Fast Givens Rotation Approach to Second Order Sequential Best Rotation Algorithms

Faizan Khattak, Stephan Weiss, Ian K. Proudler
Department of Electronic & Electrical Engineering, University of Strathclyde, Glasgow G1 1XW, Scotland
{faizan.khattak,stephan.weiss,ian.proudler}@strath.ac.uk

2021 Sensor Signal Processing for Defence Conference (SSPD) | 978-1-6654-3314-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/SSPD51364.2021.9541430

***Abstract*—The second order sequential best rotation (SBR2) algorithm is a popular algorithm to decompose a parahermitian matrix into approximate polynomial eigenvalues and eigenvectors. The work horse behind SBR2 is a Givens rotation interspersed by delay operations. In this paper, we investigate and analyse the application of a fast Givens rotation in order to reduce the computation complexity of SBR2. The proposed algorithm inherits the SBR2's proven convergence to a diagonalised and spectrally majorised solution for the polynomial eigenvalues. We provide some analysis and examples for the execution speed of this fast Givens-based SBR2 compared to a standard SBR2 implementation.**

I. Introduction

For broadband signals $\mathbf{x}[n] \in \mathbb{C}^M$ acquired by an M-element sensor arrays in discrete time $n \in \mathbb{Z}$, the space-time covariance matrix $\boldsymbol{R}[\tau] = \mathcal{E}\{\mathbf{x}[n]\mathbf{x}^{\mathrm{H}}[n-\tau]\}$ captures the complete second order statistics of the data. The explicit lag parameter $\tau \in \mathbb{Z}$ is capable of characterising the correlation between sensor signals by relative time delays, and hence bears information on aspects such as the angle of arrival of a particular source signal. This is different from the narrowband case, where e.g. AoA is resolved simply by phase shifts, and it suffices to consider the instantaneous covariance $\mathbf{R}[0]$. To generalise narrowband optimal solutions, which often are based on the eigenvalue decomposition of $\mathbf{R}[0]$, to the broadband case, techniques have been developed to diagonalise $\mathbf{R}[\tau]$ for every value of τ. Because its z-transform $\boldsymbol{R}(z) = \sum_\tau \mathbf{R}[\tau]z^{-\tau}$, or abbreviated $\boldsymbol{R}(z) \bullet\!\!-\!\!\circ\, \mathbf{R}[\tau]$, is a polynomial matrix, such techniques are referred to as polynomial matrix EVDs [1]–[3].

A polynomial matrix EVD exists in the case of an analytic $\boldsymbol{R}(z)$ that emerges from unmultiplexed data [2], [4], and two main families of algorithms with proven convergence have arisen over the last decade — sequential matrix diagonalisation (SMD, [5], [6]) and the second order sequential best rotation algorithm (SBR2, [1], [7]). Since applications such as subband coding [8], [9], beamforming [10], source separation [11], [12] or speech enhancement [13], [14] depend on low computation cost, various efforts have been directed at numerical [15]–[19] and implementational enhancements [19], [20].

Amongst the numerical enhancements of polynomial matrix EVD algorithms, the Givens rotation has attracted particular attention. The cyclic-by-row approach in [15] limited the number of rotation steps to only implement an approximate EVD within the SMD family of algorithms. Divide-and-conquer schemes [18], [19] have been aiming at reducing the spatial dimension by breaking $\mathbf{R}[\tau]$ into sub-blocks, and hence reducing the cost of matrix multiplications.

In this paper, we employ the idea of fast Givens rotations. Typically a similarity transform by a Givens rotation requires the modification of two rows and two columns of the matrix to be transformed. It is well known that square root- and division-free approaches [21]–[23] lead to simplifications and effectively split the unitary Givens rotation into a simple diagonal matrix and a simplified matrix that will only require to have the multiplications and additions of a Givens operation. In [24], this simplification is extended to successive real-valued Givens rotations and is beneficially applied to EVD and QR calculations. Here, we employ a complex-valued extension of this approach in the context of the SBR2 algorithm to reduce the SBR2's complexity.

Therefore, in the following, Sec. II will first briefly outline the SBR2 algorithm, followed by the introduction of fast Givens rotations — both as single operations and in sequence — in Sec. III. This approach then drives a fast SBR2 version introduced in Sec. IV, which is evaluated in a numerical example and simulations in Sec. V. Finally, conclusions are drawn in Sec. VI.

II. Polynomial EVD Algorithms

A. Polynomial EVD

The cross-spectral density (CSD) matrix $\boldsymbol{R}(z)$ satisfies the parahermitian property $\boldsymbol{R}^{\mathrm{P}}(z) = \boldsymbol{R}^{\mathrm{H}}(1/z^*) = \boldsymbol{R}(z)$, and admits a parahermitian matrix EVD (PhEVD) $\boldsymbol{R}(z) = \boldsymbol{Q}(z)\boldsymbol{\Lambda}(z)\boldsymbol{Q}^{\mathrm{P}}(z)$, if $\boldsymbol{R}(z)$ is analytic and the measurement vector $\boldsymbol{x}[n]$ does emerge from multiplexed data [2], [4]. In this case, the factor $\boldsymbol{Q}(z)$ is paraunitary, i.e. $\boldsymbol{Q}(z)\boldsymbol{Q}^{\mathrm{P}}(z) = \mathbf{I}$, and contains analytic eigenvectors in its columns. The corresponding analytic eigenvalues form the diagonal parahermitian matrix $\boldsymbol{\Lambda}(z)$.

