2$$

and hence by the assumption

$$\sigma_{M+1} < \varepsilon \, \|\boldsymbol{H}_{L,N-L+1}\|_2 = \varepsilon \, \sigma_1 \, .$$

This completes the proof. □

4. ESPRIT via complete/partial SVD

Assume that $M \leq L \leq N - M$ and $N \geq 2\,M \geq 2$. With the N sampled data $h_k \in \mathbb{C}$ $(k = 0, \ldots, N-1)$, additionally we form the *rectangular Hankel matrices*

$$\boldsymbol{H}_{L,N-L}(s) := \boldsymbol{H}_{L,N-L+1}(1 : L,\, 1+s : N-L+s) \quad (s = 0,\, 1) \, . \tag{4.1}$$

In the case of exactly sampled data, the Hankel matrices (4.1) are denoted by $\boldsymbol{H}_{L,N-L}^{(0)}(s)$ $(s = 0,\, 1)$.

Remark 4.1 The Hankel matrices $\boldsymbol{H}_{L,N-L}^{(0)}(s)$ $(s = 0,\, 1)$ have the same rank M for each $L \in \{M, \ldots, N-M\}$. By Lemma 3.1 it follows immediately that for $L = 1, \ldots, N-1$

$$\operatorname{rank} \boldsymbol{H}_{L,N-L}^{(0)}(0) = \operatorname{rank} \boldsymbol{H}_{L,N-L}^{(0)} = \min\{L,\, N-L,\, M\} \, .$$

Hence we obtain $\operatorname{rank} \boldsymbol{H}_{L,N-L}^{(0)}(0) = M$ for each $L \in \{M, \ldots, N-M\}$.
Note that the proof of Lemma 3.1 is mainly based on the factorization (3.6). The Hankel matrix $\boldsymbol{H}_{L,N-L}^{(0)}(1)$ has also the rank M for each $L \in \{M, \ldots, N-M\}$. This follows from the fact that $\boldsymbol{H}_{L,N-L}^{(0)}(1)$ can be factorized in a similar form as (3.6), namely

$$\boldsymbol{H}_{L,N-L}^{(0)}(1) = \boldsymbol{V}_{L,M}(\boldsymbol{z}) \, (\operatorname{diag} \boldsymbol{c}) \, (\operatorname{diag} \boldsymbol{z}) \, \boldsymbol{V}_{N-L,M}(\boldsymbol{z})^{\mathrm{T}} \, ,$$

and from the proof of Lemma 3.1. □

The rectangular matrix pencil

$$z\,\boldsymbol{H}^{(0)}_{L,N-L}(0) - \boldsymbol{H}^{(0)}_{L,N-L}(1) \quad (z \in \mathbb{C}) \tag{4.2}$$

has the nodes $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ as eigenvalues (see [25]), where $\mathbb{D}$ is the annulus introduced in Section 3. Now we explain some algorithms of ESPRIT (Estimation of Signal Parameters via Rotational Invariance Technique). For details see [27, 28, 25]. First we assume that exact sampled data $h_k = h(k)$ $(k = 0, \ldots, N-1)$ are given. Then we start the ESPRIT method by the SVD of (3.2), i.e.

$$\boldsymbol{H}^{(0)}_{L,N-L+1} = \boldsymbol{U}_L\,\boldsymbol{D}_{L,N-L+1}\,\boldsymbol{W}^*_{N-L+1}\,,$$

where $\boldsymbol{U}_L \in \mathbb{C}^{L\times L}$ and $\boldsymbol{W}_{N-L+1} \in \mathbb{C}^{(N-L+1)\times(N-L+1)}$ are unitary matrices and where $\boldsymbol{D}_{L,N-L+1} \in \mathbb{R}^{L\times(N-L+1)}$ is a rectangular diagonal matrix. The diagonal entries of $\boldsymbol{D}_{L,N-L+1}$ are the singular values $\sigma^{(0)}_j$ of (3.2) arranged in nonincreasing order $\sigma^{(0)}_1 \geq \ldots \geq \sigma^{(0)}_M > \sigma^{(0)}_{M+1} = \ldots = \sigma^{(0)}_{\min\{L,N-L+1\}} = 0$. Thus we can determine the order M of the exponential sum (3.1) by the number of positive singular values $\sigma^{(0)}_j$ of (3.2). Introducing the matrices $\boldsymbol{U}_{L,M} := \boldsymbol{U}_L(1:L, 1:M)$ and $\boldsymbol{W}_{L+1,M} := \boldsymbol{W}_{N-L+1}(1:N-L+1, 1:M)$ with orthonormal columns as well as the diagonal matrix $\boldsymbol{D}_M := \operatorname{diag}(\sigma^{(0)}_j)^M_{j=1}$, we obtain the partial SVD for the matrix (3.2) with exact entries, i.e.

$$\boldsymbol{H}^{(0)}_{L,N-L+1} = \boldsymbol{U}_{L,M}\,\boldsymbol{D}_M\,\boldsymbol{W}^*_{N-L+1,M}\,. \tag{4.3}$$

Setting

$$\boldsymbol{W}_{N-L,M}(s) := \boldsymbol{W}_{N-L+1,M}(1+s:N-L+s, 1:M) \quad (s = 0,\, 1)\,, \tag{4.4}$$

it follows by (4.3) and (4.1) that both Hankel matrices (4.1) can be simultaneously factorized in the form

$$\boldsymbol{H}^{(0)}_{L,N-L}(s) = \boldsymbol{U}_{L,M}\,\boldsymbol{D}_M\,\boldsymbol{W}_{N-L,M}(s)^* \quad (s = 0,\, 1)\,.$$

