GENERALIZED RYBICKI PRESS ALGORITHM

SIVARAM AMBIKASARAN

$\mathcal{O}(N)$ direct solver and determinant computation of general semi-separable matrices

Abstract. This article discusses a more general and numerically stable Rybicki Press algorithm, which enables inverting and computing determinants of covariance matrices, whose elements are sums of exponentials. The algorithm is true in exact arithmetic and relies on introducing new variables and corresponding equations, thereby converting the matrix into a banded matrix of larger size. Linear complexity banded algorithms for solving linear systems and computing determinants on the larger matrix enable linear complexity algorithms for the initial semi-separable matrix as well. Benchmarks provided illustrate the linear scaling of the algorithm.

Key words.

AMS subject classifications. 15A23, 15A15, 15A09

1. Introduction.

Large dense covariance matrices arise in a wide range of applications in computational statistics and data analysis. Storing and performing numerical computations on such large dense matrices is computationally intractable. However, most of these large dense matrices are structured (either in exact arithmetic or finite arithmetic), which can be exploited to construct fast algorithms. One such class of data sparse matrices are semi-separable matrices, which have raised a lot of interest and have been studied in detail across a wide range of applications including integral equations [1–3] and computational statistics [4–8]. For a detailed bibliography on semi-separable matrices, the reader is referred to Vandebril et al. [9]. Throughout the literature, there are slightly different definitions of semi-separable matrices. In this article, we will be working with the following definition:

Defn 1.1. *$A \in \mathbb{R}^{N\times N}$ is termed a semi-separable matrix with semi-separable rank p, if it can be written as*

$$A = D + \mathit{triu}(B_p) + \mathit{tril}(C_p) \tag{1.1}$$

where D is a diagonal matrix, B_p, C_p are rank p matrices, $\mathit{triu}(B_p)$ denotes the upper triangular part of B_p and $\mathit{tril}(C_p)$ denotes the lower triangular part of C_p.

Fast algorithms for solving semi-separable linear systems exists and the reader is referred to some of these references [10–15] and the references therein. In this article, we propose a new $\mathcal{O}(N)$ direct solver and determinant computation for semi-separable matrices.

The main contributions of this article include:

- A new $\mathcal{O}(N)$ direct solver for semi-separable matrices is obtained by embedding the semi-separable matrix into a larger banded matrix.
- The determinant of these semi-separable matrix is shown to equal to the determinant of the larger banded matrix, thereby enabling computing determinants of these semi-separable matrices at a computational cost of $\mathcal{O}(N)$. This is the first algorithm for computing the determinants for a general semi-separable matrix.
- A numerically stable generalized Rybicki Press algorithm is derived using these ideas. To be specific, fast direct solver and stable, fast determinant computation (both scaling as $\mathcal{O}(n)$) for covariance matrix of the form:

$$A_{ij} = \sum_{l=1}^{p} \alpha_l \exp\left(-\beta_l |t_i - t_j|\right) \tag{1.2}$$

  where $i, j \in \{1, 2, \ldots, n\}$, the points t_i are distinct and are distributed on an interval. The covariance matrix in Equation (1.2) is frequently encountered in computational statistics in the context of Continuous time AutoRegressive-Moving-Average (abbreviated as CARMA) models [16–18].
- Another advantage of this algorithm from a practical view-point is that the algorithm relies only on sparse linear algebra and thereby can easily use the existing mature sparse linear algebra libraries.

The algorithm proposed in this article relies on two key ideas: (i) The covariance matrix in Equation (1.2) is a semi-separable matrix with semi-separable rank p; (ii) Fast algorithms for solving and computing determinants for such semi-separable matrices can be obtained by embedding the dense covariance matrix in a large sparse banded matrix. The article also discusses how to perform the entire procedure in a numerically stable fashion. The implementation is made available at `https://github.com/sivaramambikasaran/ESS` [19] under the license provided by New York University.

Acknowledgements: The author would like to thank Christopher S. Kochanek for initiating the conversation on generalized Rybicki Press algorithm and David W. Hogg for putting in touch with Christopher S. Kochanek. The research was supported in part by the NYU-AIG Partnership on Innovation for Global Resilience under grant number A2014-005. The author was also supported in part by the Applied Mathematical Sciences Program of the U.S. Department of Energy under Contract DEFGO288ER25053, Office of the Assistant Secretary of Defense for Research and Engineering and AFOSR under NSSEFF Program Award FA9550-10-1-0180.

2. Sparse embedding of semi-separable matrix with semi-separable rank 1.

To motivate the general idea, we will first look at the sparse embedding for a 4×4 semi-separable matrix, whose semi-separable rank is 1. The matrix A is as shown in Equation (2.1).

$$A = \begin{bmatrix} a_{11} & u_1v_2 & u_1v_3 & u_1v_4 \\ u_1v_2 & a_{22} & u_2v_3 & u_2v_4 \\ u_1v_3 & u_2v_3 & a_{33} & u_3v_4 \\ u_1v_4 & u_2v_4 & u_3v_4 & a_{44} \end{bmatrix} \tag{2.1}$$

And the corresponding linear system is $Ax = b$, where $b = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix}^T$

Introduce the following variables:

$$r_4 = v_4x_4 \tag{2.2}$$
$$r_3 = v_3x_3 + r_4 \tag{2.3}$$
$$r_2 = v_2x_2 + r_3 \tag{2.4}$$

$$l_1 = u_1x_1 \tag{2.5}$$
$$l_2 = u_2x_2 + l_1 \tag{2.6}$$
$$l_3 = u_3x_3 + l_2 \tag{2.7}$$