A polynomial eigenvalue decomposition [1] is a modified version of the PhEVD,

$$\boldsymbol{R}(z) \approx \boldsymbol{U}(z)\boldsymbol{\Gamma}(z)\boldsymbol{U}^{\mathrm{P}}(z)\,, \tag{1}$$

where the potentially transcendental factors $\boldsymbol{Q}(z)$ and $\boldsymbol{\Lambda}(z)$ of the PhEVD are replaced by a polynomial paraunitary $\boldsymbol{U}(z)$ and a Laurent polynomial, diagonal, and parahermitian $\boldsymbol{\Gamma}(z)$.

We gratefully acknowledge support through Commonwealth Scholarship PKCS-2020-518. This work was also supported in parts by the Engineering and Physical Sciences Research Council (EPSRC) Grant number EP/S000631/1 and the MOD University Defence Research Collaboration in Signal Processing.

Specifically, the analytic $\mathbf{\Lambda}(z)$ is replaced by a spectrally majorised $\mathbf{\Gamma}(z) = \mathrm{diag}\{\gamma_1(z), \ldots, \gamma_M(z)\}$, such that on the unit circle

$$\gamma_{m+1}(\mathrm{e}^{\mathrm{j}\Omega}) \geq \gamma_m(\mathrm{e}^{\mathrm{j}\Omega}) \; \forall \Omega \in \mathbb{R}, \; m = 1, \ldots, (M-1) \; . \quad (2)$$

This spectral majorisation yields eigenvalues that are different from the PhEVD if the analytic eigenvalues intersect. In this case, the polynomial EVD in (1) will require a high polynomial order for a satisfactory approximation, but is important in applications such as subband coding, where the majorisation property in (2) guarantees the maximisation of the coding gain [7], [25].

The PEVD can be calculated by a number of algorithms with guaranteed convergence. A family of second order sequential best rotation (SBR2) algorithms [1], [7], [26] aims to iteratively eliminate the largest off-diagonal component at every iteration step, whereby spectral majorisation has been shown to be enforced [27]. More recently, sequential matrix diagonalisation (SMD) algorithms [5], [6] iteratively diagonalise a time-shifted version of the space-time covariance, which converges faster per iteration but at an overall higher computational cost than SBR2.

B. Second Order Sequential Best Rotation Algorithm

SBR2 is a generalisation of the Jacobi method, whereby successive Givens rotations are interspersed by delay operations in order to move maximum off-diagonal elements to the lag zero component, where they subsequently are eliminated.

Starting from $\boldsymbol{S}_0(z) = \boldsymbol{R}(z)$, at the ith iteration SBR2 finds the maximum off-diagonal element

$$\{\tau_i, m_i, n_i\} = \arg \max_{\substack{\tau, m, n \\ m \neq n}} |s_{i-1,m,n}[\tau]| \; , \quad (3)$$

where $s_{i-1,m,n}[\tau]$ is the element in the mth row and nth column of $\mathbf{S}_{i-1}[\tau]$, whereby $\mathbf{S}_{i-1}[\tau] \circ\!\!-\!\!\bullet \, \boldsymbol{S}_{i-1}(z)$. A delay (or advance, for $\tau_i < 0$) operation

$$\mathbf{\Delta}_i(z) = \mathrm{diag}\{\underbrace{1, \ldots, 1}_{n_i - 1}, \, z^{-\tau_i}, \, \underbrace{1, \ldots, 1}_{M - n_i}\} \quad (4)$$

then ensures that $s_{i-1,m_i,n_i}[\tau_i]$ as well as $s_{i-1,n_i,m_i}[-\tau_i] = s^*_{i-1,m_i,n_i}[\tau_i]$ are moved to lag zero by the similarity transform $\boldsymbol{S}_{i-\frac{1}{2}}(z) = \mathbf{\Delta}_i(z) \boldsymbol{S}_{i-1}(z) \mathbf{\Delta}^{\mathrm{P}}{}_i(z)$.