Since $\boldsymbol{U}_{L,M}$ has orthonormal columns and since $\boldsymbol{D}_M$ is invertible, the generalized eigenvalue problem of the matrix pencil

$$z\,\boldsymbol{W}_{N-L,M}(0)^* - \boldsymbol{W}_{N-L,M}(1)^* \quad (z \in \mathbb{C}) \tag{4.5}$$

has the same non-zero eigenvalues $z_j \in \mathbb{D}$ as the matrix pencil (4.5) except for additional zero eigenvalues. Finally we determine the nodes $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ as eigenvalues of the $M \times M$ matrix

$$\boldsymbol{F}_M^{\mathrm{SVD}} := \left(\boldsymbol{W}_{N-L,M}(0)^*\right)^\dagger \boldsymbol{W}_{N-L,M}(1)^* \,. \tag{4.6}$$

Note that one can pass on the computation of the Moore–Penrose pseudoinverse in (4.6). For this, we multiply the matrix pencil (4.5) with $\boldsymbol{W}_{N-L,M}(0)$ from the right. Then we obtain the positive definite matrix

$$\boldsymbol{A}_M := \boldsymbol{W}_{N-L,M}(0)^* \, \boldsymbol{W}_{N-L,M}(0)$$

and the matrix $\boldsymbol{B}_M := \boldsymbol{W}_{N-L,M}(1)^* \, \boldsymbol{W}_{N-L,M}(0)$. Now we solve the generalized eigenvalue problem of the square matrix pencil

$$z \, \boldsymbol{A}_M - \boldsymbol{B}_M \quad (z \in \mathbb{C}) \tag{4.7}$$

with the QZ–Algorithm (see [10, pp. 384 – 385]). Then we maintain the factorizations

$$\boldsymbol{A}_M = \boldsymbol{Q}_M \, \boldsymbol{S}_M \, \boldsymbol{Z}_M^* \,, \quad \boldsymbol{B}_M = \boldsymbol{Q}_M \, \boldsymbol{T}_M \, \boldsymbol{Z}_M^* \,,$$

with unitary matrices $\boldsymbol{Q}_M$ and $\boldsymbol{Z}_M$ and with upper triangular matrices $\boldsymbol{S}_M$ and $\boldsymbol{T}_M$. Then the eigenvalues z_j of the matrix pencil read as follows $z_j = T_{j,j}/S_{j,j}$ $(j = 1, \ldots, M)$, where $S_{j,j}$ and $T_{j,j}$ are the diagonal entries of $\boldsymbol{S}_M$ and $\boldsymbol{T}_M$, respectively. The computation of the matrices $\boldsymbol{A}_M$ and $\boldsymbol{B}_M$ costs about $4\,(N-L)M^2$ operations. The QZ–Algorithm requires only $30\,M^3$ operations for the computation of the eigenvalues z_j (see [10, p. 385]).

Analogously we can handle the general case of noisy data $h_k = h(k) + e_k$ $(k = 0, \ldots, N-1)$ with small error terms e_k. For the Hankel matrix (3.2) with the singular values $\sigma_1 \geq \ldots \geq \sigma_{\min\{L,N-L+1\}} \geq 0$, we can calculate the numerical rank M of (3.2) by the property $\sigma_M \geq \varepsilon \, \sigma_1$ and $\sigma_{M+1} < \varepsilon \, \sigma_1$ with convenient chosen tolerance ε. Using the IEEE double precision arithmetic, one can choose $\varepsilon = 10^{-10}$ for given exact data. In the case of noisy data, one has to choose a larger tolerance ε.
For the Hankel matrix (3.2) with noisy entries, we use the partial SVD

$$\boldsymbol{U}_{L,M} \, \boldsymbol{D}_M \, \boldsymbol{W}_{N-L+1,M}^*$$

as low-rank approximation, where the spectral norm of the error reads as follows

$$\|\boldsymbol{H}_{L,N-L+1} - \boldsymbol{U}_{L,M} \, \boldsymbol{D}_M \, \boldsymbol{W}_{N-L+1,M}^*\|_2 = \sigma_{M+1} \,.$$

As above, we define the matrices (4.4) and (4.6). Note that

$$\boldsymbol{U}_{L,M}\,\boldsymbol{D}_M\,\boldsymbol{W}_{N-L,M}(s)^* \quad (s=0,\,1)$$

is a low-rank approximation of $\boldsymbol{H}_{L,N-L}(s)$ with the property

$$\|\boldsymbol{H}_{L,N-L}(s) - \boldsymbol{U}_{L,M}\,\boldsymbol{D}_M\,\boldsymbol{W}_{N-L,M}(s)^*\|_2 \le \sigma_{M+1}\,.$$

Thus the SVD-based ESPRIT algorithm reads as follows:

Algorithm 4.2 (ESPRIT via complete/partial SVD)
Input: $L,\ M,\ N \in \mathbb{N}$ ($N \gg 1$, $\max\{3,\,M\} \le L \le N-M$, M is the order of (3.1)), $h_k = h(k) + e_k \in \mathbb{C}$ ($k = 0,\ldots,N-1$) noisy sampled values of (3.1), $0 < \varepsilon \ll 1$ tolerance.

1. Compute the complete/partial SVD of the rectangular Hankel matrix (3.2). Determine the numerical rank M of (3.2) such that $\sigma_M \ge \varepsilon\,\sigma_1$ and $\sigma_{M+1} < \varepsilon\sigma_1$ and form the matrices (4.4).
2. Calculate the matrix products

$$\boldsymbol{W}_{N-L,M}(0)^*\,\boldsymbol{W}_{N-L,M}(0)\,, \quad \boldsymbol{W}_{N-L,M}(1)^*\,\boldsymbol{W}_{N-L,M}(0)$$

and compute all eigenvalues $z_j \in \mathbb{D}$ ($j = 1,\ldots,M$) of the square matrix pencil (4.7) by the QZ–Algorithm. Set $f_j := \log z_j$ ($j = 1,\ldots,M$), where log denotes the principal value of the complex logarithm.
3. Compute the coefficients $c_j \in \mathbb{C}$ ($j = 1,\ldots,M$) as least squares solution of the overdetermined linear Vandermonde–type system (3.4).