Introducing the variables the linear system $Ax = b$ is now of the form

$$a_{11}x_1 + u_1r_2 = b_1 \tag{2.8}$$
$$v_2l_1 + a_{22}x_2 + u_2r_3 = b_2 \tag{2.9}$$
$$v_3l_2 + a_{33}x_3 + u_3r_4 = b_3 \tag{2.10}$$
$$v_4l_3 + a_{44}x_4 = b_4 \tag{2.11}$$

The extended linear system (after appropriate ordering of equations and unknowns) is then of the form

$$
\begin{bmatrix}
a_{11} & u_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
u_1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & -1 & 0 & v_2 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & v_2 & a_{22} & u_2 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & u_2 & 0 & -1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & -1 & 0 & v_3 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & v_3 & a_{33} & u_3 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & u_3 & 0 & -1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & v_4\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & v_4 & a_{44}
\end{bmatrix}
\begin{bmatrix} x_1\\ r_2\\ l_1\\ x_2\\ r_3\\ l_2\\ x_3\\ r_4\\ l_3\\ x_4 \end{bmatrix}
=
\begin{bmatrix} b_1\\ 0\\ 0\\ b_2\\ 0\\ 0\\ b_3\\ 0\\ 0\\ b_4 \end{bmatrix}
\tag{2.12}
$$

Note that Equation (2.12) is a banded matrix of bandwidth 2 and has a sparsity structure even within the band. In general, let A be an $N \times N$ semi-separable matrix, with the semi-separability rank 1 as written in Equation (2.13).

$$
A(i,j) = \begin{cases} a_{ii} & \text{if } i = j\\ u_j v_i & \text{if } i > j\\ u_i v_j & \text{if } i < j \end{cases}
\tag{2.13}
$$

where $i, j \in \{1, 2, \ldots, N\}$. One would then need to add the variables $r_2, r_3, \ldots, r_N$ and $l_1, l_2, \ldots, l_{N-1}$, where $r_N = v_N x_N$, $l_1 = u_1 x_1$ and

$$
r_k = v_k x_k + r_{k+1} \tag{2.14}
$$

$$
l_k = u_k x_k + l_{k-1} \tag{2.15}
$$

where $k \in \{2, \ldots, N-1\}$. Hence, we have a total of $3N-2$ variables and $3N-2$ equations. Therefore, the extended matrix will be a $(3N-2) \times (3N-2)$ banded matrix, whose bandwidth is 2. This is illustrated pictorially for a 8×8 matrix in Figure 2.1.

3. Sparse embedding of a general semi-separable matrix. Let A be a $N \times N$ matrix, whose semi-separable rank is p, i.e., we have

$$
A(i,j) = \begin{cases} a_{ii} & \text{if } i = j\\ \displaystyle\sum_{l=1}^{p} u_j^{(l)} v_i^{(l)} & \text{if } i > j\\ \displaystyle\sum_{l=1}^{p} u_i^{(l)} v_j^{(l)} & \text{if } i < j \end{cases}
\tag{3.1}
$$

where $i, j \in \{1, 2, \ldots, N\}$. We then add the following variables $r_2^{(p)}, r_3^{(p)}, \ldots, r_N^{(p)}$ and $l_1^{(p)}, l_2^{(p)}, \ldots, l_{N-1}^{(p)}$ as before. However, unlike earlier and not surprisingly, these new variables $r_i^{(p)}$'s and $l_j^{(p)}$'s will be vectors of length p. Let $U_k^{(p)} = \begin{bmatrix} u_k^{(1)} & u_k^{(2)} & u_k^{(3)} & \cdots & u_k^{(p)} \end{bmatrix}$ and $V_k^{(p)} = \begin{bmatrix} v_k^{(1)} & v_k^{(2)} & v_k^{(3)} & \cdots & v_k^{(p)} \end{bmatrix}$. We then have the following relations for the additional vector variables.