The energy of the terms $s_{i-1,m_i,n_i}[\tau_i]$ and $s_{i-1,n_i,m_i}[-\tau_i]$ is then transferred onto the diagonal via a Givens rotation $\mathbf{V}_i$,

$$\mathbf{V}_i = \begin{bmatrix} \mathbf{I}_1 & & & & \\ & c_i & & \mathrm{e}^{j\gamma_i} s_i & \\ & & \mathbf{I}_2 & & \\ & -\mathrm{e}^{-j\gamma_i} s_i & & c_i & \\ & & & & \mathbf{I}_3 \end{bmatrix} , \quad (5)$$

with elements $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$ only at the intersection of the m_ith and n_ith rows and columns, and $\mathbf{I}_i$, $i = 1, 2, 3$, identity matrices of appropriate dimensions. The values of θ_i and γ_i depend on $\mathbf{S}_{i-\frac{1}{2}}[0]$, with $\mathbf{S}_{i-\frac{1}{2}}[\tau] \circ\!\!-\!\!\bullet \, \boldsymbol{S}_{i-\frac{1}{2}}(z)$ [1], [5]. When completing the ith SBR2 iteration step, therefore

$$\boldsymbol{S}_i(z) = \mathbf{V}_i \boldsymbol{S}_{i-\frac{1}{2}}(z) \mathbf{V}_i^{\mathrm{H}} \; . \quad (6)$$

If after I iterations, the off-diagonal components are suppressed below some given threshold or a maximum number of iterations has been reached, w.r.t. (1) we obtain $\mathbf{\Gamma}(z) = \boldsymbol{S}_I(z)$ and

$$\boldsymbol{U}(z) = \mathbf{V}_I \mathbf{\Delta}_I(z) \cdots \mathbf{V}_2 \mathbf{\Delta}_2(z) \cdot \mathbf{V}_1 \mathbf{\Delta}_1(z) \; . \quad (7)$$

Note that in every iteration step, while energy at lag zero is transferred into the diagonal, Givens rotations are also applied at lags $\tau \neq 0$, where some energy may leak back into the off-diagonal components. Nonetheless, SRB2 has been proven to converge [1], [5], [27]. Since the Givens operation is the central algebraic operation in SBR2, in the following we will investigate a fast version derived for the real-valued case in [24].

III. Fast Givens Rotation

A. Givens Rotation

Within SBR2, the Givens rotation is applied in (6), which alternatively in the time domain requires

$$\mathbf{S}_i[\tau] = \mathbf{V}_i^{\mathrm{H}} \mathbf{S}_{i-\frac{1}{2}}[\tau] \mathbf{V}_i \; . \quad (8)$$

If the support of $\mathbf{S}_{i-\frac{1}{2}}[\tau]$ is restricted to $|\tau_i| \leq T_i$, such that $\mathbf{S}_{i-\frac{1}{2}}[\tau] = \mathbf{0} \; \forall |\tau| > T_i$, then we require to perform $2T_i + 1$ such similarity transforms. Due to the parahermitian nature of $\boldsymbol{S}_{i-\frac{1}{2}}(z)$, T of these transforms will be redundant. With the exception of the lag zero matrix $\mathbf{S}_{i-\frac{1}{2}}[0]$, the remaining coefficients $\mathbf{S}_{i-\frac{1}{2}}[\tau]$ will generally not be Hermitian matrices for $\tau \neq 0$. However, it is only the Hermitian zero-lag matrix $\mathbf{S}_{i-\frac{1}{2}}[0]$ where a pair of off-diagonal components must be eliminated.

If initially we focus on a single operation out off the above $T + 1$ similarity transforms for (8) and drop all subscripts, we typically face the problem $\mathbf{S}' = \mathbf{V}^{\mathrm{H}} \mathbf{S} \mathbf{V}$. The similarity transform based on a Givens rotation will modify the mth and the nth rows and columns of $\mathbf{S}$ to obtain $\mathbf{S}'$, and therefore require approximately $4M$ complex-valued multiply accumulate (MAC) operations, which translates to $16M$ real-valued MACs.

B. Fast Approach

For a faster implementation of the Givens rotation, termed a fast Givens rotation (FGR), [24] contains two interesting ideas. The first is a reduction of computations for a single Givens rotation; a second step, which will be elaborated in Sec. III-C, exploits subsequent savings if several Givens rotations are iterated. The basic idea is to factorise the orthogonal Givens operation into two non-orthogonal ones, where as many elements as possible are set to unity, hence not requiring explicit multiplications.

We first consider a Givens rotation that is applied to a square matrix $\mathbf{B}_0$ such that $\mathbf{B}_1 = \mathbf{V}_1^{\mathrm{H}} \mathbf{B}_0 \mathbf{V}_1$, with $\mathbf{V}_1$ a

Givens rotation matrix as in (5). Then, according to [24] it is possible to find diagonal matrices $\mathbf{D}_0$ and $\mathbf{D}_1$, such that $\mathbf{B}_0 = \mathbf{D}_0\mathbf{A}_0\mathbf{D}_0$, and $\mathbf{B}_1 = \mathbf{D}_1\mathbf{F}_1\mathbf{A}_0\mathbf{F}_1^{\mathrm{H}}\mathbf{D}_1$. One particular and numerically stable choice according to [24] is such that $\mathbf{D}_1$ and $\mathbf{D}_0$ only differ in the mth and nth elements, leading to

$$\mathbf{F}_1 = \begin{bmatrix} \mathbf{I}_1 & & & & \\ & 1 & & f_{1,1} & \\ & & \mathbf{I}_2 & & \\ & f_{1,2} & & 1 & \\ & & & & \mathbf{I}_3 \end{bmatrix} . \tag{9}$$