Output: $M \in \mathbb{N}$, $f_j \in (-\infty,\,0] + \mathrm{i}\,[-\pi,\pi)$, $c_j \in \mathbb{C}$ ($j = 1,\ldots,M$).

Remark 4.3 The high computational cost for the SVD of (3.2) is an essential drawback of Algorithm 4.2 based on complete SVD. In the case $L \approx \frac{N}{2}$, the Algorithm 4.2 with complete SVD requires $\frac{21}{8}\,N^3 + M^2(21\,N + \frac{91}{3}\,M)$ operations. The computational cost of Algorithm 4.2 is mainly determined by the cost of the SVD in step 1 (cf. Remark 2.1), since step 2 requires $20\,N\,M^2 + 30\,M^3$ operations and since the least squares problem in step 3 can be solved by $M^2(N + \frac{M}{3})$ operations (see [14, p. 386]). □

Remark 4.4 Let $K := N - L + 1 \in \{M, \ldots, N - M + 1\}$ be a convenient upper bound of the unknown order M of the exponential sum (3.1). In step 1 of Algorithm 4.2, a reduced SVD of the L-trajectory matrix (3.2) can be efficiently realized by a spectral decomposition of the Hermitian matrix $\boldsymbol{H}_{L,K}^* \boldsymbol{H}_{L,K}$, which can be computed by the fast Algorithm 2.4 with about $9\,KN\,\log_2 N$ operations. Compared with (4.3), we obtain the spectral decomposition

$$\boldsymbol{H}_{L,K}^* \boldsymbol{H}_{L,K} = \boldsymbol{W}_K \boldsymbol{D}_K^2 \boldsymbol{W}_K^* . \tag{4.8}$$

By Remark 2.1, the unitary matrix $\boldsymbol{W}_K$ and the diagonal matrix $\boldsymbol{W}_K$ can be computed by $9\,K^3$ operations. Consequently, step 1 of Algorithm 4.2 based on reduced SVD requires about $9\,KN\,\log_2 N + 9\,K^3$ operations. Then step 2 costs about $40\,KM^2 + 30\,M^3$ operations and step 3 requires $M^2\,(N + \frac{1}{3}\,M)$ operations. Thus the computational cost of this ESPRIT Algorithm 4.2 amounts $9\,KN\,\log_2 N + 9\,K^3 + (40\,K + N)\,M^2 + \frac{91}{3}\,M^3$ operations. □

5. ESPRIT via partial LBD

If noisy sampled data $h_k = h(k) + e_k \in \mathbb{C}$ $(k = 0, \ldots, N-1)$ are given, then the rectangular Hankel matrix $\boldsymbol{H}_{L,N-L+1}$ has full column rank in general, if $L \geq (N+1)/2$. Instead of SVD we apply partial LBD of (3.2) with S bidiagonalization steps, where $M \leq S \leq N - L + 1$. Then we obtain by Algorithm 2.2 that

$$\begin{aligned} \boldsymbol{H}_{L,N-L+1}\,\boldsymbol{Q}_{N-L+1,S} &= \boldsymbol{P}_{L,S}\,\boldsymbol{B}_S\,, \\ \boldsymbol{H}_{L,N-L+1}^*\,\boldsymbol{P}_{L,S} &= \boldsymbol{Q}_{N-L+1,S}\,\boldsymbol{B}_S^{\mathrm{T}} + \boldsymbol{r}_S\,\boldsymbol{e}_S^{\mathrm{T}}\,, \end{aligned} \tag{5.1}$$

where $\boldsymbol{B}_S$ is an upper bidiagonal $S \times S$ matrix, $\boldsymbol{e}_S := (0, \ldots, 0, 1)^{\mathrm{T}} \in \mathbb{C}^S$ and $\boldsymbol{r}_S \in \mathbb{C}^{N-L+1}$ is the Sth residual vector. The matrices $\boldsymbol{P}_{L,S}$ and $\boldsymbol{Q}_{N-L+1,S}$ have orthonormal columns. Typically, the number S of bidiagonalization steps is much smaller than N. Then we use the matrix $\boldsymbol{P}_{L,S}\,\boldsymbol{B}_S\,\boldsymbol{Q}_{N-L+1,S}^*$ as low-rank approximation of (3.2). Introducing the matrices

$$\boldsymbol{Q}_{N-L,S}(s) := \boldsymbol{Q}_{N-L+1,S}(1 + s : N - L + s, 1 : M) \quad (s = 0,\, 1)\,,$$

it follows that both Hankel matrices (4.1) have low-rank approximations of following form

$$\boldsymbol{P}_{L,S}\,\boldsymbol{B}_S\,\boldsymbol{Q}_{N-L,S}(s)^* \quad (s = 0,\, 1)\,.$$

Since $\boldsymbol{P}_{L,S}$ has orthonormal columns and since $\boldsymbol{B}_S$ is invertible, the generalized eigenvalue problem of the matrix pencil

$$z\,\boldsymbol{Q}_{N-L,S}(0)^* - \boldsymbol{Q}_{N-L,S}(1)^* \tag{5.2}$$

has approximately the non-zero eigenvalues $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ as the matrix pencil (4.2) except for additional almost zero eigenvalues. Finally we determine the nodes $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ as non-zero eigenvalues of the $S \times S$ matrix

$$\boldsymbol{F}_S^{\mathrm{LBD}} := \left(\boldsymbol{Q}_{N-L,S}(0)^*\right)^\dagger \boldsymbol{Q}_{N-L,S}(1)^* \,. \tag{5.3}$$