$$
r_N^{(p)} = V_N^T x_N \tag{3.2}
$$

$$
l_1^{(p)} = U_1^T x_1 \tag{3.3}
$$

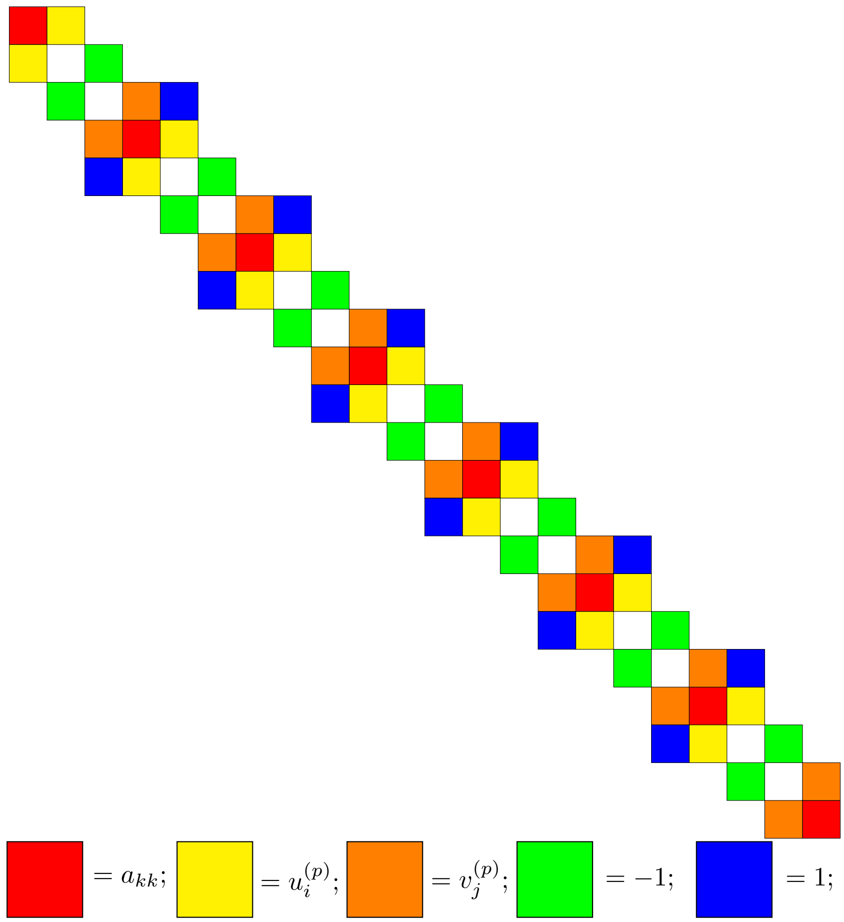


FIG. 2.1. *Pictorial description of the extended sparse matrix obtained from a rank 1 semi-separable matrix where $N = 8$. The color code is as follows:*

and

$$r_k^{(p)} = V_k^T x_k + r_{k+1}^{(p)} \tag{3.4}$$

$$l_k^{(p)} = U_k^T x_k + l_{k-1}^{(p)} \tag{3.5}$$

where $k \in \{2, \ldots, N-1\}$. Hence, we now have $(2p+1)N - 2p$ variables (this includes the N x_k's, $N-1$ vector variables $r_k^{(p)}$ and $l_k^{(p)}$ of length p) and $(2p+1)N-2p$ equations relating them. Therefore, we end up with a $((2p+1)N-2p)\times((2p+1)N-2p)$ extended sparse matrix, whose bandwidth is $(2p+1)$. This is illustrated in Figure 3.1 for 10×10 semi-separable matrix, whose semi-separable rank is 4.

The computational complexity of the algorithm clearly scales as $\mathcal{O}(N)$, since the extended sparse matrix has a bandwidth of $\mathcal{O}(p)$ and the matrix of size $\mathcal{O}(pN) \times \mathcal{O}(pN)$. It is also possible to analyze the scaling with respect to the semi-separable rank p, though this is of little practical relevance since $p = \mathcal{O}(1)$ for most interesting semi-separable matrices. A detailed analysis shows that the computational complexity of the algorithm is $\mathcal{O}(p^2N)$. Numerical benchmarks presented in Section 7 validate the scaling of the algorithm.

4. Determinant of extended sparse matrix. CLAIM 4.1. *The determinant of the extended sparse matrix is the*

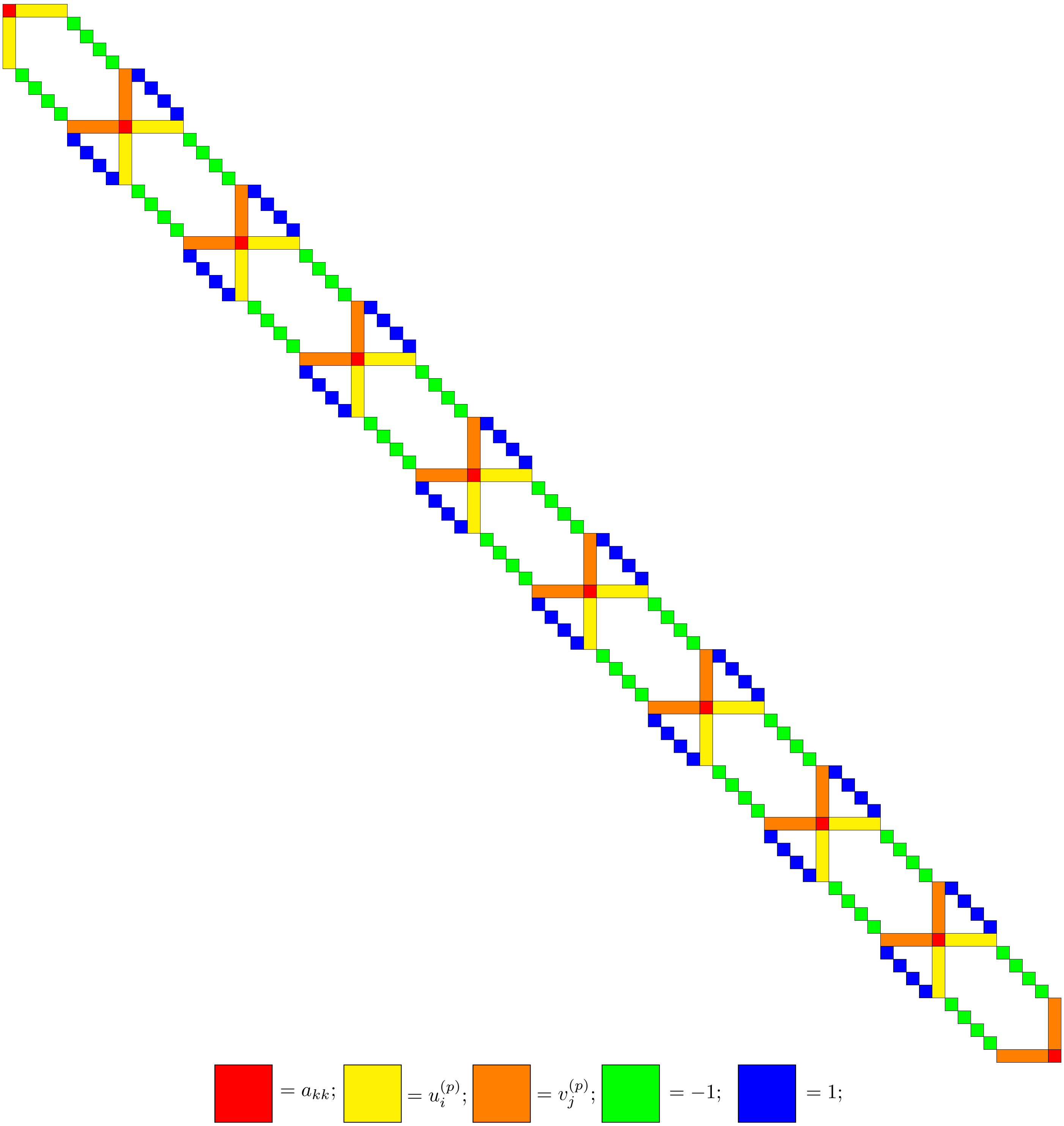


FIG. 3.1. *Pictorial description of the extended sparse matrix where $N = 10$ and $p = 4$. The color code is as follows:*

same as the determinant of the original dense matrix up to a sign.