The angles ϑ and γ that determine $\mathbf{V}_1$ (i.e. (5)) would be calculated from $\mathbf{A}_0 = \mathbf{D}_0^{-1}\mathbf{B}_0\mathbf{D}_0^{-1}$ in the standard Givens fashion, whereby [24] advocates the selection

$$p = \frac{|d_m^2 a_{mm}^2 - d_n^2 a_{nn}^2|}{\sqrt{(d_m^2 a_{mm}^2 - d_n^2 a_{nn}^2)^2 - 4d_m^2 d_n^2 |a_{mn}|^2}} , \tag{10}$$

with $0 \leq p \leq 1$, a_{mn} the element in the mth row and nth column of $\mathbf{A}_0$, and d_i the ith diagonal element of $\mathbf{D}_0$, such that e.g.

$$\cos\vartheta = \sqrt{\frac{1+p}{2}} . \tag{11}$$

Overall, we may select $\mathbf{D}_0 = \mathbf{I}$. Then for $\boldsymbol{F}_1$ to take the form in (9), we require that $\mathbf{D}_1$ matches $\mathbf{D}_0$, but that the mth and nth elements are modified to $d_m \cos\vartheta$ and $d_n \cos\vartheta$, respectively. For the complex-valued case, it can be shown that $\gamma = \angle a_{mn}$ [1]. Further, we have

$$f_{1,1} = \mathrm{e}^{\mathrm{j}\gamma}\frac{d_n}{d_m}\frac{\sin\vartheta}{\cos\vartheta} , \qquad f_{1,2} = -\mathrm{e}^{-\mathrm{j}\gamma}\frac{d_m}{d_n}\frac{\sin\vartheta}{\cos\vartheta} \tag{12}$$

for the simplified matrix $\mathbf{F}_1$.

The application of $\mathbf{F}_1$ requires only half the number of MACs compared to $\mathbf{V}_1$, but the above approach involves some overheads since $\mathbf{V}_1 = \mathbf{D}_1\mathbf{F}_1\mathbf{D}_0^{-1}$, even if $\mathbf{D}_0 = \mathbf{I}$. Nonetheless, computational saving arise [24], and the technique becomes even more powerful in case several Givens rotations need to be executed successively, such as part of an EVD or QR decomposition [28].

C. Successive Application of Fast Givens Rotations

If a number of Givens rotations $\mathbf{V}_k$, $k = 1, \ldots, K$, need to be executed successively, then note from above that [24]

$$\mathbf{V}_K \ldots \mathbf{V}_2\mathbf{V}_1 = \mathbf{D}_K\mathbf{F}_K \ldots \mathbf{F}_2\mathbf{F}_1\mathbf{D}_0^{-1} . \tag{13}$$

Particularly with $\mathbf{D}_0 = \mathbf{I}$, the evaluation is simple, and the only overhead is to track the modifications of $\mathbf{D}_k$, $k = 1, \ldots, K$ which requires a multiplication of the mth and nth element, as indicated above.

IV. Fast Givens Rotation-Based SBR2

A. Modified Fast SBR2 Algorithm

Even though the SBR2 algorithm does not comprise of a simple consecutive application of Givens rotations, we can apply the idea of (13). This is due to the fact that the interspersed delay operations $\mathbf{D}(z)$ in (7) are diagonal matrices, and hence substituting $\mathbf{V}_\ell = \mathbf{D}_\ell\mathbf{F}_\ell\mathbf{D}_{\ell-1}^{-1}$, $\ell = 1, \cdots, I$, into (7) leads to

$$\boldsymbol{U}(z) = \mathbf{D}_I\mathbf{F}_I\mathbf{D}_{I-1}^{-1}\boldsymbol{\Delta}_I(z) \cdot \cdots \cdot \mathbf{D}_1\mathbf{F}_1\mathbf{D}_0^{-1}\boldsymbol{\Delta}_1(z) \tag{14}$$

$$= \mathbf{D}_I\mathbf{F}_I\boldsymbol{\Delta}_I(z) \cdots \mathbf{F}_2\boldsymbol{\Delta}_2(z)\mathbf{F}_1\boldsymbol{\Delta}_1(z)\mathbf{D}_0^{-1} . \tag{15}$$

Therefore, the overall operation now consists of a sequence of delay operations $\boldsymbol{\Delta}_\ell(z)$ and simplified matrix operations $\mathbf{F}_\ell$, whereby a multiplication with $\mathbf{F}_\ell$ only requires approximately half the MACS compared to a matrix multiplication with $\mathbf{V}_\ell$. Some book keeping is required to update $\mathbf{D}_0$, $\mathbf{D}_1$ all the way up to $\mathbf{D}_I$, but these changes only ever affect two components of these diagonal quantities at a time.