Analogously to Section 4, one can pass on the computation of the Moore–Penrose pseudoinverse in (5.3). For this, we multiply the matrix pencil (5.2) with $\boldsymbol{Q}_{N-L,S}(0)$ from right. Then we obtain the positive definite matrix $\boldsymbol{A}_S := \boldsymbol{Q}_{N-L,S}(0)^* \boldsymbol{Q}_{N-L,S}(0)$ and the matrix $\boldsymbol{B}_S := \boldsymbol{Q}_{N-L,S}(1)^* \boldsymbol{Q}_{N-L,S}(0)$. Now we solve the generalized eigenvalue problem of the squared matrix pencil

$$z\,\boldsymbol{A}_S - \boldsymbol{B}_S \quad (z \in \mathbb{C})$$

with the QZ–Algorithm (see [10, pp. 384 – 385]). Then we maintain the factorizations

$$\boldsymbol{A}_S = \boldsymbol{Q}_S\,\boldsymbol{S}_S\,\boldsymbol{Z}_S^*\,, \quad \boldsymbol{B}_S = \boldsymbol{Q}_S\,\boldsymbol{T}_S\,\boldsymbol{Z}_S^*\,,$$

with unitary matrices $\boldsymbol{Q}_S$ and $\boldsymbol{Z}_S$ and with upper triangular matrices $\boldsymbol{S}_S$ and $\boldsymbol{T}_S$. Then the eigenvalues $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ of the matrix pencil read as follows $z_j = T_{k,k}/S_{k,k}$ for convenient $k \in \{1, \ldots, S\}$, where $S_{k,k}$ and $T_{k,k}$ denote the diagonal entries of $\boldsymbol{S}_S$ and $\boldsymbol{T}_S$, respectively. The computation of the matrices $\boldsymbol{A}_S$ and $\boldsymbol{B}_S$ costs about $4\,(N-L)S^2$ operations. The QZ–Algorithm requires only $30\,S^3$ operations for the computation of the eigenvalues z_j [10, pp. 385].

Algorithm 5.1 (ESPRIT via partial LBD)
Input: L, M, $N \in \mathbb{N}$ $(N \gg 1$, $\max\{3,\, M\} \leq L \leq N - M)$, M is the order of (3.1)), $h_k = h(k) + e_k \in \mathbb{C}$ $(k = 0, \ldots, N-1)$ noisy sampled values of (3.1), $S \in \{M, \ldots, N - L + 1\}$ number of bidiagonalization steps.

1. Compute S steps of the partial LBD of the Hankel matrix (3.2), where all Hankel matrix-vector products are calculated by Algorithm 2.4.
2. Calculate $\boldsymbol{Q}_{N-L,S}(0)^*\,\boldsymbol{Q}_{N-L,S}(0)$ and $\boldsymbol{Q}_{N-L,S}(1)^*\,\boldsymbol{Q}_{N-L,S}(0)$ and compute all eigenvalues $z_j \in \mathbb{D}$ $(j = 1, \ldots, M)$ of the square matrix pencil (4.7) by the QZ–Algorithm. Set $f_j := \log z_j$ $(j = 1, \ldots, M)$.
3. Compute the coefficients $c_j \in \mathbb{C}$ $(j = 1, \ldots, M)$ as least squares solution of the overdetermined linear Vandermonde–type system (3.4).

Output: $M \in \mathbb{N}$, $f_j \in (-\infty,\, 0] + \mathrm{i}\,[-\pi, \pi)$, $c_j \in \mathbb{C}$ $(j = 1, \ldots, M)$.

Remark 5.2 Let $S \in \{M, \ldots, N-L+1\}$ be a convenient upper bound for M. Then we apply Algorithm 5.1 with S bidiagonalization steps. The advantage of Algorithm 5.1 over Algorithm 4.2 is the lower cost with about $18\,NS\,\log_2 N + S^2(20\,N + 30\,S) + M^2(N + \frac{1}{3}\,M)$ operations, if $L \approx \frac{N}{2}$. The computational cost of Algorithm 5.1 is mainly determined by the cost of step 1, since step 2 requires $20\,N\,S^2 + 30\,S^3$ operations and since the least squares problem in step 3 can be solved by $M^2(N + \frac{1}{3}\,M)$ operations (see [14, p. 386]). Thus Algorithm 5.1 is very convenient for the analysis of times series $\{h_0, h_1, \ldots, h_{N-1}\}$ with large length N. Numerical examples in Section 6 demonstrate the performance of Algorithm 5.1. □

6. Numerical examples

Now we illustrate the behavior of the suggested algorithms. Using IEEE standard floating point arithmetic with double precision, we have implemented our algorithms in Matlab. We compare the Algorithm 4.2 and Algorithm 5.1 within different implementations. We use in Algorithm 4.2 the Matlab command "svd" in order to compute the SVD and the command "svds" in order to compute a partial SVD. Furthermore we use the partial LBD suggested in [1] and apply the corresponding Matlab program IRLBA (see http://www.netlib.org/numeralgo/na26.tgz) in combination with fast Hankel matrix-vector multiplications of Algorithm 2.4. Further we remark that we also used the Matlab program HANKELSVD written by K. Browne, see [5]. This algorithm uses the Lanczos bidiagonalization method with reorthogonalization and applies fast Hankel matrix-vector multiplications in each iteration step in order to commute the splitting (5.1). The algorithm performs very well, but is slightly more sensitive with respect to noise. Therefore we will not report numerical results of the method [5].