The extended system, denoted by A_{ex} on appropriate reordering of rows and columns can be written as

$$P_1 A_{ex} P_2 \begin{bmatrix} l_1^{(p)} \\ l_2^{(p)} \\ \vdots \\ l_{N-1}^{(p)} \\ r_2^{(p)} \\ r_3^{(p)} \\ \vdots \\ r_N^{(p)} \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} I_1 & 0 & U_a \\ 0 & I_2 & V_a \\ V_b & U_b & D \end{bmatrix} \begin{bmatrix} l_1^{(p)} \\ l_2^{(p)} \\ \vdots \\ l_{N-1}^{(p)} \\ r_2^{(p)} \\ r_3^{(p)} \\ \vdots \\ r_N^{(p)} \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \tag{4.1}$$

where P_1, P_2 are permutation matrices, the matrix I_1 is a highly sparse lower-triangular matrix with 1's on the diagonal and -1's at a few places in the lower-triangular part (the precise location is unimportant for determinant computations as we will see later), the matrix I_2 is a highly sparse upper-triangular matrix with 1's on the diagonal and -1's at a few places in the upper-triangular part and D is a diagonal matrix. The first set of rows, i.e., $\begin{bmatrix} I_1 & 0 & U_a \end{bmatrix}$, correspond to adding the variables $l_k^{(p)}$, i.e., $l_k^{(p)} = U_k^T x_k + l_{k-1}^{(p)}$. The next set of rows, i.e., $\begin{bmatrix} 0 & I_2 & V_a \end{bmatrix}$, correspond to adding the variables $r_k^{(p)}$, i.e., $r_k^{(p)} = V_k^T x_k + r_{k+1}^{(p)}$. The last set of rows, i.e., $\begin{bmatrix} V_b & U_b & D \end{bmatrix}$, correspond to the initial set of equations with the $l_k^{(p)}$'s and $r_k^{(p)}$'s introduced. We then have

$$\det(P_1 A_{ex} P_2) = \underbrace{\det\left(\begin{bmatrix} I_1 & 0 & U_a \\ 0 & I_2 & V_a \\ V_b & U_b & D \end{bmatrix}\right) = \det\left(\begin{bmatrix} I_1 & 0 \\ 0 & I_2 \end{bmatrix}\right) \det\left(D - \begin{bmatrix} U_a \\ V_a \end{bmatrix} \begin{bmatrix} I_1 & 0 \\ 0 & I_2 \end{bmatrix}^{-1} \begin{bmatrix} U_b & V_b \end{bmatrix}\right)}_{\text{Block determinant formula}} \tag{4.2}$$

Now note that $\det(I_1) = 1 = \det(I_2)$, due to the fact that I_1 and I_2 are triangular matrices with 1's on the diagonal. Hence,

$$\det\left(\begin{bmatrix} I_1 & 0 \\ 0 & I_2 \end{bmatrix}\right) = \det(I_1)\det(I_2) = 1 \times 1 = 1 \tag{4.3}$$

Further, note that the matrix $D - \begin{bmatrix} U_a \\ V_a \end{bmatrix} \begin{bmatrix} I_1 & 0 \\ 0 & I_2 \end{bmatrix}^{-1} \begin{bmatrix} U_b & V_b \end{bmatrix}$ is the Schur complement obtained by eliminating the variables $l_i^{(p)}$, $r_i^{(p)}$ and hence is the initial dense matrix A we began with. Hence, we have

$$\det(A_{ex}) = \pm \det(A) \tag{4.4}$$

5. Reinterpretation of Rybicki Press algorithm in terms of sparse embedding. We will first naively reinterpret the Rybicki Press algorithm in terms of the extended sparse matrix algebra. Recall that the Rybicki Press algorithm [20] inverts a correlation matrix A given by Equation (5.1).

$$A(i,j) = \exp\left(-\beta|t_i - t_j|\right) \tag{5.1}$$

where t_i's lies on an interval and are monotone. The original Rybicki Press algorithm relies on the fact that the inverse of A happens to be a tridiagonal matrix. The key ingredient of their algorithm is the following property of exponentials:

$$\exp\left(\beta(t_i - t_j)\right)\exp\left(\beta(t_j - t_k)\right) = \exp\left(\beta(t_i - t_k)\right) \tag{5.2}$$