This results in the modification of the SBR2 algorithm to its fast Givens rotation-based version, with its algorithmic steps outlined in Algorithm 1. The modifications are reflected in the variables $\boldsymbol{S}'_i(z)$ and $\boldsymbol{U}'_i(z)$, which differ from $\boldsymbol{S}_i(z)$ and $\boldsymbol{U}_i(z)$ in the standard SBR2 algorithm. The steps 8–11 and 13, to determine, apply and propagate the diagonal correction matrix, apply the reduced-cost matrix $\mathbf{F}_\ell$, and to finally correct the output also differ from the standard SBR2 method.

Algorithm 1: Fast Givens Rotation-Based SBR2

1: inputs: $\boldsymbol{R}(z)$, $\delta_{\max}$, $I_{\max}$
2: initialise: $\boldsymbol{S}'_0(z) = \boldsymbol{R}(z)$; $\mathbf{D}_0 = \mathbf{I}$; $\mathbf{U}'_0 = \mathbf{I}$; $i = 0$;
3: **repeat**
4: $\quad i \leftarrow i + 1$;
5: $\quad$ find maximum off-diagonal element δ of $\mathbf{D}_{i-1}\boldsymbol{S}'_{i-1}(z)\mathbf{D}_{i-1}^{\mathrm{H}}$ via (3);
6: $\quad$ determine $\boldsymbol{\Delta}_i(z)$;
7: $\quad$ calculate $\boldsymbol{S}'_{i-\frac{1}{2}}(z) = \boldsymbol{\Delta}_i(z)\boldsymbol{S}'_{i-1}(z)\boldsymbol{\Delta}^{\mathrm{P}}{}_i(z)$;
8: $\quad$ determine $f_{i,1}$ and $f_{i,2}$ based on $\mathbf{S}'_{i-\frac{1}{2}}[0]$;
9: $\quad$ calculate $\mathbf{S}'_i(z) = \mathbf{F}_i\boldsymbol{S}'_{i-\frac{1}{2}}(z)\mathbf{F}_i^{\mathrm{H}}$;
10: $\quad$ calculate $\boldsymbol{U}'_i(z) = \mathbf{F}_i\boldsymbol{\Delta}_i(z)\boldsymbol{U}'_{i-1}(z)$;
11: $\quad$ update $\mathbf{D}_i$ based on $\mathbf{D}_{i-1}$ and $\mathbf{S}'_{i-\frac{1}{2}}[0]$;
12: **until** $(|\delta| < \delta_{\max}) \vee (i \geq I_{\max})$.
13: outputs: $\boldsymbol{U}(z) = \mathbf{D}_i\boldsymbol{U}'_i(z)$ and $\boldsymbol{\Gamma}(z) = \mathbf{D}_i\boldsymbol{S}'_i(z)\mathbf{D}_i^{\mathrm{H}}$

B. Convergence

For the convergence of Algorithm 1, the essential detail is that the maximum search in step 5 within $\mathbf{D}_{i-1}\boldsymbol{S}'_{i-1}(z)\mathbf{D}_{i-1}^{\mathrm{H}} = \boldsymbol{S}_{i-1}(z)$ is performed over the same quantity as in the SBR2 algorithm in (3). Therefore the convergence proof of the SBR2 algorithm in [1], [7] holds equally for the fast Givens rotation-based SBR2 version. This also implies that Algorithm 1 is guaranteed to converge to a spectrally majorised solution as defined in (2) [27].

Note that it is not necessary to multiply out $\mathbf{D}_{i-1}\boldsymbol{S}'_{i-1}(z)\mathbf{D}_{i-1}^{\mathrm{H}}$ in Step 5 of Algorithm 1 explicitly; a maximum modulus search can first be performed over the temporal dimension of $\mathbf{S}'_{i-1}[\tau]$, and the matrix-valued result $\mathbf{A}_{\max,\tau}$ can be weighted by $\mathbf{D}_{i-1}\mathbf{A}_{\max,\tau}\mathbf{D}_{i-1}^{\mathrm{H}}$, which due to the Hermitian nature of $\mathbf{A}_{\max,\tau}$ and the exclusion of diagonal terms only takes $\frac{1}{2}M(M-1)$ MACs. The remainder of the

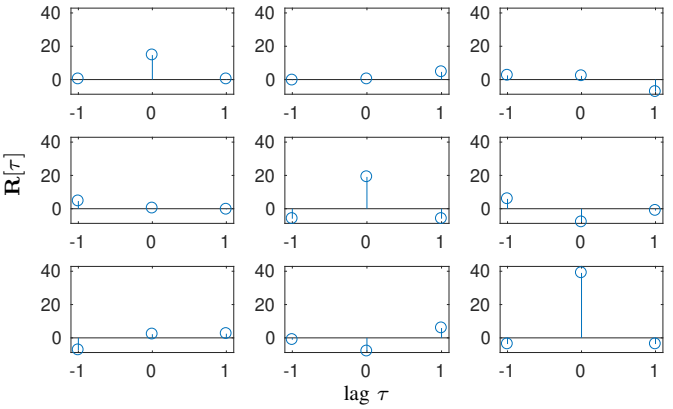

Fig. 1. Plots showing $\mathbf{R}[\tau] \in \mathbb{R}^{3\times 3}$ used as a numerical example; each subplot represents one polynomial entry of $\mathbf{R}[\tau]$.