Example 6.1 We consider the exponential sum of order $M = 5$

$$\begin{aligned} h(x) &:= 34 + 300\,\mathrm{e}^{x\pi\mathrm{i}/4} + 300\,\mathrm{e}^{-x\pi\mathrm{i}/4} + \mathrm{e}^{x\pi\mathrm{i}/2} + \mathrm{e}^{-x\pi\mathrm{i}/2} \\ &= 34 + 600\,\cos\frac{x\pi}{4} + 2\,\cos\frac{x\pi}{2} \quad (x \geq 0) \end{aligned}$$

and form the corresponding time series

$$h_k := h(k) + e_k = 34 + 600\,\cos\frac{k\pi}{4} + 2\,\cos\frac{k\pi}{2} + e_k \quad (k = 0, \ldots, N-1)\,, \tag{6.1}$$

where e_k are random variables uniformly distributed in $[-3, 3]$. A similar time series was suggested in [6]. Note that in addition to the large differences

in the magnitudes at distinct frequencies, also the noise is higher than the amplitude related to the frequency $\pi/2$. We choose $N = 1024$ and form the L-trajectory matrix (3.2) for different window lengths $L = 1004\,(-1)\,524$. Then the number of columns of (3.2) runs from 21 to 501. We investigate the behavior of the Algorithms 4.2 and 5.1 for different window lengths. For each $L = 1004\,(-1)\,524$, we analyze 10 times series (6.1) of length 1024 with different error terms. Using the L-trajectory matrix (3.2), we determine the exponents $f_k(L)$ and the coefficients $c_k(L)$ $(k = 1, \ldots, 5)$ by the Algorithms 4.2 and 5.1, respectively. Then we measure the maximum frequency errors $\max |f_1(L)|$, $\max |f_2(L) - \frac{\mathrm{i}\pi}{4}|$, $\max |f_3 + \frac{\mathrm{i}\pi}{4}|$, $\max |f_4(L) - \frac{\mathrm{i}\pi}{2}|$, and $\max |f_5(L) + \frac{\mathrm{i}\pi}{2}|$ as well as the maximum coefficient error

$$\max \left\{ |c_1(L) - 34|,\, |c_2(L) - 300|,\, |c_3(L) - 300|,\, |c_4(L) - 1|,\, |c_5(L) - 1| \right\},$$

where the maxima are formed over 10 results. In Figures 6.1 – 6.4 we show the maximum errors of the frequencies 0, $\pi/2$, $\pi/4$ as well as the maximum coefficient errors. The maximum error of the frequency $-\pi/2$ resp. $-\pi/4$ behaves as the maximum error of $\pi/2$ resp. $\pi/4$ and is omitted. In Figure 6.1 we show the maximum errors of the Algorithm 4.2 with complete SVD computed by the Matlab program "svd". The Figure 6.2 presents the maximum errors of the Algorithm 4.2 with partial SVD computed by the Matlab program "svds", which computes only the 5 largest singular values as well as the related partial SVD. The Figure 6.3 presents the maximum errors based on Remark 4.4 with partial SVD computed from (4.8). The Figure 6.4 shows the maximum errors of the fast Algorithm 5.1 with partial LBD computed by the Matlab program IRLBA. □

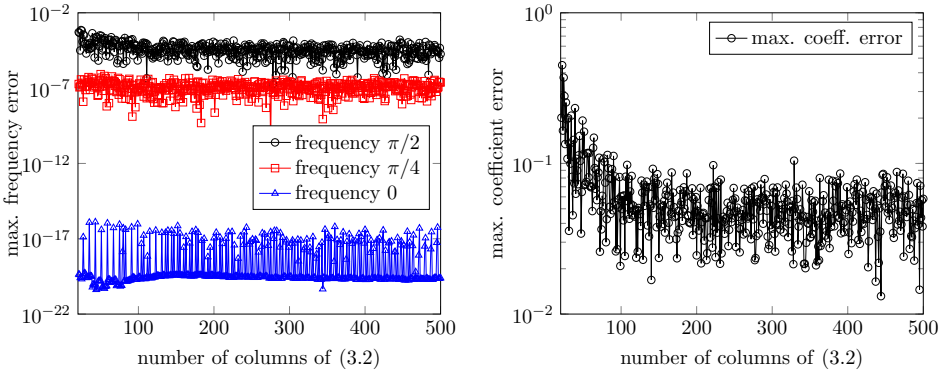

Figure 6.1: Maximum errors of Algorithm 4.2 with complete SVD for Example 6.1.

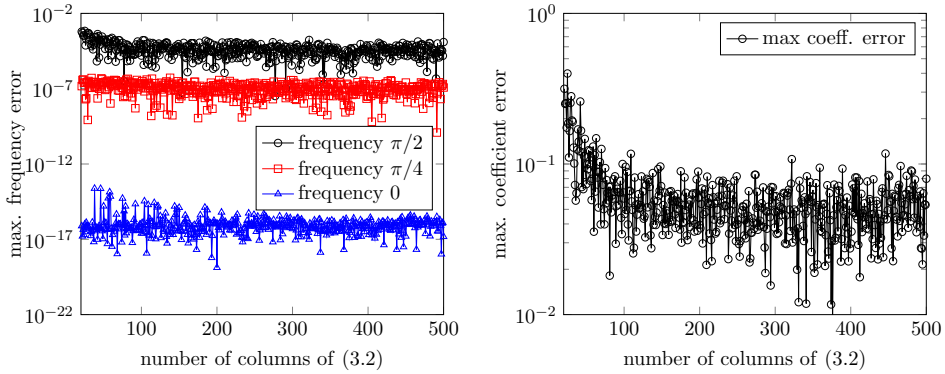

Figure 6.2: Maximum errors of Algorithm 4.2 with partial SVD for Example 6.1.