In our sparse interpretation as well, we will use this property to recognize that the matrix A is a semi-separable matrix, whose semi-separable rank is 1. This can be seen by setting $u_k = \exp(\beta t_k)$ and $v_k = \exp(-\beta t_k)$. This then gives us $(i < j)$ that $A(i,j) = u_i v_j = \exp(\beta t_i)\exp(-\beta t_j) = \exp(\beta(t_i - t_j))$ and similarly for $i > j$. This shows that the matrix A is semi-separable with semi-separable rank 1. Hence, we can mimic the same approach as in the earlier sections to obtain an $\mathcal{O}(N)$ algorithm. However, there is an issue that needs to be addressed from a numerical perspective. If the t_i's are spread over a large interval, then u_i is exponentially large, while v_i is exponentially small, and hence embedding into a sparse matrix as such could prove to be a catastrophic leading to underflow and overflow of the relevant entries. This issue though can be circumvented by a suitable analytic preconditioning, by an appropriate change of variables. This is illustrated for a 4×4 linear system. We will use the notation t_{ij} to denote $|t_i - t_j|$. The linear equation is

$$\begin{bmatrix} 1 & \exp(-\beta t_{12}) & \exp(-\beta t_{13}) & \exp(-\beta t_{14}) \\ \exp(-\beta t_{12}) & 1 & \exp(-\beta t_{23}) & \exp(-\beta t_{24}) \\ \exp(-\beta t_{13}) & \exp(-\beta t_{23}) & 1 & \exp(-\beta t_{34}) \\ \exp(-\beta t_{14}) & \exp(-\beta t_{24}) & \exp(-\beta t_{34}) & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \tag{5.3}$$

Now lets introduce the additional variables as follows:

$$r_4 = x_4 \tag{5.4}$$
$$r_3 = x_3 + \exp(-\beta t_{34}) r_4 \tag{5.5}$$
$$r_2 = x_2 + \exp(-\beta t_{23}) r_3 \tag{5.6}$$

$$l_2 = x_1 \exp(-\beta t_{12}) \tag{5.7}$$
$$l_3 = (x_2 + l_2)\exp(-\beta t_{23}) \tag{5.8}$$
$$l_4 = (x_3 + l_3)\exp(-\beta t_{34}) \tag{5.9}$$

The equations then become

$$x_1 + \exp(-\beta t_{12}) r_2 = b_1 \tag{5.10}$$
$$l_2 + x_2 + \exp(-\beta t_{23}) r_3 = b_2 \tag{5.11}$$
$$l_3 + x_3 + \exp(-\beta t_{24}) r_4 = b_3 \tag{5.12}$$
$$l_4 + x_4 = b_4 \tag{5.13}$$

Embedding this in an extended sparse matrix, we obtain

(5.14)

$$\begin{bmatrix} 1 & \exp(-\beta t_{12}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \exp(-\beta t_{12}) & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & \exp(-\beta t_{23}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & \exp(-\beta t_{23}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \exp(-\beta t_{23}) & \exp(-\beta t_{23}) & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & \exp(-\beta t_{34}) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & \exp(-\beta t_{34}) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \exp(-\beta t_{34}) & \exp(-\beta t_{34}) & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ r_2 \\ l_1 \\ x_2 \\ r_3 \\ l_2 \\ x_3 \\ r_4 \\ l_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ 0 \\ 0 \\ b_2 \\ 0 \\ 0 \\ b_3 \\ 0 \\ 0 \\ b_4 \end{bmatrix}$$

Note that the sparsity pattern of the matrix is the same as before, which is to be expected, since all we have done essentially is to scale elements appropriately and hence the zero fill-ins remain the same.

6. Numerically stable generalized Rybicki Press. The same idea carries over the generalized Rybicki Press algorithm, i.e., if we consider a CARMA(p,q) process which has the covariance matrix given by

$$K(r) = \begin{cases} d & \text{if } r = 0 \\ \displaystyle\sum_{l=1}^{p} \alpha_l(p,q)\exp(-\beta_l r) & \text{if } r > 0 \end{cases} \tag{6.1}$$

then it immediately follows that that the matrix is semi-separable with semi-separable rank being p. To avoid numerical overflow and underflow, as shown in the previous section, appropriate sets of variables need to be introduced. Let

$$\alpha = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_p \end{bmatrix}^T \tag{6.2}$$

and

$$\gamma_k = \begin{bmatrix} \exp(-\beta_1 t_{k,k+1}) & \exp(-\beta_2 t_{k,k+1}) & \cdots & \exp(-\beta_p t_{k,k+1}) \end{bmatrix}^T \tag{6.3}$$

Now introduce the variables

$$r_k = \alpha x_k + D_{k,k+1} r_{k+1} \tag{6.4}$$

$$l_k = \gamma_{k-1} x_{k-1} + D_{k-1,k} l_{k-1} \tag{6.5}$$

where $k \in \{2, 3, \ldots, N\}$, with $r_{N+1} = l_1 = 0$ and $D_{k,k+1}$ is a $p \times p$ diagonal matrix, with its diagonal being γ_k. The initial equations become

$$\alpha^T l_k + d x_k + \gamma_k^T r_{k+1} = b_k \tag{6.6}$$

where $k \in \{1, 2, \ldots, N\}$. Now form the extended sparse matrix using the variables x_k, l_k and r_k, with the equations being Equations (6.4), (6.5), (6.6). The sparsity pattern of the extended sparse matrix is the same and hence the computational complexity scales as $\mathcal{O}(N)$.

7. Numerical benchmarks. We present a few numerical benchmarks illustrating the scaling of the algorithm and the error. In all these benchmarks, the semi-separable matrix is of the form

$$A(i,j) = \begin{cases} d & \text{if } i = j \\ \displaystyle\sum_{l=1}^{p} \alpha_l \exp\left(-\beta_l |t_i - t_j|\right) & \text{if } i \neq j \end{cases} \tag{7.1}$$

where the t_i's lie on a on-dimensional manifold and are sorted in increasing fashion. Apart from the time taken for the assembly, factorization and solve, the absolute error, i.e., $\|Ax - b\|_\infty$ is also presented. For the purposes of benchmark, t_i's are chosen at random from the interval $[0, 20]$ and then sorted; α_l's, β_l's are chosen at random from the interval $[0, 2]$; and d is set equal to $1 + \displaystyle\sum_{l=1}^{p} \alpha_l$. Throughout the benchmarks the original dense matrix will be referred to as A, while the corresponding extended sparse matrix will be referred to as A_{ex}.