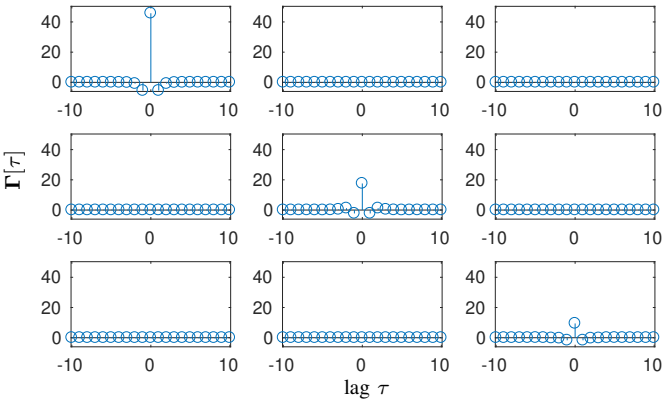

Fig. 2. Approximately diagonalised matrix $\mathbf{\Gamma}[\tau]$, obtained from $\mathbf{R}[\tau]$ in Fig. 1 by the fast Givens rotation-based SBR2 algorithm.

maximum search can then be performed over either the lower left or upper right triangular part of $\mathbf{D}_{i-1}\mathbf{A}_{\max,\tau}\mathbf{D}_{i-1}^{\mathrm{H}}$.

V. Simulation and Results

A. Numerical Example and Convergence

As a numerical example, we utilise the matrix $\boldsymbol{R}(z) : \mathbb{C} \to \mathbb{C}^{3\times 3}$ of order 2, such that $\boldsymbol{R}(z) = \mathbf{R}[-1]z + \mathbf{R}[-0] + \mathbf{R}[1]z^{-1}$. For the matrix-valued coefficients, we have

$$\mathbf{R}[0] = \begin{bmatrix} 14.7 & 0.3 & 2.2 \\ 0.3 & 19.1 & -8.0 \\ 2.2 & -8.0 & 39.0 \end{bmatrix} \tag{16}$$

$$\mathbf{R}[1] = \begin{bmatrix} 0.3 & 4.6 & -7.3 \\ -0.4 & -6.0 & -1.1 \\ 2.5 & 5.9 & -.3.7 \end{bmatrix}. \tag{17}$$

Further note that $\mathbf{R}[-1] = \mathbf{R}^{\mathrm{H}}[1]$. This matrix is also characterised in Fig. 1.

Operating on the above $\boldsymbol{R}(z)$, the fast Givens rotation-based SBR2 algorithm converges in $I = 238$ iterations, and for a maximum off-diagonal component threshold $\delta_{\max} = 10^{-5}$ yields the eigenvalues shown in Fig. 2. The extracted eigenvalues $\mathbf{\Gamma}[\tau] \circ\!\!-\!\!\bullet\, \mathbf{\Gamma}(z)$ are Laurent polynomial that decay in both positive and negative lag directions, and Fig. 2 only provides values for the central lags $|\tau| \leq 10$. The spectral majorisation of the eigenvalues as defined in (2), and to which according to [27] and Sec. IV-B the fast Givens rotation-based SBR2 algorithm is guaranteed to converge, is demonstrated in Fig. 3. These power spectral density terms are non-negative real and ordered in power, satisfying (2).

In comparison, for the above $\boldsymbol{R}(z)$, the standard SBR2 algorithm [1], [7] converges in $I = 235$ iterations. The obtained eigenvalues $\mathbf{\Gamma}_{\mathrm{SBR2}}(z)$ are near-identical to those of the proposed algorithm in Figs. 2 and 3, with an error $\sum_\tau \|\mathbf{\Gamma}_{\mathrm{SBR2}}[\tau] - \mathbf{\Gamma}[\tau]\|_{\mathrm{F}}^2$ of -96.9dB, whereby $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. The small difference in the number of iterations and in the obtained eigenvalues—which according to [2], [4] are unique—is likely due to to the numerical differences between the two algorithm versions, even though the fast Givens approach is claimed to be robust to over-and

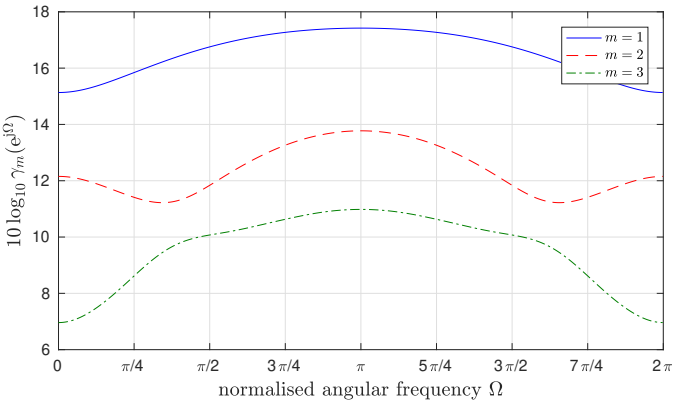

Fig. 3. Eigenvalues $\mathbf{\Gamma}(z)$ as obtained by the fast Givens rotation-based SBR2 algorithm evaluated on the unit circle, $z = \mathrm{e}^{\mathrm{j}\Omega}$.