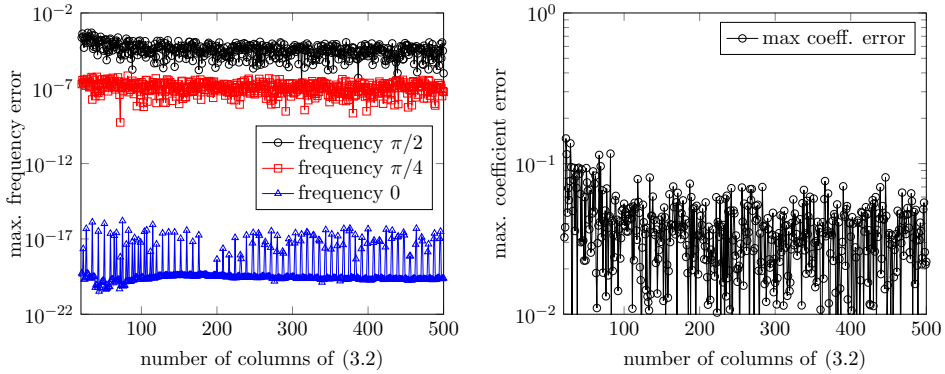

Figure 6.3: Maximum errors of Algorithm 4.2 with partial SVD based in Remark 4.4 for Example 6.1.

Example 6.2 We consider a bivariate exponential sum of order M that is a linear combination

$$h(\boldsymbol{x}) := \sum_{j=1}^{M} c_j \, \mathrm{e}^{\mathrm{i}\boldsymbol{f}_j \cdot \boldsymbol{x}} \quad (\boldsymbol{x} := (x_1, x_2)^{\mathrm{T}} \in \mathbb{R}^2) \tag{6.2}$$

of M complex exponentials with distinct complex coefficients $c_j \neq 0$ and distinct frequency vectors $\boldsymbol{f}_j := (f_{j,1}, f_{j,2})^{\mathrm{T}} \in [-\pi, \pi)^2$. In [24], a sparse Prony-like method was suggested. The main idea is based on a stepwise recovery of the frequencies. Here we use a sampling based on a special lattices. To this end, we consider the two time series $\{h_0^{(1)}, h_1^{(1)}, \ldots, h_{N-1}^{(1)}\}$

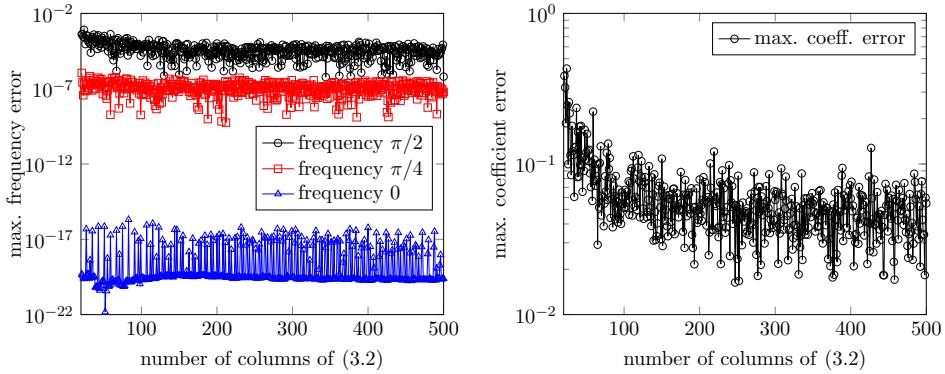

Figure 6.4: Maximum errors of Algorithm 5.1 with partial LBD (by a program of [1]) for Example 6.1.

and $\{h_0^{(2)}, h_1^{(2)}, \ldots, h_{N-1}^{(2)}\}$ with

$$h_k^{(1)} := h(\frac{k}{2}, \frac{k}{2}) = \sum_{j=1}^{M} c_j \, \mathrm{e}^{\frac{f_{j,1}+f_{j,2}}{2} k \mathrm{i}},$$

$$h_k^{(2)} := h(\frac{k}{3}, \frac{2k}{3}) = \sum_{j=1}^{M} c_j \, \mathrm{e}^{\frac{f_{j,1}+2f_{j,2}}{3} k \mathrm{i}}.$$

Since $\frac{f_{j,1}+f_{j,2}}{2} \in [-\pi, \pi)$ and $\frac{f_{j,1}+2f_{j,2}}{3} \in [-\pi, \pi)$, we recover these linear combinations of the frequencies by the univariate ESPRIT method. Now we use the fact that the coefficients c_j are distinct such that we can assign the different frequencies. We can pass on the requirement of distinct coefficients $c_j \neq 0$ by sampling on additional lines, such as $h(\alpha + k/2, k/2)$ with $\alpha \in \mathbb{R}$. We consider the bivariate exponential sum (6.2) of order $M = 4$ with the following parameters given in Table 6.1.

j	$f_{j,1}$	$f_{j,2}$	c_j
1	1.1	1.0	1.0
2	1.3	−1.2	5.0
3	−1.3	1.2	4.0
4	−1.1	−1.2	2.0

Table 6.1: Parameters of the bivariate exponential sum (6.2) of order 4.

If we work with exact sampling data of (6.2), then our algorithms recover

all parameters from a few sampling values, e.g. for $N = 10$.