Table 7.1
Scaling of the algorithm with system size N for a fixed semi-separable rank $p = 5$. The time taken is reported in seconds.

System size	Time taken in seconds						Maximum error	
N	Assembly		Factorize		Solve		in residual	
	Usual	Fast	Usual	Fast	Usual	Fast	Usual	Fast
500	1.55×10^{-2}	1.15×10^{-3}	1.2×10^{-2}	8×10^{-3}	2.33×10^{-4}	1.36×10^{-3}	2×10^{-14}	2.2×10^{-15}
1000	4.92×10^{-2}	1.73×10^{-3}	9.16×10^{-2}	1.55×10^{-2}	8.62×10^{-4}	2.02×10^{-3}	4×10^{-14}	3.8×10^{-15}
2000	1.88×10^{-1}	3.28×10^{-3}	6.43×10^{-1}	3.08×10^{-2}	2.80×10^{-3}	4.41×10^{-3}	9×10^{-14}	5.6×10^{-15}
5000	$1.15 \times 10^{+0}$	9.11×10^{-3}	$9.36 \times 10^{+0}$	8.31×10^{-2}	1.44×10^{-2}	1.04×10^{-2}	2×10^{13}	6.4×10^{-15}
10000	$4.76 \times 10^{+0}$	2.05×10^{-2}	$7.19 \times 10^{+1}$	1.67×10^{-1}	5.81×10^{-2}	2.05×10^{-2}	3×10^{-13}	8.0×10^{-15}
20000	–	4.91×10^{-2}	–	3.33×10^{-1}	–	4.29×10^{-2}	–	1.0×10^{-14}
50000	–	1.16×10^{-1}	–	8.38×10^{-1}	–	1.08×10^{-1}	–	1.5×10^{-14}
100000	–	2.16×10^{-1}	–	$1.68 \times 10^{+0}$	–	2.13×10^{-1}	–	1.8×10^{-14}
200000	–	4.41×10^{-1}	–	$3.38 \times 10^{+0}$	–	4.25×10^{-1}	–	2.6×10^{-14}
500000	–	$1.33 \times 10^{+0}$	–	$8.50 \times 10^{+0}$	–	$1.07 \times 10^{+0}$	–	3.4×10^{-14}
1000000	–	$2.70 \times 10^{+0}$	–	$1.76 \times 10^{+1}$	–	$2.33 \times 10^{+0}$	–	3.9×10^{-14}

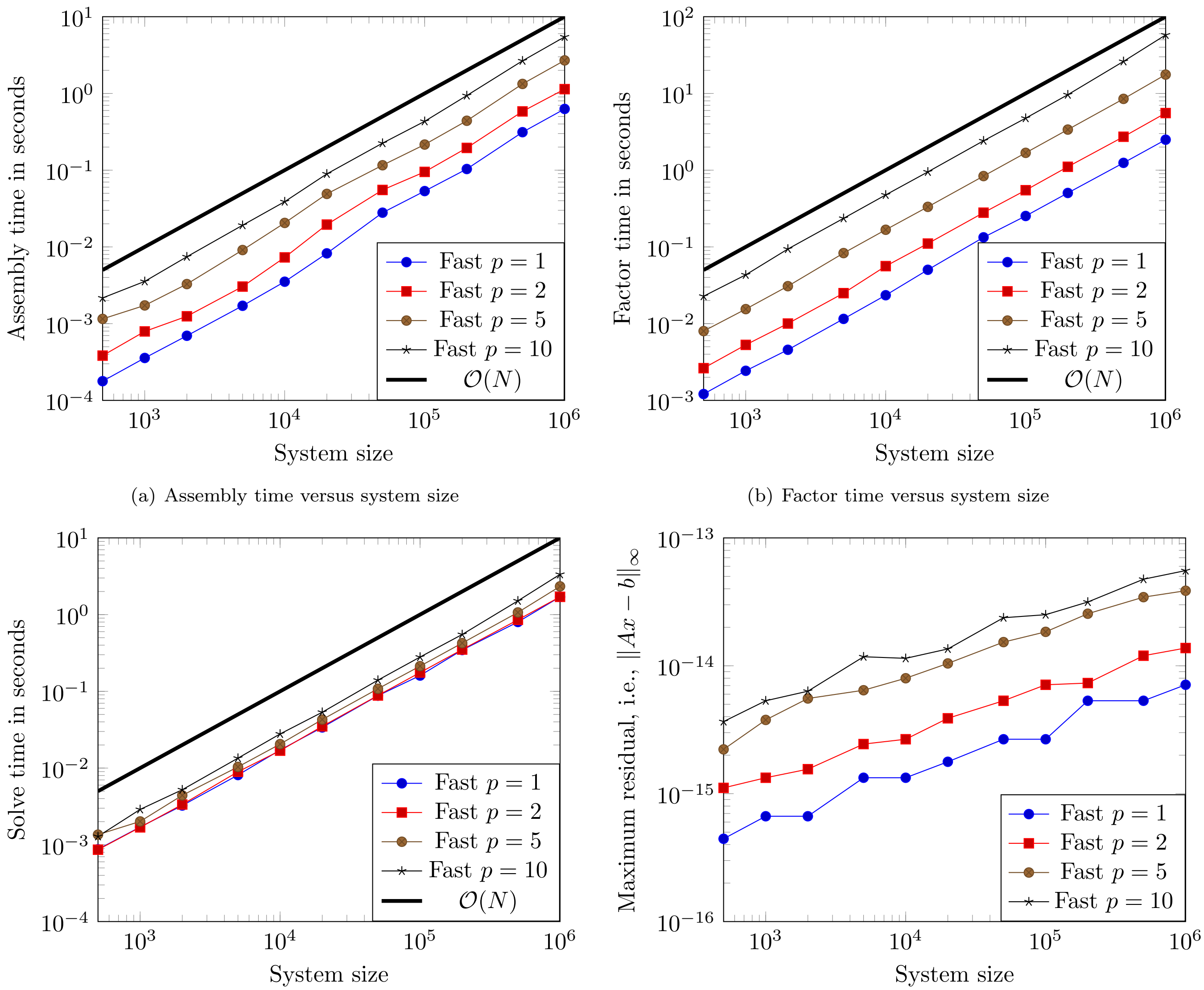
7.1. Benchmark 1. In this benchmark, we illustrate the linear scaling of the algorithm with the number of unknowns N for different choices of p. The extended sparse linear system, i.e., $A_{ex}x_{ex} = b_{ex}$ is solved using the sparse LU factorization (SparseLU) in Eigen [21]. This is compared with the partial pivoted LU algorithm (PartialPivLU) in Eigen [21] to solve the initial dense linear system $Ax = b$. Table 7.1 shows the scaling of the algorithm and the maximum error in the residual for a fixed semi-separable rank of $p = 5$.