underflow [24]. The extracted eigenvalues differ by a somewhat larger amount due to their ambiguity w.r.t. multiplications by arbitrary allpass functions [2], [16], and are therefore not shown here.

B. Computational Complexity

In a standard Matlab implementation, matrix-valued operations are favoured and often there is no relation between the sparseness of a matrix and the execution time for its multiplication. Therefore, the run time difference (averaged over 500 runs) for the example in Sec. V-A is only 98.1 ms for the standard SBR2 and 92.6 ms for the proposed fast Givens rotation-based SBR2 version. To investigate the potential of the fast Givens approach for speeding up an implementation in a non-Matlab environment, the fast and standard Givens rotation operations have been explicitly implemented in C and operated from within Matlab through pre-compiled MEX files.

Using Matlab's profiler, the MEX-routines are specifically called to perform $I = 150$ iterations on a matrix $\boldsymbol{R}(z)$ calculated from a ground truth with a diagonal $\mathbf{\Gamma}(z)$ of order 100 and an arbitrary paraunitary matrix $\boldsymbol{U}(z)$ determined from randomised elementary paraunitary operations [29] of order 50. The execution time averaged over 5000 runs for different

TABLE I
EXECUTION TIME COMPARISON BETWEEN STANDARD AND FAST GIVENS ROTATION-BASED SBR2 IMPLEMENTATIONS, SHOWING MEAN PLUS/MINUS ONE STANDARD DEVIATION.

	computation time / [ms]			
method	$M = 3$	$M = 5$	$M = 10$	$M = 20$
standard	1.03±0.02	2.67±0.04	5.18±0.29	14.56±0.22
FGR	0.99±0.03	2.35±0.04	4.66±0.26	13.03±0.26

spatial dimensions M is summarised in Tab. I. The execution includes memory allocation, maximum searches and various book keeping, and therefore is not as dramatic a reduction as direct comparison in MAC operations might suggest. Nevertheless, as M increases, a substantial gap between the run times of the standard and proposed SBR2 implementations emerges, with the computation time shrinking in excess of 10% for the largest matrix size of $M = 20$ that is employed here.

VI. CONCLUSION

This paper has exploited a fast Givens rotation trick to reduce the number of multiply-accumulate operations, particularly when operating on a sequence of Givens rotations. This trick has been adopted for a polynomial matrix eigenvalue decomposition technique known as the second order sequential best rotation algorithm, where the interspersing by delay elements can be absorbed. The modified algorithm will minimise the same cost function as the standard SBR2 algorithm, which in every iteration step will eliminate the maximum off-diagonal component. We have shown that due to the maintenance of the cost function and maximum search, the proposed algorithms inherits the convergence proof and properties of the standard SBR2 algorithm to a diagonalised and spectrally majorised solution for the polynomial eigenvalues. Particularly as the spatial dimension—i.e. the number of sensors recording the data—increases, the computational savings can become significant.

REFERENCES

[1] J. G. McWhirter, P. D. Baxter, T. Cooper, S. Redif, and J. Foster, "An EVD Algorithm for Para-Hermitian Polynomial Matrices," *IEEE Trans SP*, **55**(5):2158–2169, May 2007.

[2] S. Weiss, J. Pestana, and I. K. Proudler, "On the existence and uniqueness of the eigenvalue decomposition of a parahermitian matrix," *IEEE Trans SP*, **66**(10):2659–2672, May 2018.

[3] S. Weiss, I. K. Proudler, and F. K. Coutts, "Eigenvalue decomposition of a parahermitian matrix: Extraction of analytic eigenvalues," *IEEE Trans SP*, **69**:722–737, Jan. 2021.

[4] S. Weiss, J. Pestana, I. Proudler, and F. Coutts, "Corrections to "on the existence and uniqueness of the eigenvalue decomposition of a parahermitian matrix"," *IEEE Trans SP*, **66**(23):6325–6327, Dec. 2018.

[5] S. Redif, S. Weiss, and J. McWhirter, "Sequential matrix diagonalization algorithms for polynomial EVD of parahermitian matrices," *IEEE Trans SP*, **63**(1):81–89, Jan. 2015.

[6] J. Corr, K. Thompson, S. Weiss, J. McWhirter, S. Redif, and I. Proudler, "Multiple shift maximum element sequential matrix diagonalisation for parahermitian matrices," in *IEEE SSP*, Gold Coast, Australia, pp. 312–315, June 2014.