Now we study noisy sampling data of (6.2). Therefore we add uniformly distributed noise in the range $[-1, 1]$ to each value of the time sequences $\{h_0^{(1)}, h_1^{(1)}, \ldots, h_{N-1}^{(1)}\}$ and $\{h_0^{(2)}, h_1^{(2)}, \ldots, h_{N-1}^{(2)}\}$. We show the results in Table 6.2, where error means the maximum absolute error of the frequency vectors. Time means the elapsed CPU time in seconds, measured with the Matlab command "cputime". We see the main advantage of the Algorithm 5.1, namely lower cost by almost the same accuracy as Algorithm 4.2 with complete SVD. □

N	L	Algorithm	error	time
50	25	4.2	3.907e-02	1.000e-02
50	25	5.1	1.279e-02	2.00e-02
100	50	4.2	3.991e-03	2.00e-02
100	50	5.1	6.136e-03	3.00e-02
500	250	4.2	2.403e-04	3.10e-02
500	250	5.1	1.136e-03	3.00e-02
1000	500	4.2	2.387e-04	9.30e-01
1000	500	5.1	3.431e-04	1.30e-01
5000	2500	4.2	1.872e-05	1.32e+02
5000	2500	5.1	1.570e-05	9.00e-01
10000	5000	4.2	4.164e-06	9.20e+02
10000	5000	5.1	1.533e-05	7.00e-01
50000	25000	4.2	–	–
50000	25000	5.1	4.014e-07	1.046e+01
100000	50000	4.2	–	–
100000	50000	5.1	2.583e-07	4.288e+01

Table 6.2: Errors and CPU times of Algorithms 4.2 and 5.1 for Example 6.2.

Example 6.3 Now we consider exponential sums (3.1) of large order M with uniformly distributed exponents $f_j \in \mathrm{i}\,[-\pi, \pi)$ and uniformly distributed coefficients $c_j \in [0, 1) + \mathrm{i}\,[0, 1)$. Assume that exact sampling data $h_k = h(k) \in \mathbb{C}$ $(k = 0, \ldots, N-1)$ are given. The relative maximum error

and the relative 2-error of the complex exponents are defined by

$$e_{max}(\boldsymbol{f}) := \frac{\max\limits_{j=1,\ldots,M} |f_j - \tilde{f}_j|}{\max\limits_{j=1,\ldots,M} |f_j|}, \quad e_2(\boldsymbol{f})^2 := \frac{\sum\limits_{j=1}^{M} |f_j - \tilde{f}_j|^2}{\sum\limits_{j=1}^{M} |f_j|^2},$$

where $\boldsymbol{f} := (f_j)_{j=1}^M$ and where $\tilde{f}_j$ are the exponents computed by Algorithm 5.1. Analogously, the relative errors of the coefficients are explained by

$$e_{max}(\boldsymbol{c}) := \frac{\max\limits_{j=1,\ldots,M} |c_j - \tilde{c}_j|}{\max\limits_{j=1,\ldots,M} |c_j|}, \quad e_2(\boldsymbol{c})^2 := \frac{\sum\limits_{j=1}^{M} |c_j - \tilde{c}_j|^2}{\sum\limits_{j=1}^{M} |c_j|^2},$$

where $\boldsymbol{c} := (c_j)_{j=1}^M$ and where $\tilde{c}_j$ are the coefficients computed by Algorithm 5.1. The following Table 6.3 contains the average results for 10 different tests. In addition we show after the values $e_{\max}(\boldsymbol{f})$ and $e_{\max}(\boldsymbol{c})$ as subscript the smallest and as superscript the minimum and the maximum of the 10 runs, respectively. We remark that we solve the Vandermonde-type system (3.4) in Algorithm 5.1 by the "Backslash" command of Matlab in order to compute the errors. Clearly, a faster iterative method based on the nonequispaced fast Fourier transform [18] can be used too. □

Acknowledgment

The first named author gratefully acknowledges the support by the German Research Foundation within the project PO 711/10–2. The authors would like to thank the referees for careful reading the paper and for comments that improved the presentation.

References

[1] J. Baglama, L. Reichel, Augmented implicitly restarted Lanczos bidiagonalization methods, SIAM J. Sci. Comput. 27 (1) (2005) 19 – 42.

[2] M. Ben-Or, P. Tiwari, A deterministic algorithm for sparse multivariate polynomial interpolation, in: Twentieth Annual ACM Symp. Theory Comput., ACM Press New York, 1988, pp. 301 – 309.

M	N	$e_{\max}(\boldsymbol{f})\,{}^{\max}_{\min}$	$e_{\max}(\boldsymbol{c})\,{}^{\max}_{\min}$	$e_2(\boldsymbol{f})$	$e_2(\boldsymbol{c})$
2^5	2^9	5.280e-13 $^{4.762e-12}_{7.256e-16}$	3.055e-08 $^{3.016e-07}_{4.848e-13}$	1.743e-13	1.150e-08
2^6	2^9	3.180e-11 $^{2.720e-10}_{3.693e-15}$	1.092e-06 $^{7.614e-06}_{2.306e-12}$	8.031e-12	2.857e-07
2^7	2^9	3.035e-03 $^{3.035e-02}_{3.603e-12}$	5.017e-02 $^{5.017e-01}_{2.094e-09}$	6.821e-04	1.257e-02
2^8	2^{10}	8.107e-03 $^{4.565e-02}_{1.007e-12}$	1.673e-01 $^{8.396e-01}_{6.733e-09}$	2.711e-03	1.045e-01
2^8	2^{11}	1.109e-10 $^{9.561e-10}_{8.251e-14}$	6.820e-06 $^{6.056e-05}_{9.570e-10}$	1.545e-11	1.021e-06
2^9	2^{11}	7.950e-03 $^{3.105e-02}_{3.942e-10}$	2.667e-01 $^{9.227e-01}_{1.008e-05}$	2.208e-03	1.624e-01
2^9	2^{12}	4.412e-09 $^{2.965e-08}_{2.109e-13}$	8.076e-04 $^{6.930e-03}_{2.014e-08}$	3.405e-10	8.571e-05
2^{10}	2^{12}	8.496e-03 $^{1.775e-02}_{4.823e-09}$	5.372e-01 $^{9.679e-01}_{1.106e-02}$	2.219e-03	2.918e-01
2^{10}	2^{13}	2.305e-08 $^{1.986e-07}_{4.308e-12}$	7.733e-02 $^{7.202e-01}_{6.172e-07}$	1.400e-09	6.734e-03
2^{11}	2^{13}	4.791e-03 $^{8.104e-03}_{5.396e-08}$	6.324e-01 $^{9.282e-01}_{3.627e-03}$	1.026e-03	2.927e-01
2^{11}	2^{14}	1.877e-04 $^{1.876e-03}_{2.374e-10}$	1.193e-01 $^{7.945e-01}_{6.997e-05}$	9.990e-06	7.661e-03