Figure 7.1 illustrates the scaling of the assembly, factorization and solve time with system size.

- Assembly time - Time taken to assemble the extended sparse matrix.
- Factorization time - Time taken to factorize the extended sparse matrix.
- Solve time - Time taken to solve the linear system (once the factorization has been obtained).

7.2. Benchmark 2. In this benchmark, we illustrate the scaling of the time taken (assembly, factorization and solve) for algorithm with p, the number of exponentials added (equivalently the semi-separable rank).

8. Conclusion. The article discusses a numerically stable, generalized Rybicki Press algorithm, which relies on the fact that a semi-separable matrix can be embedded into a larger banded matrix. This enables $\mathcal{O}(N)$ inversion and determinant computation of covariance matrices, whose entries are sums of exponentials. This publication also serves to formally announce the release of the implementation of the extended sparse semi-separable factorization and the generalized Rybicki Press algorithm. The implementation is in C++ and is made available at `https://github.com/sivaramambikasaran/ESS` [19] under the license provided by New York University.



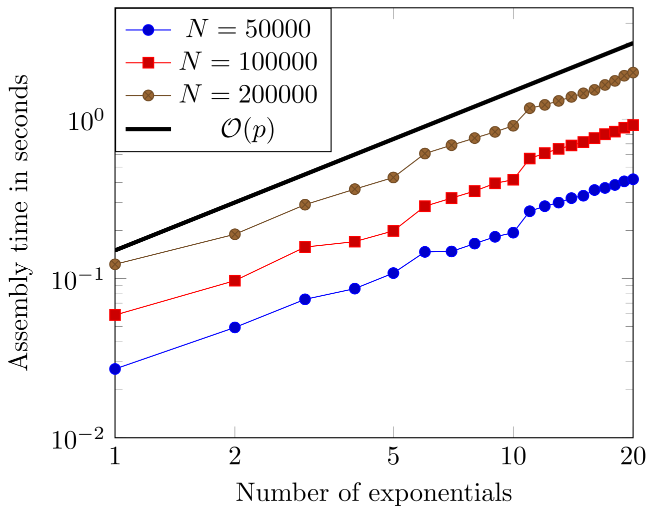
(a) Assembly time versus system size

(b) Factor time versus system size

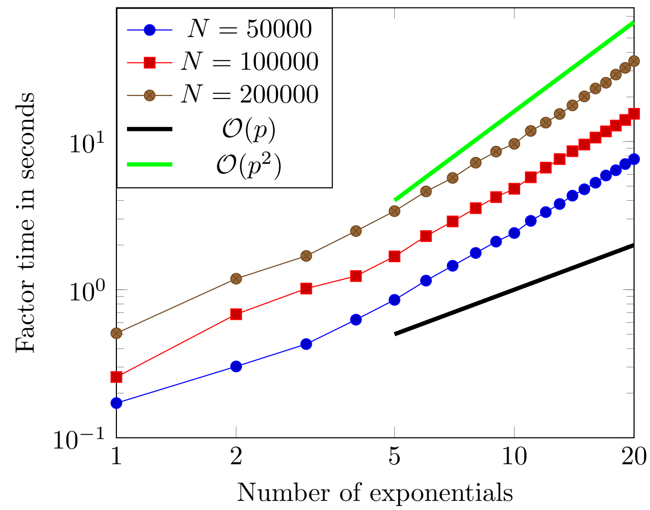
(c) Solve time versus system size

(d) Error versus system size

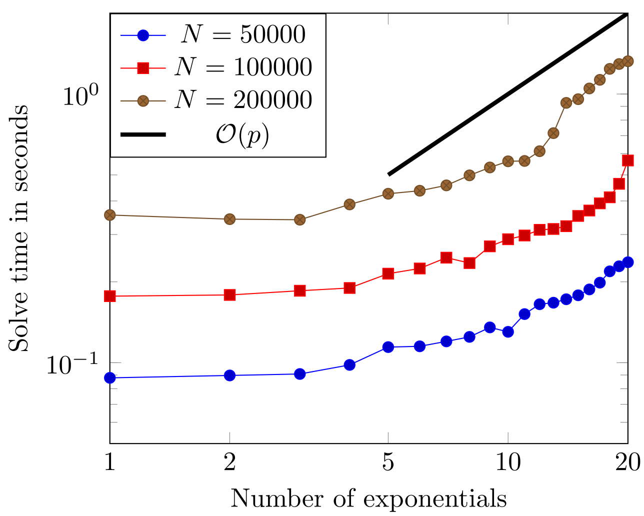
FIG. 7.1. *Scaling of the algorithm with system size. From the benchmarks, it is clear that the computational cost for the fast algorithm scales as $\mathcal{O}(N)$ for assembly, factorization and solve stages, where N is the number of unknowns. The maximum residual is less than 10^{-13} even for a system with million unknowns.*