[7] S. Redif, J. McWhirter, and S. Weiss, "Design of FIR paraunitary filter banks for subband coding using a polynomial eigenvalue decomposition," *IEEE Trans SP*, 59(11):5253–5264, Nov. 2011.

[8] S. Weiss, S. Redif, T. Cooper, C. Liu, P. Baxter, and J. McWhirter, "Paraunitary oversampled filter bank design for channel coding," *EURASIP J. Advances Signal Processing*, vol. 2006, pp. 1–10, 2006.

[9] N. Moret, A. Tonello, and S. Weiss, "Mimo precoding for filter bank modulation systems based on PSVD," in *IEEE 73rd Vehicular Technology Conference*, May 2011.

[10] S. Weiss, S. Bendoukha, A. Alzin, F. Coutts, I. Proudler, and J. Chambers, "MVDR broadband beamforming using polynomial matrix techniques," in *EUSIPCO*, Nice, France, pp. 839–843, Sep. 2015.

[11] S. Redif, S. Weiss, and J. McWhirter, "Relevance of polynomial matrix decompositions to broadband blind signal separation," *Signal Processing*, **134**:76–86, May 2017.

[12] S. Weiss, C. Delaosa, J. Matthews, I. Proudler, and B. Jackson, "Detection of weak transient signals using a broadband subspace approach," in *Int. Conf. Sensor Signal Processing for Defence*, Edinburgh, UK, Sept. 2021.

[13] V. W. Neo, C. Evers, and P. A. Naylor, "Speech enhancement using polynomial eigenvalue decomposition," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, pp. 125–129, Oct. 2019.

[14] A. Hogg, V. Neo, S. Weiss, C. Evers, and P. Naylor, "A polynomial eigenvalue decomposition music approach for broadband sound source localization," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, Oct. 2021.

[15] J. Corr, K. Thompson, S. Weiss, J. McWhirter, and I. Proudler, "Cyclic-by-row approximation of iterative polynomial EVD algorithms," in *Sensor Signal Processing for Defence*, Edinburgh, Scotland, pp. 1–5, Sep. 2014.

[16] J. Corr, K. Thompson, S. Weiss, I. Proudler, and J. McWhirter, "Row-shift corrected truncation of paraunitary matrices for PEVD algorithms," in *EUSIPCO*, Nice, France, pp. 849–853, Sep. 2015.

[17] ——, "Reduced search space multiple shift maximum element sequential matrix diagonalisation algorithm," in *IET/EURASIP Intelligent Signal Processing*, London, UK, Dec. 2015.

[18] F. Coutts, J. Corr, K. Thompson, I. Proudler, and S. Weiss, "Divide-and-conquer sequential matrix diagonalisation for parahermitian matrices," in *Sensor Signal Processing for Defence Conference*, London, UK, pp. 1–5, Dec. 2017.

[19] F. K. Coutts, I. K. Proudler, and S. Weiss, "Efficient implementation of iterative polynomial matrix evd algorithms exploiting structural redundancy and parallelisation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 12, pp. 4753–4766, Dec. 2019.

[20] S. Kasap and S. Redif, "Novel field-programmable gate array architecture for computing the eigenvalue decomposition of para-hermitian polynomial matrices," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 3, pp. 522–536, Mar. 2014.

[21] R. W. Stewart, R. Chapman, and T. S. Durrani, "The Square Root In Signal Processing," in *Real-Time Signal Processing XII*, J. P. Letellier, Ed., vol. 1154, International Society for Optics and Photonics. SPIE, pp. 89 – 101, 1989.

[22] J. Götze and U. Schwiegelshohn, "A square root and division free givens rotation for solving least squares problems on systolic arrays," *SIAM Journal on Scientific and Statistical Computing*, vol. 12, no. 4, pp. 800–807, 1991.

[23] M. Moonen and I. Proudler, "Generating 'fast qr' algorithms using signal flow graph techniques," in *Conference Record of The Thirtieth Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 410–414, Nov. 1996.

[24] W. Rath, "Fast Givens rotations for orthogonal similarity transformations," *Numerische Mathematik*, vol. 40, no. 1, pp. 46–56, 1982.

[25] P. Vaidyanathan, "Theory of optimal orthonormal subband coders," *IEEE Transactions on Signal Processing*, vol. 46, no. 6, pp. 1528–1543, Jun. 1998.

[26] Z. Wang, J. G. McWhirter, J. Corr, and S. Weiss, "Multiple shift second order sequential best rotation algorithm for polynomial matrix EVD," in *23rd European Signal Processing Conference*, , pp. 844–848, Nice, France, Sep. 2015.

[27] J. G. McWhirter and Z. Wang, "A novel insight to the SBR2 algorithm for diagonalising para-hermitian matrices," in *11th IMA Conference on Mathematics in Signal Processing*, Birmingham, UK, Dec. 2016.

[28] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, Maryland: John Hopkins University Press, 1996.

[29] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englewood Cliffs: Prentice Hall, 1993.