Table 6.3: Errors of Algorithm 5.1 for exponential sums of large order M in Example 6.3.

[3] G. Beylkin, L. Monzón, On approximations of functions by exponential sums, Appl. Comput. Harmon. Anal. 19 (2005) 17 – 48.

[4] Å. Björck, Numerical Methods for Least Squares Problems, SIAM, Philadelphia, 1996.

[5] K. Browne, S. Qiao, Y. Wei, A Lanczos bidiagonalization algorithm for Hankel matrices, Linear Algebra Appl. 430 (5-6) (2009) 1531 – 1543.

[6] F. Filbir, H. N. Mhaskar, J. Prestin, On the problem of parameter estimation in exponential sums, Constr. Approx. 35 (2012) 323 – 343.

[7] M. Giesbrecht, G. Labahn, W.-s. Lee, Symbolic-numeric sparse interpolation of multivariate polynomials, J. Symbolic Comput. 44 (2009) 943 – 959.

[8] G. Golub, W. Kahan, Calculating the singular values and pseudo-inverse of a matrix, J. Soc. Indust. Appl. Math. Ser. B Numer. Anal. 2 (1965) 205 – 224.

[9] G. H. Golub, P. Milanfar, J. Varah, A stable numerical method for

inverting shape from moments, SIAM J. Sci. Comput. 21 (1999) 1222 – 1243.

[10] G. H. Golub, C. F. Van Loan, Matrix Computations, third ed., Johns Hopkins Univ. Press, Baltimore, 1996.

[11] N. Golyandina, A. Zhigljavsky, Singular Spectrum Analysis for Time Series, Springer, Heidelberg, 2013.

[12] W. Hackbusch, Tensor Spaces and Numerical Tensor Calculus, Springer, Berlin, 2012.

[13] G. Heinig, K. Rost, Algebraic Methods for Toeplitz-like Matrices and Operators, Akademie-Verlag, Berlin, 1984.

[14] N. J. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, Philadelphia, 2002.

[15] N. J. Higham, Functions of Matrices. Theory and Computation, SIAM, Philadelphia, 2008.

[16] R. A. Horn, C. R. Johnson, Matrix Analysis, second ed., Cambridge Univ. Press, Cambridge, 2013.

[17] Y. Hua, T. K. Sarkar, Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise, IEEE Trans. Acoust. Speech Signal Process. 38 (1990) 814 – 824.

[18] J. Keiner, S. Kunis, D. Potts, Using NFFT3 - a software library for various nonequispaced fast Fourier transforms, ACM Trans. Math. Software 36 (2009) Article 19, 1 – 30.

[19] A. Korobeynikov, Computation- and space-efficient implementation of SSA, Stat. Interface 3 (2010) 357 – 368.

[20] R. Larsen, Lanczos bidiagonalization with partial reorthogonalization, Ph.D. thesis, Univ. Aarhus, 1998.

[21] D. G. Manolakis, V. K. Ingle, S. M. Kogon, Statistical and Adaptive Signal Processing, McGraw-Hill, Boston, 2005.

[22] V. Pereyra, G. Scherer, Exponential Data Fitting and its Applications, Bentham Sci. Publ., Sharjah, 2010.

[23] T. Peter, G. Plonka, A generalized prony method for reconstruction of sparse sums of eigenfunctions of linear operators, Inverse Probl. 29 (2013) Article 025001.

[24] D. Potts, M. Tasche, Parameter estimation for multivariate exponential sums, Electron. Trans. Numer. Anal. 40 (2013) 204 – 224.

[25] D. Potts, M. Tasche, Parameter estimation for nonincreasing exponential sums by Prony-like methods, Linear Algebra Appl. 439 (2013) 1024 – 1039.

[26] D. Potts, M. Tasche, Sparse polynomial interpolation in Chebyshev bases, Linear Algebra Appl. 441 (2014), 61 – 87.

[27] R. Roy, T. Kailath, ESPRIT—estimation of signal parameters via rotational invariance techniques, IEEE Trans. Acoust. Speech Signal Process. 37 (1989) 984 – 994.

[28] T. K. Sarkar, O. Pereira, Using the matrix pencil method to estimate the parameters of a sum of complex exponentials, IEEE Antennas Propag. 37 (1995) 48 – 55.

[29] H. D. Simon, H. Zha, Low-rank matrix approximation using the Lanczos bidiagonalization process with applications, SIAM J. Sci. Comput. 21 (6) (2000) 2257 – 2274.

[30] C. F. Van Loan, Computational Frameworks for the Fast Fourier Transform, SIAM, Philadelphia, 1992.

[31] K. Wu, H. Simon, Thick-restart Lanczos method for large symmetric eigenvalue problems, SIAM J. Matrix Anal. Appl. 22 (2) (2000) 602 – 616.