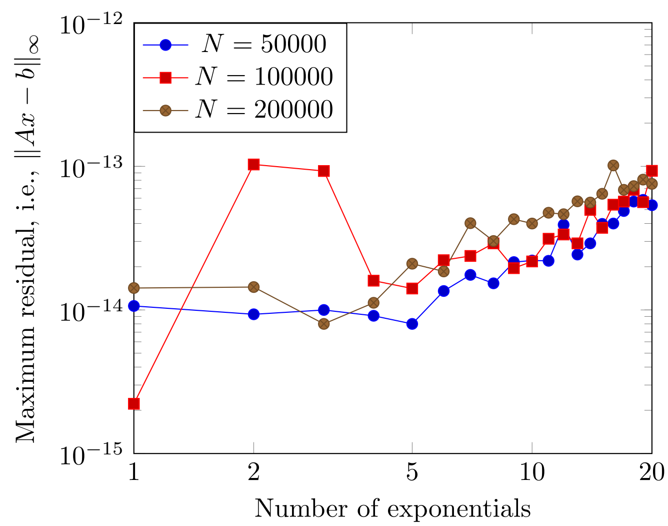
(a) Assembly time versus number of exponentials



(b) Factor time versus number of exponentials



(c) Solve time versus number of exponentials



(d) Error versus number of exponentials

Fig. 7.2. *Scaling of the fast algorithm with the number of exponentials added. From the benchmarks, it is clear that the computational cost for the fast algorithm scales as $\mathcal{O}(p)$ for assembly and solve stages, while it scales as $\mathcal{O}(p^2)$ for the factorization stage, where p is the number of exponentials (equivalently the semi-separable rank). The maximum residual is less than 10^{-13} almost always.*

References.

[1] S. O. Asplund. Finite boundary value problems solved by Green's matrix. *Mathematica Scandinavica*, 7:49–56, 1959.
[2] I Gohberg, MA Kaashoek, and F Van Schagen. Non-compact integral operators with semi-separable kernels and their discrete analogues: Inversion and Fredholm properties. *Integral Equations and Operator Theory*, 7(5):642–703, 1984.
[3] Fritz Gesztesy and Konstantin A Makarov. Modified Fredholm determinants for operators with matrix-valued semi-separable integral kernels revisited. *Integral Equations and Operator Theory*, 47(4):457–497, 2003.
[4] SN Roy and AE Sarhan. On inverting a class of patterned matrices. *Biometrika*, pages 227–231, 1956.
[5] SN Roy, BG Greenberg, and AE Sarhan. Evaluation of determinants, characteristic equations and their roots for a class of patterned matrices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 348–359, 1960.
[6] Chandan K Mustafi. The inverse of a certain matrix, with an application. *The Annals of Mathematical Statistics*, pages 1289–1292, 1967.
[7] VR Rao Uppuluri and JA Carpenter. The inverse of a matrix occurring in first-order moving-average models. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 79–82, 1969.
[8] G Greenberg and AHMED E Sarhan. Matrix inversion, its interest and application in analysis of data. *Journal of the American Statistical Association*, pages 755–766, 1959.
[9] Raf Vandebril, Marc Van Barel, Gene Golub, and Nicola Mastronardi. A bibliography on semi-separable matrices*. *Calcolo*, 42(3-4):249–270, 2005.
[10] Y Eidelman and I Gohberg. A modification of the Dewilde–van der Veen method for inversion of finite structured matrices. *Linear Algebra and its Applications*, 343:419–450, 2002.
[11] Ellen Van Camp, Nicola Mastronardi, and Marc Van Barel. Two fast algorithms for solving diagonal-plus-semi-separable linear systems. *Journal of Computational and Applied Mathematics*, 164:731–747, 2004.
[12] Y Eidelman and I Gohberg. Inversion formulas and linear complexity algorithm for diagonal plus semi-separable matrices. *Computers & Mathematics with Applications*, 33(4):69–79, 1997.
[13] I Gohberg, T Kailath, and I Koltracht. Linear complexity algorithms for semi-separable matrices. *Integral Equations and Operator Theory*, 8(6):780–804, 1985.
[14] Jitesh Jain, Hong Li, Cheng-Kok Koh, and Venkataramanan Balakrishnan. O(n) algorithms for banded plus semi-separable matrices. In *Numerical Methods for Structured Matrices and Applications*, pages 347–358. Springer, 2010.
[15] Shiv Chandrasekaran and Ming Gu. Fast and stable algorithms for banded plus semi-separable systems of linear equations. *SIAM Journal on Matrix Analysis and Applications*, 25(2):373–384, 2003.
[16] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*, volume 1. Taylor & Francis, 2002.
[17] Peter J Brockwell. Lévy-driven CARMA processes. *Annals of the Institute of Statistical Mathematics*, 53(1):113–124, 2001.
[18] Peter J Brockwell. On continuous-time threshold ARMA processes. *Journal of Statistical Planning and Inference*, 39(2):291–303, 1994.
[19] Sivaram Ambikasaran. ESS. https://github.com/sivaramambikasaran/ESS, 2014.
[20] George B Rybicki and William H Press. Class of fast methods for processing irregularly sampled or otherwise inhomogeneous one-dimensional data. *Physical review letters*, 74(7):1060, 1995.
[21] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.